TMS320F2838x Real-Time Microcontrollers With Connectivity Manager

Technical Reference Manual

Literature Number: SPRUII0E
MAY 2019 – REVISED SEPTEMBER 2023

Table of Contents

List of Figures

List of Tables

This page intentionally left blank.

Read This First

About This Manual

This Technical Reference Manual (TRM) details the integration, the environment, the functional description, and the programming models for each peripheral and subsystem in the device.

The TRM should not be considered a substitute for the data manual, rather a companion guide that should be used alongside the device-specific data manual to understand the details to program the device. The primary purpose of the TRM is to abstract the programming details of the device from the data manual. This allows the data manual to outline the high-level features of the device without unnecessary information about register descriptions or programming models.

Notational Conventions

This document uses the following conventions.

- Hexadecimal numbers can be shown with the suffix h or the prefix 0x. For example, the following number is 40 hexadecimal (decimal 64): 40h or 0x40.
- Registers in this document are shown in figures and described in tables.
  - Each register figure shows a rectangle divided into fields that represent the fields of the register. Each field is labeled with its bit name, its beginning and ending bit numbers above, and its read/write properties with default reset value below. A legend explains the notation used for the properties.
  - Reserved bits in a register figure can have one of multiple meanings:
    - Not implemented on the device
    - Reserved for future device expansion
    - Reserved for TI testing
    - Reserved configurations of the device that are not supported
  - Writing nondefault values to the Reserved bits could cause unexpected behavior and should be avoided.

Glossary

TI Glossary This glossary lists and explains terms, acronyms, and definitions.

Related Documentation From Texas Instruments

For a complete listing of related documentation and development-support tools for these devices, visit the Texas Instruments website at http://www.ti.com. Additionally, the following documents must be used in conjunction with this TRM:

- *TMS320C28x DSP CPU and Instruction Set Reference Guide*
- *TMS320C28x Floating Point Unit and Instruction Set Reference Guide*

Support Resources

TI E2E™ support forums are an engineer's go-to source for fast, verified answers and design help — straight from the experts. Search existing answers or ask your own question to get the quick design help you need.

Linked content is provided "AS IS" by the respective contributors. They do not constitute TI specifications and do not necessarily reflect TI's views; see TI's Terms of Use.

Trademarks

TI E2E™, C2000™, Code Composer Studio™, and Texas Instruments™ are trademarks of Texas Instruments.
Xilinx™ is a trademark of Advanced Micro Devices, Inc.
USB Specification Revision 2.0™ is a trademark of Compaq Computer Corp.
EtherCAT® and Beckhoff ® are registered trademarks of Beckhoff Automation GmbH.
Arm®, Cortex®, and Arm7® are registered trademarks of Arm Limited (or its subsidiaries).
All trademarks are the property of their respective owners.

Chapter 1

► C28x SYSTEM RESOURCES

The following chapters describe the C28x Configuration and System Resources.

1.1 Technical Reference Manual Overview

The block diagram for the F2838x device is shown in Figure 1-1. This Technical Reference Manual is organized into five major sections:

- C28x SYSTEM RESOURCES

  These chapters describe the C28x CPU subsystem, C28x Boot ROM, device configuration, and other system peripherals.

- ANALOG PERIPHERALS

  These chapters describe the Analog-to-Digital Converter (ADC), Buffered Digital-to-Analog Converter (DAC), Comparator Subsystem (CMPSS), and general analog subsystem configuration.

- CONTROL PERIPHERALS

  These chapters describe the Enhanced Capture (eCAP), High Resolution Capture (HRCAP), Enhanced Pulse Width Modulator (ePWM), Enhanced Quadrature Encoder Pulse (eQEP), and Sigma Delta Filter Module (SDFM) peripherals.

- COMMUNICATION PERIPHERALS

  These chapters describe the communication peripherals available to the C28x subsystem such as the I2C, SCI, FSI, McBSP, PMBUS, and SPI. The CAN, EtherCAT, and USB peripherals are also described in this section and can be assigned to the CM subsystem.

- CONNECTIVITY MANAGER (CM)

  These chapters describe the Connectivity Manager (CM) subsystem as well as the AES, GCRC, CM-I2C, CM-UART, SSI, and Ethernet peripherals.

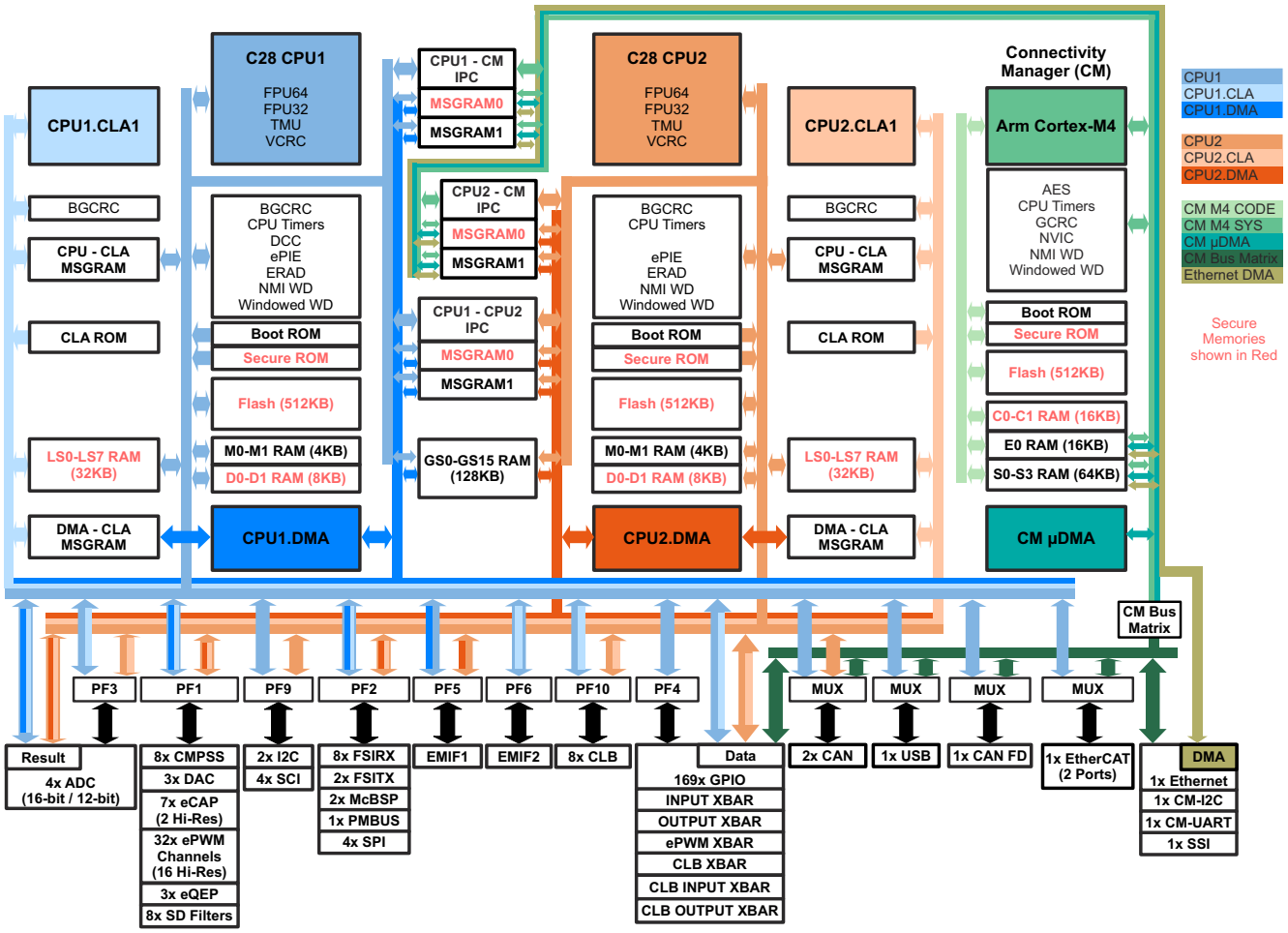


Figure 1-1. F2838x Block Diagram

Chapter 2

C2000™ Microcontrollers Software Support

This chapter discusses the C2000Ware for the C2000™ microcontrollers. The C2000Ware can be downloaded from: www.ti.com/tool/C2000WARE

2.1 Introduction

C2000Ware for the C2000™ microcontrollers is a cohesive set of development software and documentation designed to minimize software development time. From device-specific drivers and libraries to device peripheral examples, C2000Ware provides a solid foundation to begin development and evaluation of your product.

C2000Ware can be downloaded from: www.ti.com/tool/C2000WARE

2.2 C2000Ware Structure

The C2000Ware software package is organized into the following directory structure as shown in Table 2-1.

Table 2-1. C2000Ware Root Directories

Directory Name	Description
boards	Contains the hardware design schematics, BOM, Gerber files, and documentation for C2000 controlCARDS.
device_support	Contains all device-specific support files, bit field headers and device development user's guides.
docs	Contains the C2000Ware package user's guides and the HTML index page of all package documentation.
driverlib	Contains the device-specific driver library and driver-based peripheral examples.
libraries	Contains the device-specific and core libraries.

2.3 Documentation

Within C2000Ware, there is an extensive amount of development documentation ranging from board design documentation, to library user's guides, to driver API documentation. The "boards" directory contains all the hardware design, BOM, Gerber files, and more for controlCARDs. To assist with locating the necessary documentation, an HTML page is provided that contains a full list of all the documents in the C2000Ware package. Locate this page in the "docs" directory.

2.4 Devices

C2000Ware contains the necessary software and documentation to jumpstart development for C2000™ microcontrollers. Each device includes device-specific common source files, peripheral example projects, bit field headers, and if available, a device peripheral driver library. Additionally, documentation is provided for each device on how to set up a CCS project, as well as give an overview of all the included example projects and assist with troubleshooting. For devices with a driver library, documentation is also included that details all the peripheral APIs available.

To learn more about C2000™ microcontrollers, visit: www.ti.com/c2000.

2.5 Libraries

The libraries included in C2000Ware range from fixed-point and floating-point math libraries, to specialized DSP libraries, as well as calibration libraries. Each library includes documentation and examples, where applicable. Additionally, the Flash API files and boot ROM source code are located in the "libraries" directory.

2.6 Code Composer Studio™ Integrated Development Environment (IDE)

Code Composer Studio™ is an integrated development environment (IDE) that supports TI's microcontroller and embedded processors portfolio. The Code Composer Studio™ IDE comprises a suite of tools used to develop and debug embedded applications. The latest version of Code Composer Studio™ IDE can be obtained at: www.ti.com/ccstudio

All projects and examples in C2000Ware are built for and tested with the Code Composer Studio™ IDE. Although the Code Composer Studio™ IDE is not included with the C2000Ware installer, Code Composer Studio™ IDE is easily obtainable in a variety of versions.

2.7 SysConfig and PinMUX Tool

To help simplify configuration challenges and accelerate software development, Texas Instruments™ created SysConfig, an intuitive and comprehensive collection of graphical utilities for configuring pins, peripherals, subsystems, and other components. SysConfig helps you manage, expose, and resolve conflicts visually so that you have more time to create differentiated applications.

The tool's output includes C header and code files that can be used with C2000Ware examples or used to configure custom software.

The SysConfig tool automatically selects the pinmux settings that satisfy the entered requirements. The SysConfig tool is delivered integrated in the Code Composer Studio™ IDE, in the C2000Ware GPIO example, as a standalone installer, or can be used by way of the cloud tools portal at: dev.ti.com

This page intentionally left blank.

Chapter 3

C28x System Control and Interrupts

This chapter explains system control and interrupts for the C28x cores found on this MCU. The system control module configures and manages the overall operation of the device, and provides information about the device status. Configurable features in system control include reset control, NMI operation, peripheral interrupts, power control, clock control, and low-power modes.

3.1 C28x System Control Introduction

On this device, the CPU1 subsystem acts as a master, and by default (upon reset), it owns all the configuration and control. Through software running on CPU1, peripherals and I/Os can be configured to be accessible by the CPU2 subsystem or the CM subsystem and the chosen configurations can be locked.

The PLL clock configuration is also owned by the CPU1 subsystem by default, but a clock control semaphore is provided by which CPU2 can grab access to the clock configuration registers.

Each CPU can be independently configured to accept interrupts from different peripherals. The interrupt path is divided into three stages – the peripheral, the PIE, and the CPU. All stages must be configured and enabled for an interrupt to propagate to the CPU.

Each CPU has its own NMI module to handle different exceptions during run time. If the NMI was on CPU1, any NMI exception that is unhandled before the NMI Watchdog (NMIWD) timer expiration will reset the entire device. If the NMI was on the CPU2 subsystem, then the CPU2 subsystem alone will be reset, in which case the CPU1 subsystem will be informed by another NMI that the CPU2 subsystem was reset because of NMIWD timer expiration.

Each CPU subsystem has its own watchdog timer module for software to use. Watchdog timer expiration on CPU2 will reset the CPU2 subsystem alone when configured to generate a reset, but watchdog timer expiration on CPU1 will reset the entire device.

Except for a CPU2 standalone internal reset such as CPU2.NMIWD or CPU2.WD each time the device is reset, the CPU2 subsystem will be held under reset until the CPU1 subsystem brings it out of reset. This is done by the boot ROM software running on the CPU1 core.

The register space of the device system control module can be found in Section 3.16.

This chapter explains the system control module on both the CPU subsystems.

3.1.1 SYSCTL Related Collateral

Foundational Materials

- *C2000 MCU JTAG Connectivity Debug Application Report*

Getting Started Materials

- C28x Interrupt Nesting
- Debugging JTAG
- *Enhancing Device Security by Using JTAGLOCK Feature Application Report*
- XDS Target Connection Guide

Expert Materials

- *C2000 CPU Memory Built-In Self-Test Application Report*
- *C2000 Memory Power-On Self-Test (M-POST) Application Report*
- *Programming of External Nonvolatile Memory Using SDFlash for TMS320C28x Devices Application Report*
- *Software Phased-Locked Loop (PLL) Design Using C2000 Microcontrollers Application Report*

3.2 System Control Functional Description

The system control module provides the following capabilities:

- Device identification and configuration registers
- Reset control
- Exceptions and Interrupt control
- Safety and error handling features of the device
- Power control
- Clock control
- Low Power modes
- Security module
- Inter-Processor Communication (IPC)

3.2.1 Device Identification

Device identification registers provide information on device class, device family, revision, part number, pin count, and device qualification status.

All of the device information is part of the DEV_CFG_REGS space and is accessible only by the software running on the CPU1 subsystem.

The C28x device identification registers are: PARTIDL, PARTIDH, and REVID.

A 256-bit Unique ID (UID) is available in UID_REGS. The 256 bits are separated into these registers:

- UID_PSRAND0-5: 192 bits of pseudo-random data
- UID_UNIQUE: 32-bit unique data; the value in this register will be unique across all devices with the same PARTIDH
- UID_CHECKSUM: 32-bit fletcher checksum of UID_PSRAND0-5 and UID_UNIQUE and calculated as either little- or big-endian during factory testing
- CPU ID: 16-bit location in OTP. The value at this location provides the information about CPU (CPU1 or CPU2).

3.2.2 Device Configuration Registers

Several registers provide users with configuration information for the C28x subsystems for debug and identification purposes. This information includes the features of the peripherals and how much RAM and FLASH memory is available on this part.

These registers are part of DEV_CFG_REGS space and are accessible only by the software running on the CPU1 subsystem.

- DC0 – DC27: Device Configuration or Capabilities registers.
  If a particular bit in these registers is set to 1, then the associated/feature or module is available in the device.
- PERCNF: Peripheral configuration register.
  This register configures ADC capabilities, and enables or disables the USB internal PHY.
- CPUID: CPU identification register.
  This register is available for software to identify on which CPU the software is executing.

3.3 Resets

This section explains the types and effects of the different resets on this device.

3.3.1 Reset Sources

Table 3-1 summarizes the various reset signals and their effect on the device. CM subsystem resets are described in Section 41.2.

Table 3-1. Reset Signals

Reset Source	CPU1 Core Reset (C28x, TMU, FPU, VCRC)	CPU1 Peripheral Reset	CPU2 Core Reset (C28x, TMU, FPU, VCRC)	CPU2 and CM Peripheral Reset	CPU2 and CM Held In Reset	JTAG / Debug Logic Reset	IOs	XRSn Output
POR	Yes	Yes	Yes	Yes	Yes	Yes	Hi-Z	Yes
XRSn Pin	Yes	Yes	Yes	Yes	Yes	-	Hi-Z	-
CPU1.SIMRESET.XRSn	Yes	Yes	Yes	Yes	Yes	-	Hi-Z	Yes
CPU1.$\overline{\text{WDRS}}$	Yes	Yes	Yes	Yes	Yes	-	Hi-Z	Yes
CPU1.$\overline{\text{NMIWDRS}}$	Yes	Yes	Yes	Yes	Yes	-	Hi-Z	Yes
CPU1.$\overline{\text{SYSRS}}$ (Debugger Reset)	Yes	Yes	Yes	Yes	Yes	-	Hi-Z	-
CPU1.SIMRESET.CPU1RSn	Yes	Yes	Yes	Yes	Yes	-	Hi-Z	-
CPU1.$\overline{\text{SCCRESET}}$	Yes	Yes	Yes	Yes	Yes	-	Hi-Z	-
CPU1.$\overline{\text{HWBISTRS}}$	Yes	-	-	-	-	-	-	-
CPU2.$\overline{\text{SYSRS}}$ (Debugger Reset)	-	-	Yes	Yes	-	-	-	-
CPU2.$\overline{\text{WDRS}}$	-	-	Yes	Yes	-	-	-	-
CPU2.$\overline{\text{NMIWDRS}}$	-	-	Yes	Yes	-	-	-	-
CPU2.$\overline{\text{SCCRESET}}$	-	-	Yes	Yes	-	-	-	-
CPU2.$\overline{\text{HWBISTRS}}$	-	-	Yes	-	-	-	-	-
ECAT_RESET_OUT	Yes	Yes	Yes	Yes	Yes	-	Hi-Z	Yes
TRSTn	-	-	-	-	-	Yes	-	-

Note

After a reset, the reset cause register (RESC) is updated with the reset cause. The bits in this register maintain their state across multiple resets. These can only be cleared by a power-on reset (POR) or by writing 1 to the corresponding bit in RESCCLR register (status can be cleared by writing 1 to RESC register bits also). Each CPU has a RESC register, referred to as CPU1.RESC and CPU2.RESC.

The resets can be divided into a few groups:

- Chip-level resets (XRSn, POR, CPU1.$\overline{\text{WDRS}}$, and CPU1.$\overline{\text{NMIWDRS}}$, SIMRESET.XRSn, ECAT_RESET_OUT (if enabled)), which reset all or almost all of the device.
- System resets (CPU1.$\overline{\text{SYSRS}}$ and CPU1.$\overline{\text{SCCRESET}}$, SIMRESET.CPU1RSn), which reset a large subset of the device but maintain some system-level configuration.
- CPU2 subsystem resets (CPU2.$\overline{\text{SYSRS}}$, CPU2.$\overline{\text{WDRS}}$, CPU2.$\overline{\text{NMIWDRS}}$, and CPU2.$\overline{\text{SCCRESET}}$), which reset only CPU2 and the peripherals owned by it.
- Special resets (CPU1.$\overline{\text{HWBISTRS}}$, CPU2.$\overline{\text{HWBISTRS}}$, and TRSTn), which enable specific device functions.

Whenever the CPU1 subsystem is reset, CPU2, CM and the peripherals owned by them also get reset and held in reset until CPU1 brings CPU2 and CM out of reset by writing to the CPU2RESCTL and CMRESCTL registers respectively. This is done by user application code on CPU1.

Many peripheral modules have individual resets accessible through the system control registers. For information about a module's reset state, refer to the appropriate chapter for that module.

After a POR the boot ROMs clear all of the system and message RAMs on both CPUs.

3.3.2 External Reset (XRSn)

The external reset ($\overline{\text{XRS}}$) is the main chip-level reset for the device. $\overline{\text{XRS}}$ resets both C28x CPUs , the CM subsystem, all peripherals and I/O pin configurations, and most of the system control registers. $\overline{\text{XRS}}$ also holds CPU2 and the CM subsystem in reset. There is a dedicated open-drain pin for XRSn. This pin can be used to drive reset pins for other ICs in the application, and can be driven by an external source. The XRSn is driven internally during watchdog, NMI, and power-on resets.

The XRSn bit in the RESC register is set whenever $\overline{\text{XRS}}$ is driven low for any reason. This bit is then cleared by the boot ROM.

3.3.3 Simulate External Reset

In some cases, there can be a need to simulate the external reset ($\overline{\text{XRS}}$) in software. This can be done by setting XRSn bit to 1 in SIMRESET register by CPU1 software. This toggles $\overline{\text{XRS}}$ pin; hence, resets full device (just like external reset).

After this reset SIMRESET_XRSn bit in the RESC register is set. Software can read this bit to know the cause of reset and clear the status by writing 1 into corresponding bit in RESCCLR register.

3.3.4 Power-On Reset (POR)

The power-on reset (POR) circuit creates a clean reset throughout the device during power-up, suppressing glitches on the GPIOs. The $\overline{\text{XRS}}$ pin is held low for the duration of the POR. In most applications, $\overline{\text{XRS}}$ is held low long enough to reset other system ICs, but some applications may require a longer pulse. In these cases, $\overline{\text{XRS}}$ can be driven low externally to provide the correct reset duration. A POR resets everything that $\overline{\text{XRS}}$ does, along with a few other registers – the reset cause register (RESC), the NMI shadow flag register (NMISHDFLG).

After a POR, the POR and XRSn bits in RESC are set. These bits are then cleared by the boot ROM.

3.3.5 Debugger Reset ($\overline{\text{SYSRS}}$)

During development, it is sometimes necessary to reset the CPU and the peripherals without disconnecting the debugger or disrupting the system-level configuration. To facilitate this, each CPU has a subsystem reset, which can be triggered by a debugger using Code Composer Studio IDE. CPU2 subsystem reset (CPU2.$\overline{\text{SYSRS}}$) resets only CPU2, the peripherals owned by it, clock gating and LPM configuration. It does not hold CPU2 in reset. CPU1 subsystem reset (CPU1.$\overline{\text{SYSRS}}$) resets CPU1, the peripherals owned by it, many system control registers (including its clock gating and LPM configuration and the peripherals' CPU ownership), and all I/O pin configurations. It also produces a CPU2.$\overline{\text{SYSRS}}$ and CM.RESETn and holds both, CPU2 and CM, in reset (CCS Gel file has code to release CPU2 and CM out of reset on CPU1 debug reset).

Neither $\overline{\text{SYSRS}}$ resets the ICEPick debug module, the device capability registers, the clock source and PLL configurations, the missing clock detection state, the PIE vector fetch error handler address, the NMI flags, the analog trims, or anything reset only by a POR (see Section 3.3.4).

3.3.6 Simulate CPU1 Reset

In some cases, there can be a need to simulate the CPU1 reset (CPU1.$\overline{\text{SYSRS}}$) in software. This can be done by setting CPU1RSn bit to 1 in SIMRESET register by CPU1 software. This toggles CPU1.$\overline{\text{SYSRS}}$ signals; hence, resetting CPU1 as well as CPU2 and CM subsystem (just like the debugger reset).

After this reset SIMRESET_CPU1RSn bit in the RESC register is set. Software can read this bit to know the cause of reset and clear the status by writing 1 into corresponding bit in RESCCLR register.

3.3.7 Watchdog Reset (WDRS)

Each CPU has a watchdog timer that can optionally trigger a reset that lasts for 512 INTOSC1 cycles. CPU1 watchdog reset (CPU1.WDRS) produces an XRS. CPU2 watchdog reset (CPU2.WDRS) produces a CPU2.SYSRS and triggers an NMI on CPU1.

After a watchdog reset, the WDRSn bit in RESC is set. Software can read this bit to know the cause of reset and clear the status by writing 1 into corresponding bit in RESCCLR register.

3.3.8 NMI Watchdog Reset (NMIWDRS)

Each CPU has a non-maskable interrupt (NMI) module that detects hardware errors in the system. Each NMI module has a watchdog timer that triggers a reset if the CPU does not respond to an error within a user-specified amount of time. CPU1 NMI watchdog reset (CPU1.NMIWDRS) produces an XRS. CPU2 NMI watchdog reset (CPU2.NMIWDRS) produces a CPU2.SYSRS and triggers an NMI on CPU1.

After an NMI watchdog reset, the NMIWDRSn bit in RESC is set.

3.3.9 Secure Code Copy Reset (SCCRESET)

Dual-zone Code Security Module (DCSM) on this device locks read access to secure memories of each CPU subsystem. To facilitate CRC checks and copying of CLA code, TI provides ROM functions to securely access those memory areas. To prevent security breaches, interrupts must be disabled before calling these functions. If a vector fetch occurs in a secure copy or CRC function, the DCSM triggers a reset. CPU1'security reset (CPU1.SCCRESET) is similar to a CPU1.SYSRS, and CPU2 security reset (CPU2.SCCRESET) is similar to a CPU2.SYSRS. However, the security reset also resets the debug logic to deny access to a potential attacker.

After a security reset, the SCCRESETn bit in RESC is set. Software can read this bit to know the cause of reset and clear the status by writing 1 into corresponding bit in RESCCLR register.

3.3.10 ESC Reset Output

The user can configure the EtherCAT Slave Controller (ESC) module to drive the XRSn pin low whenever the ESC module receives a reset. This is done by setting the DEVICE_RESET_EN bit to 1 in the ECAT_RESET_DEST_CONFIG register of the ESC module (EtherCAT subsystem). By default this is not enabled. Since this toggles the XRSn pin, all effects of an external XRSn reset take effect.

After an ECAT_RESET_OUT reset, the ECAT_RESET_OUT bit in RESC is set. Software can read this bit to know the cause of reset and clear the status by writing 1 into corresponding bit in RESCCLR register.

3.3.11 Test Reset (TRST)

The ICEPick debug module and associated JTAG logic has a reset (TRST) that is controlled by a dedicated pin. This reset is normally active unless the user connects a debugger to the device. For more information on the debug module, see the TI Processors Wiki page on ICEPick: http://processors.wiki.ti.com/index.php/ICEPICK.

The TRST does not have a normal RESC bit, but the TRSTn_pin_status bit indicates the state of the pin.

3.4 Peripheral Interrupts

This section explains the peripheral interrupt handling on the device. Non-maskable interrupts are covered in Section 3.5. Software interrupts and emulation interrupts are not covered in this document. For information on those, see the *TMS320C28x CPU and Instruction Set Reference Guide*.

3.4.1 Interrupt Concepts

An interrupt is a signal that causes the CPU to pause its current execution and branch to a different piece of code known as an interrupt service routine (ISR). This is a useful mechanism for handling peripheral events, and involves less CPU overhead or program complexity than register polling. However, because interrupts are asynchronous to the program flow, care must be taken to avoid conflicts over resources that are accessed both in interrupts and in the main program code.

Interrupts propagate to the CPU through a series of flag and enable registers. The flag registers store the interrupt until it is processed. The enable registers block the propagation of the interrupt. When an interrupt signal reaches the CPU, the CPU fetches the appropriate ISR address from a list called the vector table.

3.4.2 Interrupt Architecture

The C28x CPU has fourteen peripheral interrupt lines. Two of them (INT13 and INT14) are connected directly to CPU timers 1 and 2, respectively. The remaining twelve are connected to peripheral interrupt signals through the enhanced Peripheral Interrupt Expansion module (ePIE, or PIE as a shortened version). The PIE multiplexes up to sixteen peripheral interrupts into each CPU interrupt line. It also expands the vector table to allow each interrupt to have its own ISR. This allows the CPU to support a large number of peripherals.

An interrupt path is divided into three stages – the peripheral, the PIE, and the CPU. Each stage has its own enable and flag registers. This system allows the CPU to handle one interrupt while others are pending, implement and prioritize nested interrupts in software, and disable interrupts during certain critical tasks.

Figure 3-1 shows the interrupt architecture for this device.

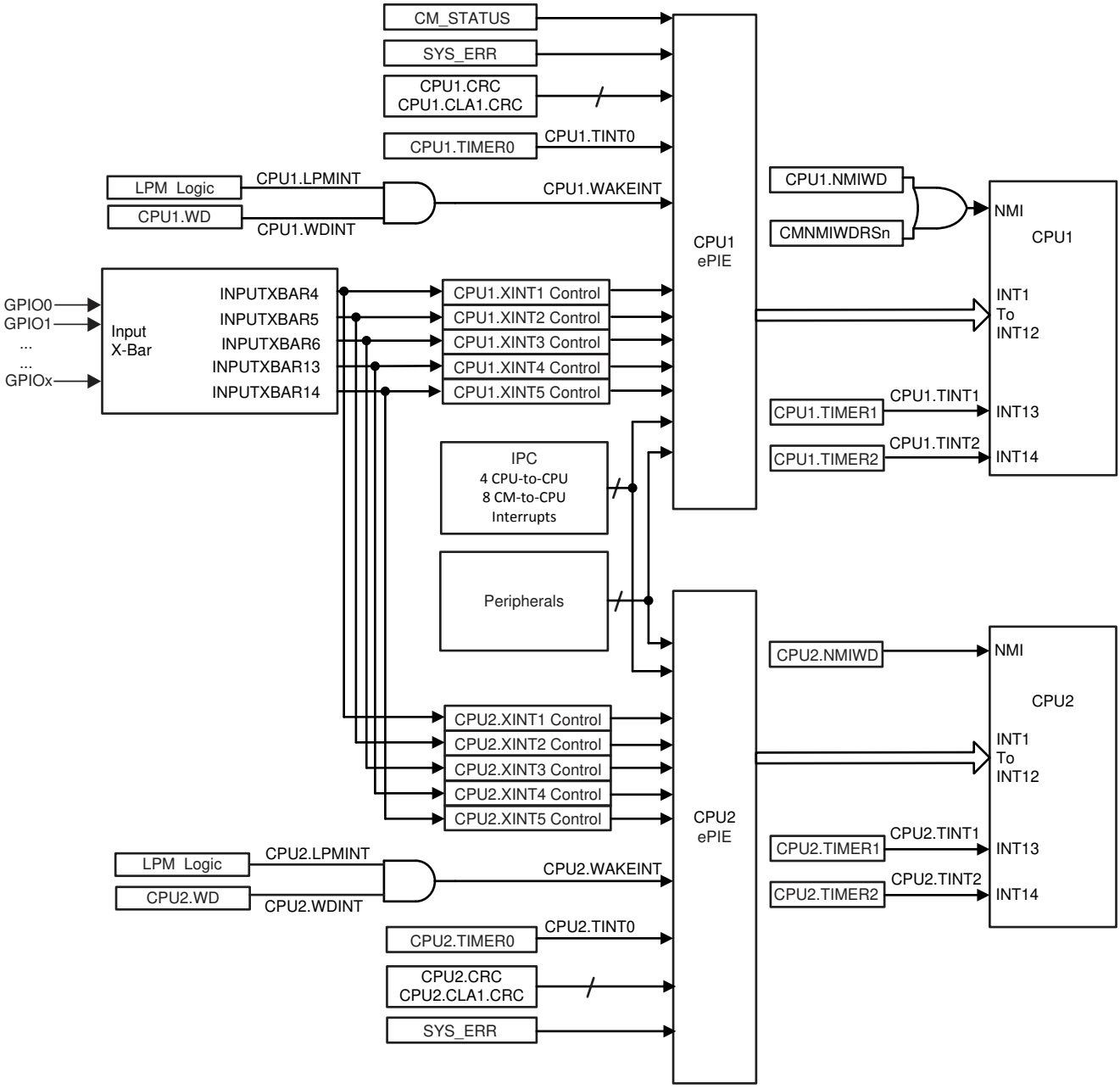


Figure 3-1. Device Interrupt Architecture

3.4.2.1 Peripheral Stage

Each peripheral has its own unique interrupt configuration, which is described in that peripheral's chapter. Some peripherals allow multiple events to trigger the same interrupt signal. For example, a communications peripheral might use the same interrupt to indicate that data has been received or that there has been a transmission error. The cause of the interrupt can be determined by reading the peripheral's status register. Often, the bits in the status register must be cleared manually before another interrupt will be generated.

3.4.2.2 PIE Stage

The PIE provides individual flag and enable register bits for each of the peripheral interrupt signals, which are sometimes called PIE channels. These channels are grouped according to their associated CPU interrupt. Each PIE group has one 16-bit enable register (PIEIERx), one 16-bit flag register (PIEIFRx), and one bit in the PIE acknowledge register (PIEACK). The PIEACK register bit acts as a common interrupt mask for the entire PIE group.

When the CPU receives an interrupt, it fetches the address of the ISR from the PIE. The PIE returns the vector for the lowest-numbered channel in the group that is both flagged and enabled. This gives lower-numbered interrupts a higher priority when multiple interrupts are pending.

If no interrupt is both flagged and enabled, the PIE returns the vector for channel 1. This condition will not happen unless software changes the state of the PIE while an interrupt is propagating. Section 3.4.4 contains procedures for safely modifying the PIE configuration once interrupts have been enabled.

3.4.2.3 CPU Stage

Like the PIE, the CPU provides flag and enable register bits for each of its interrupts. There is one enable register (IER) and one flag register (IFR), both of which are internal CPU registers. There is also a global interrupt mask, which is controlled by the INTM bit in the ST1 register. This mask can be set and cleared using the CPU's SETC instruction. In C code, C2000Ware's DINT and EINT macros can be used for this purpose.

Writes to IER and INTM are atomic operations. In particular, if INTM is cleared, the next instruction in the pipeline will run with interrupts disabled. No software delays are needed.

3.4.2.4 Dual-CPU Interrupt Handling

Each CPU has its own PIE. Both PIEs must be configured independently.

Some interrupts come from shared peripherals that can be owned by either CPU, such as the ADCs and SPIs. These interrupts are sent to both PIEs regardless of the ownership of the peripheral. Thus, a peripheral owned by one CPU can cause an interrupt on the other CPU, if that interrupt is enabled in the other CPU's PIE.

3.4.3 Interrupt Entry Sequence

Figure 3-2 shows how peripheral interrupts propagate to the CPU.

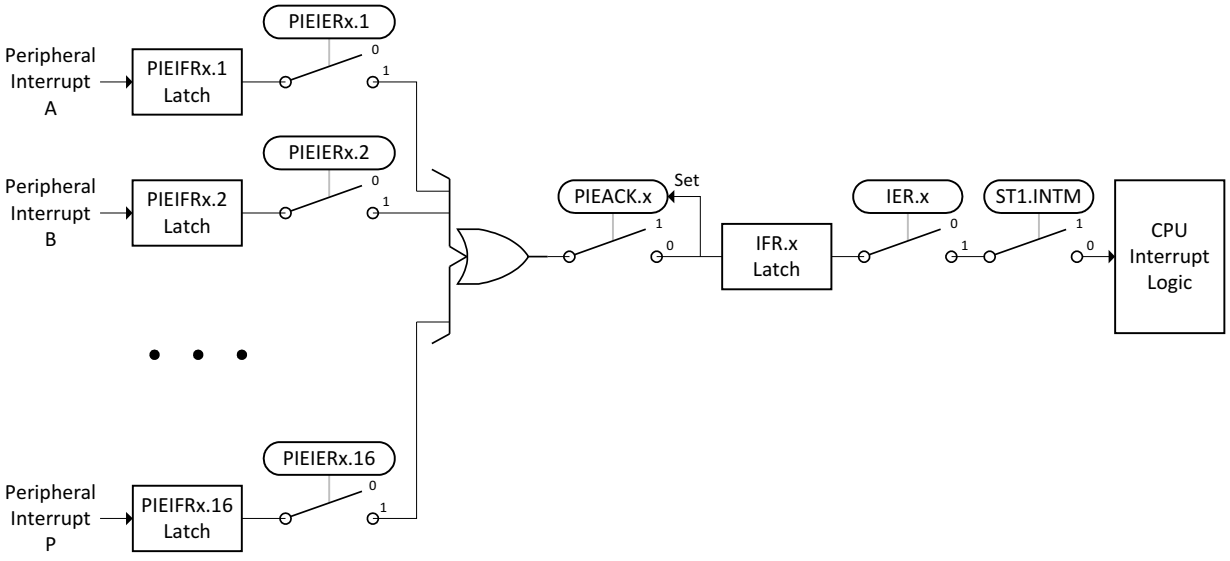


Figure 3-2. Interrupt Propagation Path

When a peripheral generates an interrupt (on PIE group x, channel y), it triggers the following sequence of events:

1. The interrupt is latched in PIEIFRx.y.
2. If PIEIERx.y is set, the interrupt propagates.
3. If PIEACK.x is clear, the interrupt propagates and PIEACK.x is set.
4. The interrupt is latched in IFR.x.
5. If IER.x is set, the interrupt propagates.
6. If INTM is clear, the CPU receives the interrupt.
7. Any instructions in the D2 or later stage of the pipeline are run to completion. Instructions in earlier stages are flushed.
8. The CPU saves its context on the stack.
9. IFR.x and IER.x are cleared. INTM is set. EALLOW is cleared.
10. The CPU fetches the ISR vector from the PIE. PIEIFRx.y is cleared.
11. The CPU branches to the ISR.

The interrupt latency is the time between PIEIFRx.y latching the interrupt and the first ISR instruction entering the execution stage of the CPU pipeline. The minimum interrupt latency is 14 SYSCLK cycles. Wait states on the ISR or stack memories will add to the latency. External interrupts add a minimum of two SYSCLK cycles for GPIO synchronization plus extra time for input qualification (if used). Loops created using the C28x RPT instruction cannot be interrupted.

3.4.4 Configuring and Using Interrupts

At power-up, no interrupts are enabled by default. The PIEIER and IER registers are cleared and INTM is set. The application code is responsible for configuring and enabling all peripheral interrupts.

3.4.4.1 Enabling Interrupts

To enable a peripheral interrupt, perform the following steps:

1. Disable interrupts globally (DINT or SETC INTM).
2. Enable the PIE by setting the ENPIE bit of the PIECTRL register.
3. Write the ISR vector for each interrupt to the appropriate location in the PIE vector table, which is found in Table 3-2.
4. Set the appropriate PIEIERx bit for each interrupt. The PIE group and channel assignments are found in Table 3-2.
5. Set the CPU IER bit for any PIE group containing enabled interrupts.
6. Enable the interrupt in the peripheral.
7. Enable interrupts globally (EINT or CLRC INTM).

Step 4 does not apply to the Timer1 and Timer2 interrupts, which connect directly to the CPU.

3.4.4.2 Handling Interrupts

ISRs are similar to normal functions, but must do the following:

1. Save and restore the state of certain CPU registers (if used).
2. Clear the PIEACK bit for the interrupt group.
3. Return using the IRET instruction.

Requirements 1 and 3 are handled automatically by the TMS320C28x C compiler if the function is defined using the __interrupt keyword. For information on this keyword, see the Keywords section of the *TMS320C28x Optimizing C/C++ Compiler v6.2.4 User's Guide*. For information on writing assembly code to handle interrupts, see the Standard Operation for Maskable Interrupts section of the *TMS320C28x CPU and Instruction Set Reference Guide*.

The PIEACK bit for the interrupt group must be cleared manually in user code. This is normally done at the end of the ISR. If the PIEACK bit is not cleared, the CPU will not receive any further interrupts from that group. This does not apply to the Timer1 and Timer2 interrupts, which do not go through the PIE.

3.4.4.3 Disabling Interrupts

To disable all interrupts, set the CPU's global interrupt mask via DINT or SETC INTM. It is not necessary to add NOPs after setting INTM or modifying IER – the next instruction will execute with interrupts disabled.

Individual interrupts can be disabled using the PIEIERx registers, but care must be taken to avoid race conditions. If an interrupt signal is already propagating when the PIEIER write completes, it may reach the CPU and trigger a spurious interrupt condition. To avoid this, use the following procedure:

1. Disable interrupts globally (DINT or SETC INTM).
2. Clear the PIEIER bit for the interrupt.
3. Wait 5 cycles to make sure that any propagating interrupt has reached the CPU IFR register.
4. Clear the CPU IFR bit for the interrupt's PIE group.
5. Clear the PIEACK bit for the interrupt's PIE group.
6. Enable interrupts globally (EINT or CLRC INTM).

Interrupt groups can be disabled using the CPU IER register. This cannot cause a race condition, so no special procedure is needed.

PIEIFR bits must never be cleared in software since the read/modify/write operation may cause incoming interrupts to be lost. The only safe way to clear a PIEIFR bit is to have the CPU take the interrupt. The following procedure can be used to bypass the normal ISR:

1. Disable interrupts globally (DINT or SETC INTM).
2. Modify the PIE vector table to map the PIEIFR bit's interrupt vector to an empty ISR. This ISR will only contain a return from interrupt instruction (IRET).
3. Disable the interrupt in the peripheral registers.
4. Enable interrupts globally (EINT or CLRC INTM).
5. Wait for the pending interrupt to be serviced by the empty ISR.
6. Disable interrupts globally.
7. Modify the PIE vector table to map the interrupt vector back to its original ISR.
8. Clear the PIEACK bit for the interrupt's PIE group.
9. Enable interrupts globally.

3.4.4.4 Nesting Interrupts

By default, interrupts do not nest. It is possible to nest and prioritize interrupts via software control of the IER and PIEIERx registers. Example code can be found in C2000Ware and documentation is available at software-dl.ti.com/C2000/docs/c28x_interrupt_nesting/html/index.html.

3.4.5 PIE Channel Mapping

Table 3-2 shows the PIE group and channel assignments for each peripheral interrupt. Each row is a group, and each column is a channel within that group. When multiple interrupts are pending, the lowest-numbered channel in the lowest-numbered group is serviced first. Thus, the interrupts at the top of the table have the highest priority, and the interrupts at the bottom have the lowest priority.

Note

Cells marked "-" are Reserved. CPUx is CPU1 for CPU1 PIE and CPU2 for CPU2 PIE.

Table 3-2. PIE Channel Mapping

	INTx.1	INTx.2	INTx.3	INTx.4	INTx.5	INTx.6	INTx.7	INTx.8	INTx.9	INTx.10	INTx.11	INTx.12	INTx.13	INTx.14	INTx.15	INTx.16
INT1.y	ADCA1	ADCB1	ADCC1	XINT1	XINT2	ADCD1	TIMER0	WAKE/ WDINT	I2CA	SYS_ ERR	ECAT SYNC0 (CPU1 only)	ECAT INTn (CPU1 only)	CIPC0	CIPC1	CIPC2	CIPC3
INT2.y	EPWM1_ TZ	EPWM2_ TZ	EPWM3_ TZ	EPWM4_ TZ	EPWM5_ TZ	EPWM6_ TZ	EPWM7_ TZ	EPWM8_ TZ	EPWM9_ TZ	EPWM10_ TZ	EPWM11_ TZ	EPWM12_ TZ	EPWM13_ TZ	EPWM14_ TZ	EPWM15_ TZ	EPWM16_ TZ
INT3.y	EPWM1	EPWM2	EPWM3	EPWM4	EPWM5	EPWM6	EPWM7	EPWM8	EPWM9	EPWM10	EPWM11	EPWM12	EPWM13	EPWM14	EPWM15	EPWM16
INT4.y	ECAP1	ECAP2	ECAP3	ECAP4	ECAP5	ECAP6	ECAP7	-	FSITXA_ INT1	FSITXA_ INT2	FSITXB_ INT1	FSITXB_ INT2	FSIRXA_ INT1	FSIRXA_ INT2	FSIRXB_ INT1	FSIRXB_ INT2
INT5.y	EQEP1	EQEP2	EQEP3	-	CLB1	CLB2	CLB3	CLB4	SDFM1	SDFM2	ECAT RSTINTn (CPU1 only)	ECAT SYNC1 (CPU1 only)	SDFM1 DR1	SDFM1 DR2	SDFM1 DR3	SDFM1 DR4
INT6.y	SPIA_RX	SPIA_TX	SPIB_RX	SPIB_TX	MCBSPA_ RX	MCBSPA_ TX	MCBSPB_ RX	MCBSPB_ TX	SPIC_RX	SPIC_TX	SPID_RX	SPID_TX	SDFM2 DR1	SDFM2 DR2	SDFM2 DR3	SDFM2 DR4
INT7.y	DMA_CH1	DMA_CH2	DMA_CH3	DMA_CH4	DMA_CH5	DMA_CH6	-	-	FSIRXC_ INT1	FSIRXC_ INT2	FSIRXD_ INT1	FSIRXD_ INT2	FSIRXE_ INT1	FSIRXE_ INT2	FSIRXF_ INT1	FSIRXF_ INT2
INT8.y	I2CA	I2CA_ FIFO	I2CB	I2CB_ FIFO	SCIC_RX	SCIC_TX	SCID_RX	SCID_TX	FSIRXG_ INT1	FSIRXG_ INT2	FSIRXH_ INT1	FSIRXH_ INT2	CLB5	CLB6	CLB7	CLB8
INT9.y	SCIA_RX	SCIA_TX	SCIB_RX	SCIB_TX	CANA_0	CANA_1	CANB_0	CANB_1	MCANSS_ INT0 (CPU1 only)	MCANSS_ INT1 (CPU1 only)	MCANSS_ ECC_ CORR_ PUL_INT (CPU1 only)	MCANSS_ WAKE_ AND_TS_ PLS_INT (CPU1 only)	PMBUSA	CM_ STATUS (CPU1 only)	USBA (CPU1 only)	-
INT10.y	ADCA_ EVT	ADCA2	ADCA3	ADCA4	ADCB_ EVT	ADCB2	ADCB3	ADCB4	ADCC_EVT	ADCC2	ADCC3	ADCC4	ADCD_EVT	ADCD2	ADCD3	ADCD4
INT11.y	CLA1_1	CLA1_2	CLA1_3	CLA1_4	CLA1_5	CLA1_6	CLA1_7	CLA1_8	CMTOCPUx IPCINTR0	CMTOCPUx IPCINTR1	CMTOCPUx IPCINTR2	CMTOCPUx IPCINTR3	CMTOCPUx IPCINTR4	CMTOCPUx IPCINTR5	CMTOCPUx IPCINTR6	CMTOCPUx IPCINTR7
INT12.y	XINT3	XINT4	XINT5	MPOST	FMC. DONE	VCRC	FPU OVER FLOW	FPU UNDER FLOW	-	ECAP6 INT2	ECAP7 INT2	-	CPUxCRC_ INT	CLA1CRC_ INT	CLA OVER FLOW	CLA UNDER FLOW

3.4.5.1 PIE Interrupt Priority

3.4.5.1.1 Channel Priority

For every PIE group, the low number channels in the group have the highest priority. For instance in PIE group 1, channel 1.1 has priority over channel 1.3. If those two enabled interrupts occurred simultaneously, channel 1.1 will be serviced first with channel 1.3 left pending. Once the ISR for channel 1.1 completes and provided there are no other enabled and pending interrupts for PIE group 1, channel 1.3 will be serviced. However, for the CPU to service any more interrupts from a PIE group, PIEACK for the group must be cleared. For this specific example, in order for channel 1.3 to be serviced, channel 1.1's ISR has to clear PIEACK for group 1.

The following example describes an alternative scenario: channel 1.1 is currently being serviced by the CPU, channel 1.3 is pending and before channel 1.1's ISR completes, channel 1.2 which is enabled also comes in. Since channel 1.2 has a higher priority than channel 1.3, the CPU will service channel 1.2 and channel 1.3 will still be left pending. Using the steps from the Interrupt Entry Sequence (Section 3.4.3), channel 1.2 interrupt can happen as late as step 10 (The CPU fetches the ISR vector from the PIE. PIEIFRx.y is cleared) and it will still be serviced ahead of channel 1.3.

3.4.5.1.2 Group Priority

Generally, the lowest channel in the lowest PIE group has the highest priority. An example of this is channels 1.1 and 2.1. Those two channels have the highest priority in their respective groups. If the interrupts for those two enabled channels happened simultaneously and provided there are no other enabled and pending interrupts, channel 1.1 will be serviced first by the CPU with channel 2.1 left pending.

However, there are cases where channel priority supersedes group priority. This special case happens depending on which step the CPU is currently at in the Interrupt Entry Sequence (Section 3.4.3).

The following illustrates an example of this special case.

The CPU is about to service channel 2.3 and is currently going through the steps in the Interrupt Entry Sequence (Section 3.4.3).

1. As the CPU reaches step 10 (The CPU fetches the ISR vector from the PIE. PIEIFRx.y is cleared), two enabled interrupts: channel 1.1 and channel 2.1 come in.
2. Due to channel priority, channel 2.1 will be serviced ahead of channel 2.3. However, group priority dictates that channel 1.1 be serviced ahead of channels 2.1 and 2.3.
3. Channel priority supersedes here and channel 2.1 will be serviced ahead of 1.1 and 2.3.
4. After channel 2.1 completes, channel 1.1 is serviced followed by channel 2.3.

Group priority is only guaranteed if no interrupts are currently being serviced, that is, the Interrupt Entry Sequence (Section 3.4.3) is not executing.

3.4.6 System Error and CM Status Interrupts

SYS_ERR and CM_STATUS consolidate several sources of interrupts. These sources set the respective bit in the SYS_ERR_INT_FLG and CM_STATUS_INT_FLG registers. Any set bit in the SYS_ERR_INT_FLG and CM_STATUS_INT_FLG registers will also set the GINT (Global Interrupt) bit. GINT has to be cleared before anymore SYS_ERR or CM_STATUS interrupts are generated. If GINT is cleared with the source flags still set, another SYS_ERR or CM_STATUS interrupt will be fired, therefore it is recommended to clear the source flags before clearing GINT.

Figure 3-3 shows the sources for SYS_ERR and CM_STATUS interrupts.

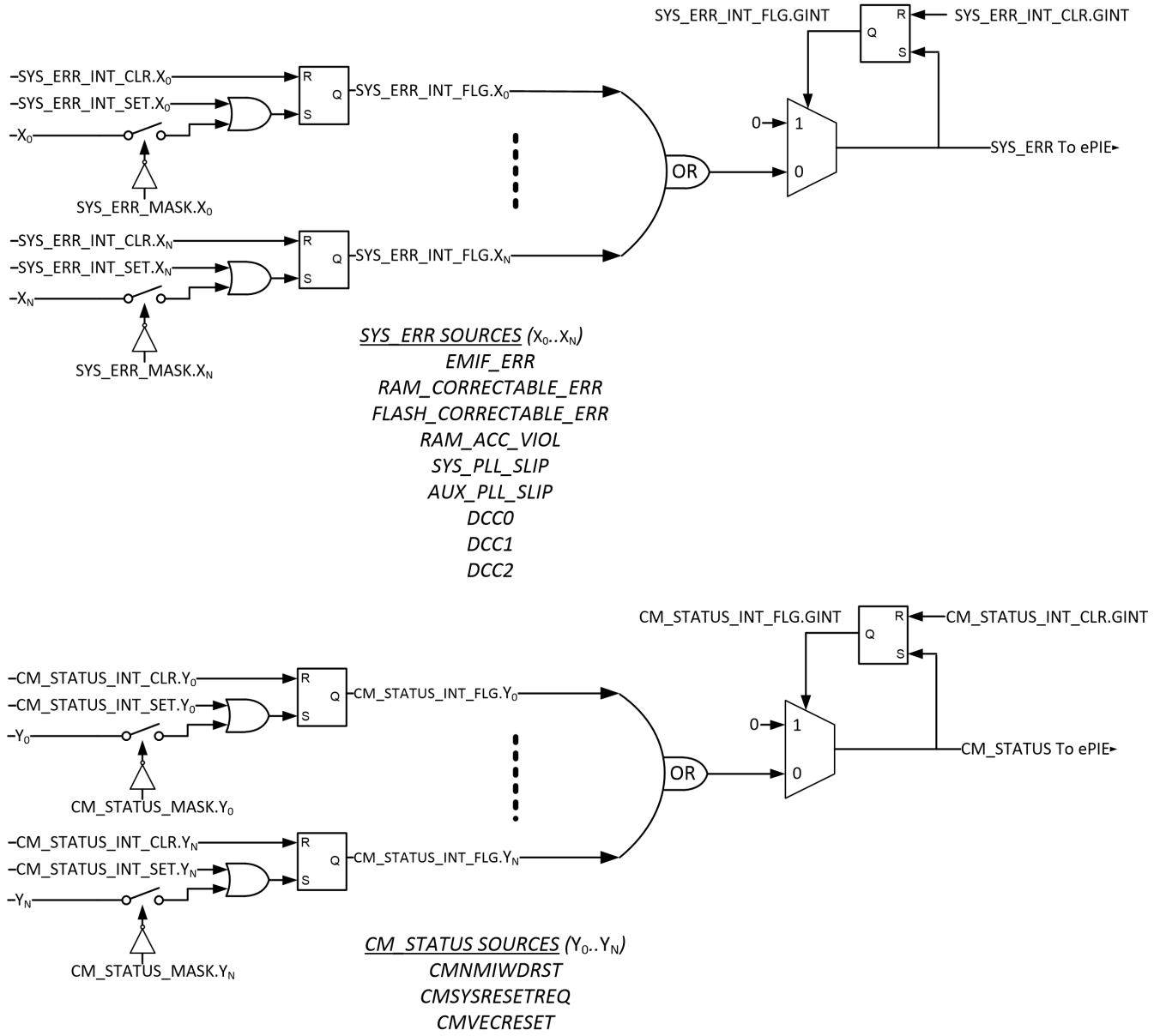


Figure 3-3. System Error and CM Status Interrupt Sources

3.4.7 Vector Tables

Table 3-3 shows the CPU interrupt vector table. The vectors for INT1–INT12 are not used in this device. The reset vector is fetched from the boot ROM instead of from this table.

Table 3-3. CPU Interrupt Vectors

Name	Vector ID	Address	Size (x16)	Description	Core Priority	ePIE Group Priority
Reset	0	0x0000 0D00	2	Reset is always fetched from location 0x003F_FFC0 in Boot ROM	1 (Highest)	-
INT1	1	0x0000 0D02	2	Not used. See PIE Group 1	5	-
INT2	2	0x0000 0D04	2	Not used. See PIE Group 2	6	-
INT3	3	0x0000 0D06	2	Not used. See PIE Group 3	7	-
INT4	4	0x0000 0D08	2	Not used. See PIE Group 4	8	-
INT5	5	0x0000 0D0A	2	Not used. See PIE Group 5	9	-
INT6	6	0x0000 0D0C	2	Not used. See PIE Group 6	10	-
INT7	7	0x0000 0D0E	2	Not used. See PIE Group 7	11	-
INT8	8	0x0000 0D10	2	Not used. See PIE Group 8	12	-
INT9	9	0x0000 0D12	2	Not used. See PIE Group 9	13	-
INT10	10	0x0000 0D14	2	Not used. See PIE Group 10	14	-
INT11	11	0x0000 0D16	2	Not used. See PIE Group 11	15	-
INT12	12	0x0000 0D18	2	Not used. See PIE Group 12	16	-
INT13	13	0x0000 0D1A	2	CPU TIMER1 Interrupt	17	-
INT14	14	0x0000 0D1C	2	CPU TIMER2 Interrupt (for TI/RTOS use)	18	-
DATALOG	15	0x0000 0D1E	2	CPU Data Logging Interrupt	19 (Lowest)	-
RTOSINT	16	0x0000 0D20	2	CPU Real-Time OS Interrupt	4	-
RSVD	17	0x0000 0D22	2	Reserved	2	-
NMI	18	0x0000 0D24	2	Non-Maskable Interrupt	3	-
ILLEGAL	19	0x0000 0D26	2	Illegal Instruction (ITRAP)	-	-
USER 1	20	0x0000 0D28	2	User-Defined Trap	-	-
USER 2	21	0x0000 0D2A	2	User-Defined Trap	-	-
USER 3	22	0x0000 0D2C	2	User-Defined Trap	-	-
USER 4	23	0x0000 0D2E	2	User-Defined Trap	-	-
USER 5	24	0x0000 0D30	2	User-Defined Trap	-	-
USER 6	25	0x0000 0D32	2	User-Defined Trap	-	-
USER 7	26	0x0000 0D34	2	User-Defined Trap	-	-
USER 8	27	0x0000 0D36	2	User-Defined Trap	-	-
USER 9	28	0x0000 0D38	2	User-Defined Trap	-	-
USER 10	29	0x0000 0D3A	2	User-Defined Trap	-	-
USER 11	30	0x0000 0D3C	2	User-Defined Trap	-	-
USER 12	31	0x0000 0D3E	2	User-Defined Trap	-	-

Table 3-4 shows the PIE vector table.

Table 3-4. PIE Interrupt Vectors

Name	Vector ID	Address	Size (x16)	Description	Core Priority	ePIE Group Priority
PIE Group 1 Vectors - Muxed into CPU INT1						
INT1.1	32	0x0000 0D40	2	ADCA1 interrupt	5	1 (Highest)
INT1.2	33	0x0000 0D42	2	ADCB1 interrupt	5	2
INT1.3	34	0x0000 0D44	2	ADCC1 interrupt	5	3
INT1.4	35	0x0000 0D46	2	XINT1 interrupt	5	4
INT1.5	36	0x0000 0D48	2	XINT2 interrupt	5	5
INT1.6	37	0x0000 0D4A	2	ADCD1 interrupt	5	6
INT1.7	38	0x0000 0D4C	2	TIMER0 interrupt	5	7
INT1.8	39	0x0000 0D4E	2	WAKE/WD interrupt	5	8
INT1.9	128	0x0000 0E00	2	I2CA interrupt	5	9
INT1.10	129	0x0000 0E02	2	SYS_ERR interrupt	5	10
INT1.11	130	0x0000 0E04	2	ECAT SYNC0 interrupt (CPU1 only)	5	11
INT1.12	131	0x0000 0E06	2	ECAT interrupt n (CPU1 only)	5	12
INT1.13	132	0x0000 0E08	2	CIPC0 interrupt	5	13
INT1.14	133	0x0000 0E0A	2	CIPC1 interrupt	5	14
INT1.15	134	0x0000 0E0C	2	CIPC2 interrupt	5	15
INT1.16	135	0x0000 0E0E	2	CIPC3 interrupt	5	16 (Lowest)
PIE Group 2 Vectors - Muxed into CPU INT2						
INT2.1	40	0x0000 0D50	2	EPWM1_TZ interrupt	6	1 (Highest)
INT2.2	41	0x0000 0D52	2	EPWM2_TZ interrupt	6	2
INT2.3	42	0x0000 0D54	2	EPWM3_TZ interrupt	6	3
INT2.4	43	0x0000 0D56	2	EPWM4_TZ interrupt	6	4
INT2.5	44	0x0000 0D58	2	EPWM5_TZ interrupt	6	5
INT2.6	45	0x0000 0D5A	2	EPWM6_TZ interrupt	6	6
INT2.7	46	0x0000 0D5C	2	EPWM7_TZ interrupt	6	7
INT2.8	47	0x0000 0D5E	2	EPWM8_TZ interrupt	6	8
INT2.9	136	0x0000 0E10	2	EPWM9_TZ interrupt	6	9
INT2.10	137	0x0000 0E12	2	EPWM10_TZ interrupt	6	10
INT2.11	138	0x0000 0E14	2	EPWM11_TZ interrupt	6	11
INT2.12	139	0x0000 0E16	2	EPWM12_TZ interrupt	6	12
INT2.13	140	0x0000 0E18	2	EPWM13_TZ interrupt	6	13
INT2.14	141	0x0000 0E1A	2	EPWM14_TZ interrupt	6	14
INT2.15	142	0x0000 0E1C	2	EPWM15_TZ interrupt	6	15
INT2.16	143	0x0000 0E1E	2	EPWM16_TZ interrupt	6	16 (Lowest)
PIE Group 3 Vectors - Muxed into CPU INT3						
INT3.1	48	0x0000 0D60	2	EPWM1 interrupt	7	1 (Highest)
INT3.2	49	0x0000 0D62	2	EPWM2 interrupt	7	2
INT3.3	50	0x0000 0D64	2	EPWM3 interrupt	7	3
INT3.4	51	0x0000 0D66	2	EPWM4 interrupt	7	4
INT3.5	52	0x0000 0D68	2	EPWM5 interrupt	7	5
INT3.6	53	0x0000 0D6A	2	EPWM6 interrupt	7	6
INT3.7	54	0x0000 0D6C	2	EPWM7 interrupt	7	7

Table 3-4. PIE Interrupt Vectors (continued)

Name	Vector ID	Address	Size (x16)	Description	Core Priority	ePIE Group Priority
INT3.8	55	0x0000 0D6E	2	EPWM8 interrupt	7	8
INT3.9	144	0x0000 0E20	2	EPWM9 interrupt	7	9
INT3.10	145	0x0000 0E22	2	EPWM10 interrupt	7	10
INT3.11	146	0x0000 0E24	2	EPWM11 interrupt	7	11
INT3.12	147	0x0000 0E26	2	EPWM12 interrupt	7	12
INT3.13	148	0x0000 0E28	2	EPWM13 interrupt	7	13
INT3.14	149	0x0000 0E2A	2	EPWM14 interrupt	7	14
INT3.15	150	0x0000 0E2C	2	EPWM15 interrupt	7	15
INT3.16	151	0x0000 0E2E	2	EPWM16 interrupt	7	16 (Lowest)
PIE Group 4 Vectors - Muxed into CPU INT4						
INT4.1	56	0x0000 0D70	2	ECAP1 interrupt	8	1 (Highest)
INT4.2	57	0x0000 0D72	2	ECAP2 interrupt	8	2
INT4.3	58	0x0000 0D74	2	ECAP3 interrupt	8	3
INT4.4	59	0x0000 0D76	2	ECAP4 interrupt	8	4
INT4.5	60	0x0000 0D78	2	ECAP5 interrupt	8	5
INT4.6	61	0x0000 0D7A	2	ECAP6 interrupt	8	6
INT4.7	62	0x0000 0D7C	2	ECAP7 interrupt	8	7
INT4.8	63	0x0000 0D7E	2	Reserved	8	8
INT4.9	152	0x0000 0E30	2	FSITXA interrupt 1	8	9
INT4.10	153	0x0000 0E32	2	FSITXA interrupt 2	8	10
INT4.11	154	0x0000 0E34	2	FSITXB interrupt 1	8	11
INT4.12	155	0x0000 0E36	2	FSITXB interrupt 2	8	12
INT4.13	156	0x0000 0E38	2	FSIRXA interrupt 1	8	13
INT4.14	157	0x0000 0E3A	2	FSIRXA interrupt 2	8	14
INT4.15	158	0x0000 0E3C	2	FSIRXB interrupt 1	8	15
INT4.16	159	0x0000 0E3E	2	FSIRXB interrupt 2	8	16 (Lowest)
PIE Group 5 Vectors - Muxed into CPU INT5						
INT5.1	64	0x0000 0D80	2	EQEP1 interrupt	9	1 (Highest)
INT5.2	65	0x0000 0D82	2	EQEP2 interrupt	9	2
INT5.3	66	0x0000 0D84	2	EQEP3 interrupt	9	3
INT5.4	67	0x0000 0D86	2	Reserved	9	4
INT5.5	68	0x0000 0D88	2	CLB1 interrupt	9	5
INT5.6	69	0x0000 0D8A	2	CLB2 interrupt	9	6
INT5.7	70	0x0000 0D8C	2	CLB3 interrupt	9	7
INT5.8	71	0x0000 0D8E	2	CLB4 interrupt	9	8
INT5.9	160	0x0000 0E40	2	SDFM1 interrupt	9	9
INT5.10	161	0x0000 0E42	2	SDFM2 interrupt	9	10
INT5.11	162	0x0000 0E44	2	ECATRST interrupt n (CPU1 only)	9	11
INT5.12	163	0x0000 0E46	2	ECATSYNC1 interrupt (CPU1 only)	9	12
INT5.13	164	0x0000 0E48	2	SDFM1DR1 interrupt	9	13
INT5.14	165	0x0000 0E4A	2	SDFM1DR2 interrupt	9	14
INT5.15	166	0x0000 0E4C	2	SDFM1DR3 interrupt	9	15
INT5.16	167	0x0000 0E4E	2	SDFM1DR4 interrupt	9	16 (Lowest)

Table 3-4. PIE Interrupt Vectors (continued)

Name	Vector ID	Address	Size (x16)	Description	Core Priority	ePIE Group Priority
PIE Group 6 Vectors - Muxed into CPU INT6						
INT6.1	72	0x0000 0D90	2	SPIA_RX interrupt	10	1 (Highest)
INT6.2	73	0x0000 0D92	2	SPIA_TX interrupt	10	2
INT6.3	74	0x0000 0D94	2	SPIB_RX interrupt	10	3
INT6.4	75	0x0000 0D96	2	SPIB_TX interrupt	10	4
INT6.5	76	0x0000 0D98	2	MCBSPA_RX interrupt	10	5
INT6.6	77	0x0000 0D9A	2	MCBSPA_TX interrupt	10	6
INT6.7	78	0x0000 0D9C	2	MCBSPB_RX interrupt	10	7
INT6.8	79	0x0000 0D9E	2	MCBSPB_TX interrupt	10	8
INT6.9	168	0x0000 0E50	2	SPIC_RX interrupt	10	9
INT6.10	169	0x0000 0E52	2	SPIC_TX interrupt	10	10
INT6.11	170	0x0000 0E54	2	SPID_RX interrupt	10	11
INT6.12	171	0x0000 0E56	2	SPID_TX interrupt	10	12
INT6.13	172	0x0000 0E58	2	SDFM2DR1 interrupt	10	13
INT6.14	173	0x0000 0E5A	2	SDFM2DR2 interrupt	10	14
INT6.15	174	0x0000 0E5C	2	SDFM2DR3 interrupt	10	15
INT6.16	175	0x0000 0E5E	2	SDFM2DR4 interrupt	10	16 (Lowest)
PIE Group 7 Vectors - Muxed into CPU INT7						
INT7.1	80	0x0000 0DA0	2	DMA_CH1 interrupt	11	1 (Highest)
INT7.2	81	0x0000 0DA2	2	DMA_CH2 interrupt	11	2
INT7.3	82	0x0000 0DA4	2	DMA_CH3 interrupt	11	3
INT7.4	83	0x0000 0DA6	2	DMA_CH4 interrupt	11	4
INT7.5	84	0x0000 0DA8	2	DMA_CH5 interrupt	11	5
INT7.6	85	0x0000 0DAA	2	DMA_CH6 interrupt	11	6
INT7.7	86	0x0000 0DAC	2	Reserved	11	7
INT7.8	87	0x0000 0DAE	2	Reserved	11	8
INT7.9	176	0x0000 0E60	2	FSIRXC interrupt 1	11	9
INT7.10	177	0x0000 0E62	2	FSIRXC interrupt 2	11	10
INT7.11	178	0x0000 0E64	2	FSIRXD interrupt 1	11	11
INT7.12	179	0x0000 0E66	2	FSIRXD interrupt 2	11	12
INT7.13	180	0x0000 0E68	2	FSIRXE interrupt 1	11	13
INT7.14	181	0x0000 0E6A	2	FSIRXE interrupt 2	11	14
INT7.15	182	0x0000 0E6C	2	FSIRXF interrupt 1	11	15
INT7.16	183	0x0000 0E6E	2	FSIRXF interrupt 2	11	16 (Lowest)
PIE Group 8 Vectors - Muxed into CPU INT8						
INT8.1	88	0x0000 0DB0	2	I2CA interrupt	12	1 (Highest)
INT8.2	89	0x0000 0DB2	2	I2CA_FIFO interrupt	12	2
INT8.3	90	0x0000 0DB4	2	I2CB interrupt	12	3
INT8.4	91	0x0000 0DB6	2	I2CB_FIFO interrupt	12	4
INT8.5	92	0x0000 0DB8	2	SCIC_RX interrupt	12	5
INT8.6	93	0x0000 0DBA	2	SCIC_TX interrupt	12	6
INT8.7	94	0x0000 0DBC	2	SCID_RX interrupt	12	7
INT8.8	95	0x0000 0DBE	2	SCID_TX interrupt	12	8
INT8.9	184	0x0000 0E70	2	FSIRXG interrupt 1	12	9

Table 3-4. PIE Interrupt Vectors (continued)

Name	Vector ID	Address	Size (x16)	Description	Core Priority	ePIE Group Priority
INT8.10	185	0x0000 0E72	2	FSIRXG interrupt 2	12	10
INT8.11	186	0x0000 0E74	2	FSIRXH interrupt 1	12	11
INT8.12	187	0x0000 0E76	2	FSIRXH interrupt 2	12	12
INT8.13	188	0x0000 0E78	2	CLB5 interrupt	12	13
INT8.14	189	0x0000 0E7A	2	CLB6 interrupt	12	14
INT8.15	190	0x0000 0E7C	2	CLB7 interrupt	12	15
INT8.16	191	0x0000 0E7E	2	CLB8 interrupt	12	16 (Lowest)
PIE Group 9 Vectors - Muxed into CPU INT9						
INT9.1	96	0x0000 0DC0	2	SCIA_RX interrupt	13	1 (Highest)
INT9.2	97	0x0000 0DC2	2	SCIA_TX interrupt	13	2
INT9.3	98	0x0000 0DC4	2	SCIB_RX interrupt	13	3
INT9.4	99	0x0000 0DC6	2	SCIB_TX interrupt	13	4
INT9.5	100	0x0000 0DC8	2	CANA interrupt 0	13	5
INT9.6	101	0x0000 0DCA	2	CANA interrupt 1	13	6
INT9.7	102	0x0000 0DCC	2	CANB interrupt 0	13	7
INT9.8	103	0x0000 0DCE	2	CANB interrupt 1	13	8
INT9.9	192	0x0000 0E80	2	MCANSS interrupt 0 (CPU1 only)	13	9
INT9.10	193	0x0000 0E82	2	MCANSS interrupt 1 (CPU1 only)	13	10
INT9.11	194	0x0000 0E84	2	MCANSS_ECC_CORR_ PUL interrupt (CPU1 only)	13	11
INT9.12	195	0x0000 0E86	2	MCANSS_WAKE_AND_ TS_PLS interrupt (CPU1 only)	13	12
INT9.13	196	0x0000 0E88	2	PMBUSA interrupt	13	13
INT9.14	197	0x0000 0E8A	2	CM_STATUS interrupt (CPU1 only)	13	14
INT9.15	198	0x0000 0E8C	2	USBA interrupt (CPU1 only)	13	15
INT9.16	199	0x0000 0E8E	2	Reserved	13	16 (Lowest)
PIE Group 10 Vectors - Muxed into CPU INT10						
INT10.1	104	0x0000 0DD0	2	ADCA_EVT interrupt	14	1 (Highest)
INT10.2	105	0x0000 0DD2	2	ADCA2 interrupt	14	2
INT10.3	106	0x0000 0DD4	2	ADCA3 interrupt	14	3
INT10.4	107	0x0000 0DD6	2	ADCA4 interrupt	14	4
INT10.5	108	0x0000 0DD8	2	ADCB_EVT interrupt	14	5
INT10.6	109	0x0000 0DDA	2	ADCB2 interrupt	14	6
INT10.7	110	0x0000 0DDC	2	ADCB3 interrupt	14	7
INT10.8	111	0x0000 0DDE	2	ADCB4 interrupt	14	8
INT10.9	200	0x0000 0E90	2	ADCC_EVT interrupt	14	9
INT10.10	201	0x0000 0E92	2	ADCC2 interrupt	14	10
INT10.11	202	0x0000 0E94	2	ADCC3 interrupt	14	11
INT10.12	203	0x0000 0E96	2	ADCC4 interrupt	14	12
INT10.13	204	0x0000 0E98	2	ADCD_EVT interrupt	14	13
INT10.14	205	0x0000 0E9A	2	ADCD2 interrupt	14	14

Table 3-4. PIE Interrupt Vectors (continued)

Name	Vector ID	Address	Size (x16)	Description	Core Priority	ePIE Group Priority
INT10.15	206	0x0000 0E9C	2	ADCD3 interrupt	14	15
INT10.16	207	0x0000 0E9E	2	ADCD4 interrupt	14	16 (Lowest)
PIE Group 11 Vectors - Muxed into CPU INT11						
INT11.1	112	0x0000 0DE0	2	CLA1_1 interrupt	15	1 (Highest)
INT11.2	113	0x0000 0DE2	2	CLA1_2 interrupt	15	2
INT11.3	114	0x0000 0DE4	2	CLA1_3 interrupt	15	3
INT11.4	115	0x0000 0DE6	2	CLA1_4 interrupt	15	4
INT11.5	116	0x0000 0DE8	2	CLA1_5 interrupt	15	5
INT11.6	117	0x0000 0DEA	2	CLA1_6 interrupt	15	6
INT11.7	118	0x0000 0DEC	2	CLA1_7 interrupt	15	7
INT11.8	119	0x0000 0DEE	2	CLA1_8 interrupt	15	8
INT11.9	208	0x0000 0EA0	2	CMTOCPUx IPC interrupt 0	15	9
INT11.10	209	0x0000 0EA2	2	CMTOCPUx IPC interrupt 1	15	10
INT11.11	210	0x0000 0EA4	2	CMTOCPUx IPC interrupt 2	15	11
INT11.12	211	0x0000 0EA6	2	CMTOCPUx IPC interrupt 3	15	12
INT11.13	212	0x0000 0EA8	2	CMTOCPUx IPC interrupt 4	15	13
INT11.14	213	0x0000 0EAA	2	CMTOCPUx IPC interrupt 5	15	14
INT11.15	214	0x0000 0EAC	2	CMTOCPUx IPC interrupt 6	15	15
INT11.16	215	0x0000 0EAE	2	CMTOCPUx IPC interrupt 7	15	16 (Lowest)
PIE Group 12 Vectors - Muxed into CPU INT12						
INT12.1	120	0x0000 0DF0	2	XINT3 interrupt	16	1 (Highest)
INT12.2	121	0x0000 0DF2	2	XINT4 interrupt	16	2
INT12.3	122	0x0000 0DF4	2	XINT5 interrupt	16	3
INT12.4	123	0x0000 0DF6	2	MPOST interrupt	16	4
INT12.5	124	0x0000 0DF8	2	FMC.DONE interrupt	16	5
INT12.6	125	0x0000 0DFA	2	VCRC interrupt	16	6
INT12.7	126	0x0000 0DFC	2	FPU OVERFLOW interrupt	16	7
INT12.8	127	0x0000 0DFE	2	FPU UNDERFLOW interrupt	16	8
INT12.9	216	0x0000 0EB0	2	Reserved	16	9
INT12.10	217	0x0000 0EB2	2	ECAP6 interrupt	16	10
INT12.11	218	0x0000 0EB4	2	ECAP7 interrupt	16	11
INT12.12	219	0x0000 0EB6	2	Reserved	16	12
INT12.13	220	0x0000 0EB8	2	CPUxCRC interrupt	16	13
INT12.14	221	0x0000 0EBA	2	CLA1CRC interrupt	16	14
INT12.15	222	0x0000 0EBC	2	CLA OVERFLOW interrupt	16	15
INT12.16	223	0x0000 0EBE	2	CLA UNDERFLOW interrupt	16	16 (Lowest)

3.5 Exceptions and Non-Maskable Interrupts

This section describes system-level error conditions that can trigger a non-maskable interrupt (NMI). The interrupt allows the application to respond to the error.

3.5.1 Configuring and Using NMIs

Each CPU subsystem has its own NMI module. This section will provide detail of NMI on C28x subsystems. An incoming NMI sets a status bit in the NMIFLG register and starts the NMI watchdog counter. This counter is clocked by the SYSCLK, and if it reaches the value programmed in NMIWDPRD register, it triggers an NMI watchdog reset (NMIWDRS). To prevent this, the NMI handler must clear the flag bit using the NMIFLGCLR register. Once all flag bits are clear, the NMIINT bit in the NMIFLG register should be cleared to allow future NMIs to be taken.

The NMI module is enabled by the boot ROM during the startup process. To respond to NMIs, an NMI handler vector must be written to the PIE vector table.

3.5.2 Emulation Considerations

The NMI watchdog counter behaves as follows under debug conditions:

CPU Suspended:	When the CPU is suspended, the NMI watchdog counter is suspended.
Run-Free Mode:	When the CPU is placed in run-free mode, the NMI watchdog counter resumes operation as normal.
Real-Time Single-Step Mode:	When the CPU is in real-time single-step mode, the NMI watchdog counter is suspended. The counter remains suspended even within real-time interrupts.
Real-Time Run-Free Mode:	When the CPU is in real-time run-free mode, the NMI watchdog counter operates as normal.

3.5.3 NMI Sources

There are several types of hardware errors that can trigger an NMI. Additional information about the error is usually available from the module that detects it.

3.5.3.1 Missing Clock Detection

The missing clock detection logic monitors OSCCLK for failure. If the OSCCLK source stops, the PLL is bypassed, OSCCLK is connected to INTOSC1, and NMIs are fired to both CPUs. For more information on missing clock detection, see Section 3.7.7.1.

3.5.3.2 RAM Uncorrectable Error

A single-bit parity error, double-bit ECC data error, or single-bit ECC address error in a RAM read will trigger an NMI. This applies to CPU, CLA, and DMA reads. Single-bit ECC data errors do not trigger an NMI, but can optionally trigger a normal peripheral interrupt. For more information on RAM error detection, see Section 3.12.1.9.

3.5.3.3 Flash Uncorrectable ECC Error

A double-bit ECC data error or single-bit ECC address error in a Flash read will trigger an NMI. Single-bit ECC data errors do not trigger an NMI, but can optionally trigger a normal peripheral interrupt.

3.5.3.4 ROM Uncorrectable Error

On this device ROM has a parity feature and a parity mismatch during read or execution will trigger an NMI.

3.5.3.5 NMI Vector Fetch Mismatch

Each CPU's Peripheral Interrupt Expansion module (PIE) has redundant vector tables. If a mismatch in these tables is detected during a vector fetch, a user-specified error handler is run instead of the ISR. If the vector fetch was caused by an NMI, a second NMI is fired to the other CPU. Mismatches for other interrupts do not trigger an NMI. For more information about the vector address check, see Section 3.6.2.

3.5.3.6 CPU2 Watchdog or NMI Watchdog Reset

A watchdog reset or NMI watchdog reset on CPU2 will trigger an NMI on CPU1. Since a CPU1 reset also resets CPU2, this NMI source is not available on CPU2. Watchdog interrupts do not trigger an NMI.

3.5.3.7 CM NMI Watchdog Reset

A NMI watchdog reset on CM can generate NMI on CPU1. To enable this feature user need to set CMNMIWDRST configuration bit in CMTOCPU1NMICTL register.

3.5.3.8 EtherCAT Reset out

A reset out from EtherCAT module can generate NMI on CPU1. To enable this feature user need to set CPU_NMI_EN configuration bit in ESCSS_RESET_DEST_CONFIG register.

3.5.3.9 CRC Fail

A CRC fail result from BGCRC module can generate NMI to respective CPU. By default this NMI is enable. To disable this feature user need to configure NMIDIS configuration field in BGCRC_CTRL1 register with value "1010".

3.5.3.10 ERAD NMI

ERAD module can generate NMI based on different events which user can configure in ERAD.

3.5.4 Illegal Instruction Trap (ITRAP)

If the CPU tries to execute an illegal instruction, it generates a special interrupt called an illegal instruction trap (ITRAP). This interrupt is non-maskable and has its own vector in the PIE vector table. For more information about ITRAPs, see the Illegal-Instruction Trap section of the *TMS320C28x DSP CPU and Instruction Set Reference Guide*.

Note

A RAM fetch access violation will trigger an ITRAP in addition to the normal peripheral interrupt for RAM access violations. The CPU will handle the ITRAP first.

3.6 Safety Features

This section gives details on features that monitor device operation during run-time to detect any error in operation.

3.6.1 Write Protection on Registers

3.6.1.1 LOCK Protection on System Configuration Registers

Several system configuration registers are protected from spurious CPU writes by “LOCK” registers. Once these associated LOCK register bits are set, the respective locked registers can no longer be modified by software. See the register descriptions for details.

3.6.1.2 EALLOW Protection

Several control registers are protected from spurious CPU writes by the EALLOW protection mechanism. The EALLOW bit in status register 1 (ST1) indicates the state of protection as shown in Table 3-5.

Table 3-5. Access to EALLOW-Protected Registers

EALLOW Bit	CPU Writes	CPU Reads	JTAG Writes	JTAG Reads
0	Ignored	Allowed	Allowed(1)	Allowed
1	Allowed	Allowed	Allowed	Allowed

(1) The EALLOW bit is overridden via the JTAG port, allowing full access of protected registers during debug from the Code Composer Studio IDE.

At reset, the EALLOW bit is cleared, enabling EALLOW protection. While protected, all writes to protected registers by the CPU are ignored and only CPU reads, JTAG reads, and JTAG writes are allowed. If this bit is set, by executing the EALLOW instruction, the CPU is allowed to write freely to protected registers. After modifying registers, the registers can once again be protected by executing the EDIS instruction to clear the EALLOW bit.

3.6.2 CPU1 and CPU2 ePIE Vector Address Validity Check

The ePIE vector table on each CPU is duplicated into these two parts:

- Main ePIE Vector Table mapped from 0xD00 to 0xEFF in the C28x memory space
- Redundant ePIE Vector Table mapped from 0x1000D00 to 0x1000EFF in the C28x memory space

Following is the behavior of accesses to the ePIE memories:

- Data Writes to Main Vector Table: Writes to both memories
- Data Writes to Redundant Vector Table: Writes only to the Redundant Vector Table
- Vector Fetch: Data from both the vector tables are compared
- Data Read: Can read the Main and Redundant vector table separately

On every vector fetch from the ePIE, a hardware comparison (no cycle penalty is incurred to do the comparison) of both the vector table outputs is performed and if there is a mismatch between the two vector table outputs, the following occurs:

1. If the PIEVERRADDR register (default value 0x3F FFFF) is not initialized, the default error handler at address 0x3FFFBE gets executed.

   But, when the PIEVERRADDR register is initialized to the address of the user-defined routine, the user-defined routine is executed instead of the default error handler.

   Note: Each CPU has a copy of the PIE Vector Fetch Error Handler register (CPU1.PIEVERRADDR and CPU2.PIEVERRADDR).
2. Hardware also generates EPWM Trip signals that trip the PWM outputs using TRIPIN15.
3. An NMI to the other CPU is sent, if the current mismatch is during a vector fetch. For example, on an NMI vector fetch error for CPU2, an NMI is also fired to CPU1.NMIWD.

If there is no mismatch, the correct vector is jammed onto the C28 program control.

3.6.3 NMIWDs

Each CPU has user-programmable NMIWD period registers, in which users can set a limit on how much time they want to allocate for the device to acknowledge the NMI. If the NMI is not acknowledged, it will cause a device reset.

3.6.4 ECC and Parity Enabled RAMs, Shared RAMs Protection

Each CPU subsystem has different RAM blocks. Some RAM blocks are ECC-enabled and others are parity-enabled. All single-bit errors in ECC RAM are auto-corrected and an error counter is incremented every time a single bit error is detected. If the error counter reaches a predefined user configured limit, an interrupt is generated to the corresponding CPU. Refer to Section 3.12 for more details on RAM errors.

All uncorrectable double-bit errors end up triggering an NMI to corresponding CPUs.

3.6.5 ECC Enabled Flash Memory

When ECC is programmed and enabled, Flash single-bit errors are corrected automatically by ECC logic before giving data to the CPU, but they are not corrected in Flash memory. Flash memory will still contain wrong data until another erase/program operation happens to correct the Flash contents. Irrespective of whether the error interrupt is enabled or disabled, single-bit errors are always corrected before giving data to the CPU. When the interrupt is disabled, users can check the single-bit error counter register for any single-bit error occurrences. The error counter stops incrementing once its value is equal to the threshold + 1. It is always suggested to set the threshold register to a non-zero value so that the error counter can increment. It is up to the user to decide the threshold value at which they have to reprogram the Flash with the correct data.

When ECC is programmed and enabled, Flash uncorrectable errors end up triggering an NMI to the respective CPU. Refer to Section 3.12 for more details on Flash error correction and error catching mechanisms.

3.6.6 ERRORSTS Pin

The ERRORSTS pin is an ‘always output’ pin and remains high until an error is detected inside the chip. On an error, the ERRORSTS pin goes low (default polarity) until the corresponding internal error status flag for that error source is cleared. Figure 3-4 shows the functionality of the ERRORSTS pin.

The ERRORSTS pin is tri-stated until the chip power-rails ramp up to the lower operational limit. As the ERRORSTS pin is an active-low pin (default polarity), users who care about the state of this pin during power-up must connect an external pull-down on this pin.

Following enhancement has been made on this device for ERRORSTS pin logic:

- Polarity of Error pin has been made configurable (default setting is active-low polarity).
- To enable testing of the Error pin, capability to force and clear the Error pin from software has been provided.
- Additional sources of Error have been added to ERRORSTS:
  - CPU1 Watchdog reset
  - Error on a PIE vector fetch
  - NMI on CM

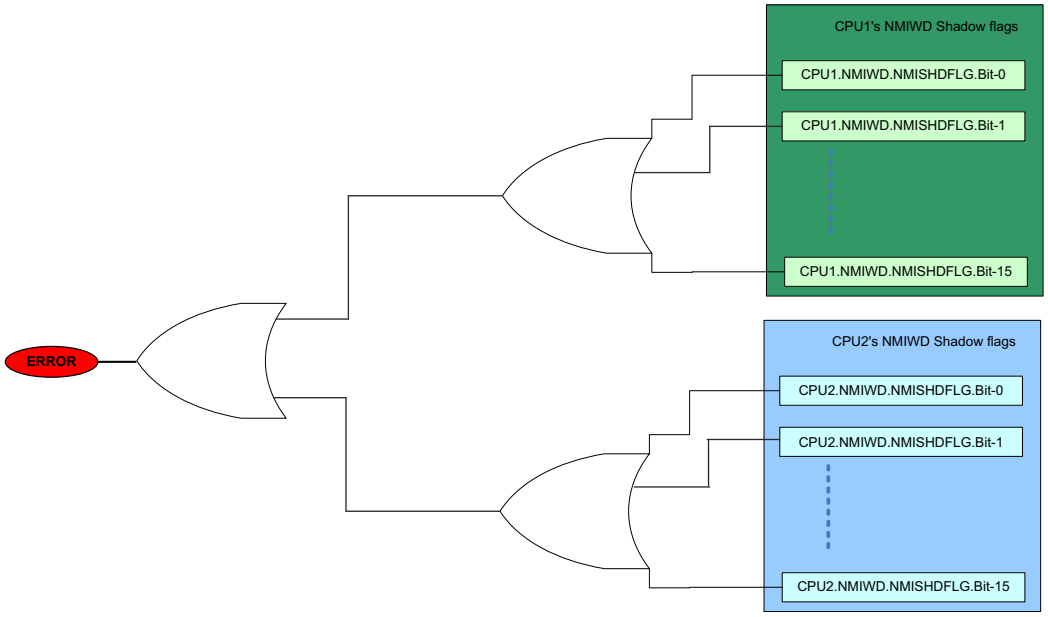


Figure 3-4. ERRORSTS Pin Diagram

3.7 Clocking

This section explains the clock sources and clock domains on this device, and how to configure them for application use. Figure 3-5 provides an overview of the device's clocking system.

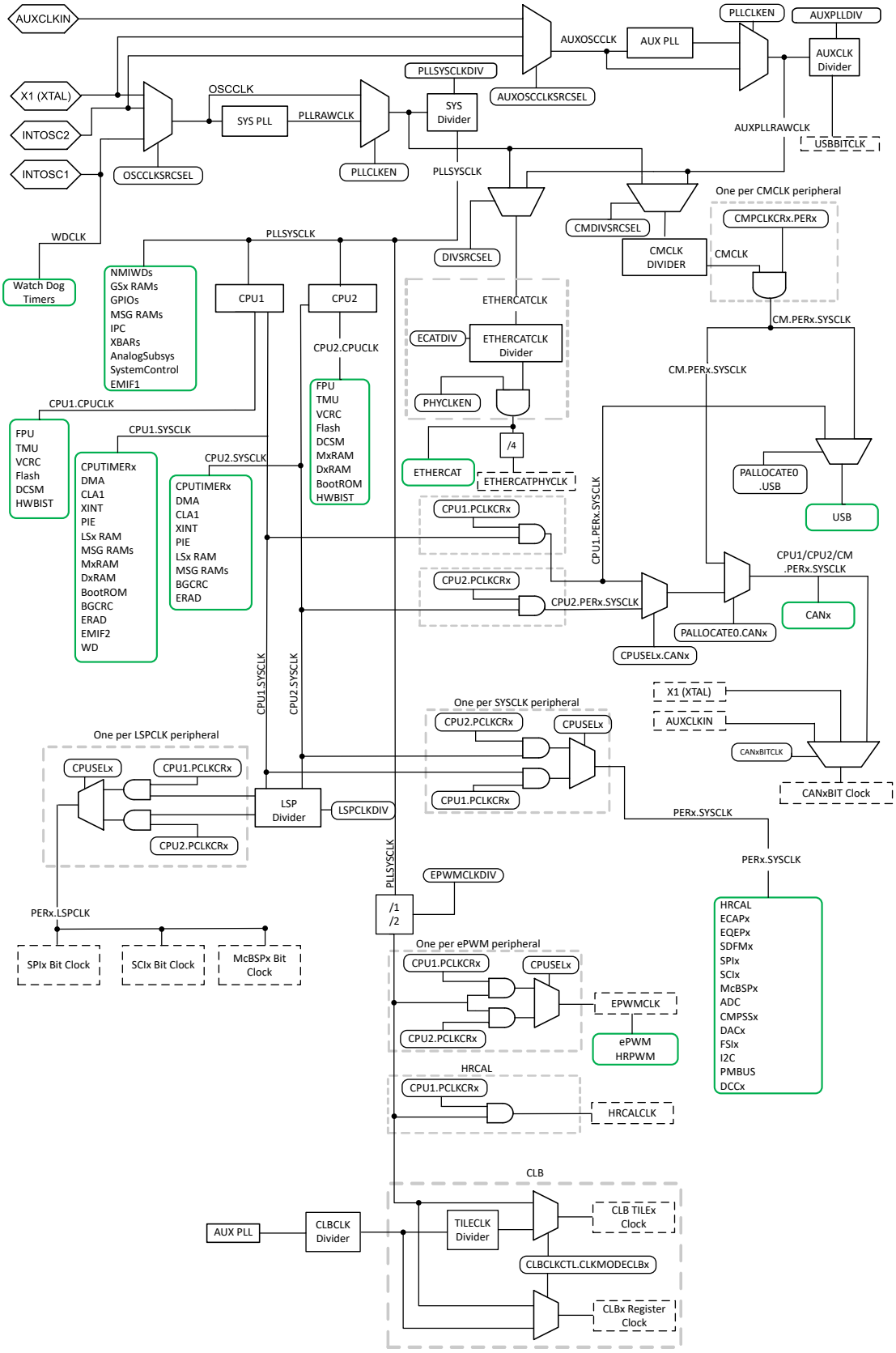


Figure 3-5. Clocking System

3.7.1 Clock Sources

The needs of the application are what ultimately determine the clock configuration. Specific concerns such as application performance, power consumption, total system cost, and EMC are beyond the scope of this document, but must provide answers to the following questions:

1. What is the desired CPU frequency?
2. Are there any additional communication protocols or peripherals, such as CAN or USB, required?
3. What types of external oscillators or clock sources are available?

If CAN or USB is required, an external clock source with a precise frequency must be used as a reference clock. Otherwise, use only INTOSC2 and avoid the need for more external components.

All of the clocks in the device are derived from one of four clock sources.

3.7.1.1 Primary Internal Oscillator (INTOSC2)

At power-up, the device is clocked from an on-chip 10 MHz oscillator (INTOSC2). INTOSC2 is the primary internal clock source, and is the default system clock at reset. It is used to run the boot ROM and can be used as the system clock source for the application.

Note

INTOSC2's frequency tolerance is too loose to meet the timing requirements for some peripherals such as CAN and USB, so an external clock must be used to support those features.

3.7.1.2 Backup Internal Oscillator (INTOSC1)

The device also includes a redundant on-chip 10 MHz oscillator (INTOSC1). INTOSC1 is a backup clock source that normally only clocks the watchdog timers and missing clock detection circuit (MCD). If MCD is enabled and a missing system clock is detected, the system PLL is bypassed and all system clocks are connected to INTOSC1 automatically. INTOSC1 may also be manually selected as the system and auxiliary clock source for debug purposes.

3.7.1.3 External Oscillator (XTAL)

The dedicated X1 and X2 pins support an external clock source (XTAL), which can be used as the main system and auxiliary clock source. Frequency limits and timing requirements can be found in the device datasheet. Three types of external clock sources are supported:

- A single-ended 3.3V external clock. The clock signal should be connected to X1 while X2 is left unconnected, as shown in Figure 3-6.

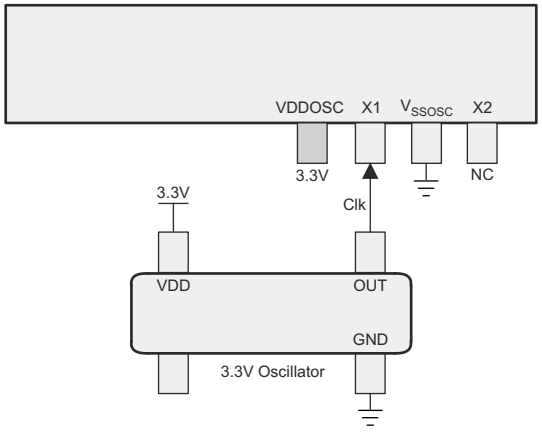


Figure 3-6. Single-ended 3.3V External Clock

- An external crystal. The crystal should be connected across X1 and X2 with its load capacitors connected to VSSOSC as shown in Figure 3-7.

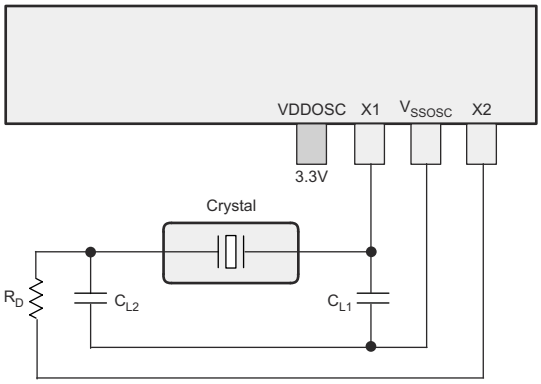


Figure 3-7. External Crystal

- An external resonator. The resonator should be connected across X1 and X2 with its ground connected to VSSOSC as shown in Figure 3-8.

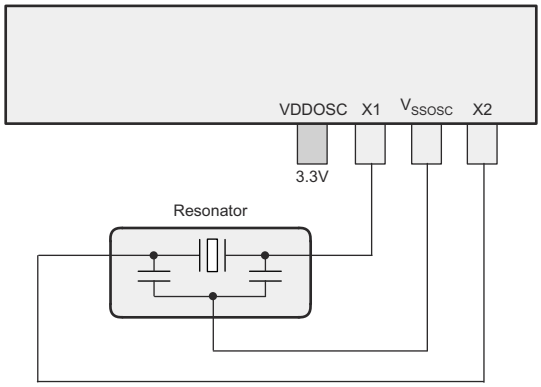


Figure 3-8. External Resonator

3.7.1.4 Auxiliary Clock Input (AUXCLKIN)

An additional external clock source is supported on GPIO133 (AUXCLKIN). This must be a single-ended 3.3V external clock. It can be used as the clock source for the USB, CAN, and Communication Manager Subsystem. Frequency limits and timing requirements can be found in the device datasheet. The external clock should be connected directly to the GPIO133 pin, as shown in Figure 3-9.

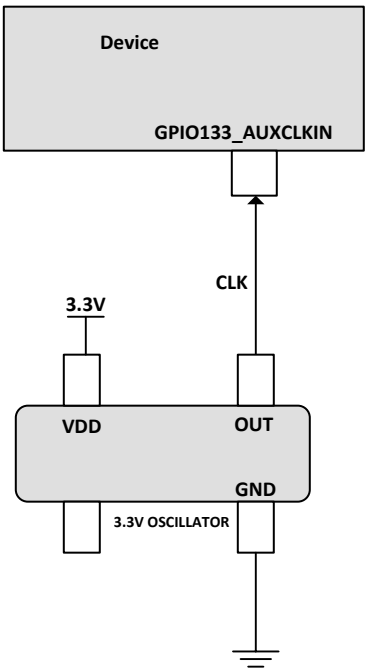


Figure 3-9. AUXCLKIN

3.7.2 Derived Clocks

The clock sources discussed in the previous section can be multiplied (via PLL) and divided down to produce the desired clock frequencies for the application. This process produces a set of derived clocks, which are described in this section.

3.7.2.1 Oscillator Clock (OSCCLK)

One of INTOSC2, XTAL, or INTOSC1 must be chosen to be the master reference clock (OSCCLK) for the CPU and most of the peripherals. OSCCLK may be used directly or fed through the system PLL to reach a higher frequency. At reset, OSCCLK is the default system clock, and is connected to INTOSC2.

3.7.2.2 System PLL Output Clock (PLLRAWCLK)

The system PLL allows the device to run at the maximum rated operating frequency, and in most applications generates the main system clock. This PLL uses OSCCLK as a reference. For configuration instructions, see Section 3.7.6.

3.7.2.3 Auxiliary Oscillator Clock (AUXOSCCLK)

One of INTOSC2, XTAL, or AUXCLKIN may be chosen to be the auxiliary reference clock (AUXOSCCLK) for the USB module. (This selection does not affect the CAN bit clock, which uses AUXCLKIN directly). AUXOSCCLK may be used directly or fed through the auxiliary PLL to reach a higher frequency. At reset, AUXOSCCLK is connected to INTOSC2, but only an external oscillator can meet the USB timing requirements.

3.7.2.4 Auxiliary PLL Output Clock (AUXPLLRAWCLK)

The auxiliary PLL is used to generate a clock for USB, CAN, EtherCAT, and the CM Subsystem. This PLL uses AUXOSCCLK as a reference. For configuration instructions, see Section 3.7.6.

3.7.3 Device Clock Domains

The device clock domains feed the clock inputs of the various modules in the device. They are connected to the derived clocks, either directly or through an additional divider.

3.7.3.1 System Clock (PLLSYSCLK)

The system control registers, GS RAMs, IPC module, GPIO qualification, and NMI watchdog timers have their own clock domain (PLLSYSCLK). Despite the name, PLLSYSCLK may be connected to the system PLL (PLLRAWCLK) or to OSCCLK. The chosen clock source is run through a frequency divider, which is configured via the SYSCLKDIVSEL register.

3.7.3.2 CPU Clock (CPUCLK)

Each CPU has its own clock (CPU1.CPUCLK and CPU2.CPUCLK) which is used to clock the CPU, its coprocessors, its private RAMs (M0, M1, D0, and D1), and its boot ROM and Flash wrapper. This clock is identical to PLLSYSCLK, but is gated when the CPU enters IDLE or STANDBY mode.

3.7.3.3 CPU Subsystem Clock (SYSCLK and PERx.SYSCLK)

Each CPU provides a clock (CPU1.SYSCLK and CPU2.SYSCLK) to its CLA, DMA, and most owned peripherals. This clock is identical to PLLSYSCLK, but is gated when the CPU enters STANDBY mode.

Each peripheral clock can be connected to either CPU1.SYSCLK or CPU2.SYSCLK. This selection is made by CPU1 via the CPUSELx registers. Each peripheral clock also has its own independent clock gating which is controlled by the CPU's PCLKCRx registers. By default, the ePWM, EMIF1, and EMIF2 clocks each have an additional /2 divider, which is required to support CPU frequencies over 100 MHz. At slower CPU frequencies, these dividers can be disabled via the PERCLKDIVSEL register.

A peripheral may be assigned to either CPU. That is, code for a peripheral can be executed from either CPU1 or CPU2. CPUSELx register is used to assign a peripheral to either CPU1 or CPU2. This register must be configured prior to enabling the clock for the chosen peripheral since the clock for each peripheral is derived from the selected CPU subsystem. The clock multiplexer controlled by the CPUSELx register is not glitch-free. Therefore the CPUSELx register must be configured before the PCLKCRx register. Note that the reset for each peripheral is also driven from the selected CPU.

Note

Application needs to wait for 5 SYSCLK cycles after enabling clock to the peripherals when using PCLKCRx.

3.7.3.4 Low-Speed Peripheral Clock (LSPCLK and PERx.LSPCLK)

The SCI, SPI, and McBSP modules can communicate at bit rates that are much slower than the CPU frequency. These modules are connected to a shared clock divider, which generates a low-speed peripheral clock (LSPCLK) derived from SYSCLK. LSPCLK uses a /4 divider by default, but the ratio can be changed via the LOSPCP register. Each SCI, SPI, and McBSP module's clock (PERx.LSPCLK) can be gated independently via the PCLKCRx registers.

3.7.3.5 USB Auxiliary Clock (AUXPLLCLK)

The USB module requires a fixed 60 MHz clock for bit sampling. Since the main system clock is usually not a multiple of 60 MHz, the correct frequency cannot be achieved with a simple divider. Instead, the USB clock is provided through an auxiliary clock path (AUXPLLCLK), which can use an independent clock source and PLL to generate the correct frequency.

USB clock tolerances are very tight. As stated in section 7.1.11 of the USB 2.0 specification, low-speed devices (1.50 Mb/s) have a tolerance of +/- 1.5% , while high-speed devices (12.000 Mb/s) have a tolerance of +/- 0.25%. Typically these tolerances are achieved by using an external crystal or resonator as the source for AUXOSCCLK.

3.7.3.6 CAN Bit Clock

The required frequency tolerance for the CAN bit clock depends on the bit timing setup and network configuration, and can be as tight as 0.1%. Since the main system clock (in the form of PERx.SYSCLK) may not be precise enough, the bit clock can also be connected to XTAL or AUXCLKIN via the CLKSRCCTL2 register. There is an independent selection for each CAN module.

3.7.3.7 CPU Timer2 Clock (TIMER2CLK)

CPU timers 0 and 1 are connected to PERx.SYSCLK. Timer 2 is connected to PERx.SYSCLK by default, but may also be connected to INTOSC1, INTOSC2, XTAL, or AUXPLLCLK via the TMR2CLKCTL register. This register also provides a separate pre-scale divider for timer 2. If a source other than SYSCLK is used, the SYSCLK frequency must be at least twice the source frequency to ensure correct sampling. Each CPU has its own independent CPU timers and TMR2CLKCTL register.

The main reason to use a non-SYSCLK source would be for internal frequency measurement. In most applications, timer 2 will run off of the SYSCLK.

3.7.4 External Clock Output (XCLKOUT)

It is sometimes necessary to observe a clock directly for debug and testing purposes. The external clock output (XCLKOUT) feature supports this by connecting a clock to an external pin, GPIO73. The available clock sources are PLLSYSCLK, PLLRAWCLK, CPU1.SYSCLK, AUXPLLRAWCLK, CPU2.SYSCLK, ,INTOSC1, and INTOSC2.

To use XCLKOUT, first select the clock source via the CLKSRCCTL3 register. Next, select the desired output divider via the XCLKOUTDIVSEL register. Finally, connect GPIO73 to mux channel 3 using the GPIO configuration registers.

3.7.5 Clock Connectivity

Table 3-6 provide details on the clock connections of every module present in the device.

Table 3-6. Clock Connections Sorted by Clock Domain

Clock Domain	CPU1 Subsystem	CPU2 Subsystem	Shared Modules
CPUx.CPUCLK	CPU1	CPU2	
	CPU1VCRC	CPU2.VCRC	
	CPU1.FPU	CPU2.FPU	
	CPU1.TMU	CPU2.TMU	
	CPU1.Flash	CPU2.Flash	
	CPU1.DCSM	CPU2.DCSM	
	CPU1.HWBIST	CPU2.HWBIST	

Table 3-6. Clock Connections Sorted by Clock Domain (continued)

Clock Domain	CPU1 Subsystem	CPU2 Subsystem	Shared Modules
CPUx.SYSCLK	CPU1.ePIE CPU1.LS0 - LS7 RAMs CPU1.M0 - M1 RAMs CPU1.D0 - D1 RAMs CPU1.BootROM CPU1.CLA1 Message RAMs CPU1.Timer0-2 CPU1.DMA CPU1.XINT CPU1.CLA1 CPU1.BGCRC CPU1.ERAD CPU1.CM Message RAMs EMIF2	CPU2.ePIE CPU2.LS0 - LS7 RAMs CPU2.M0 - M1 RAMs CPU2.D0 - D1 RAMs CPU2.BootROM CPU2.CLA1 Message RAMs CPU2.Timer0-2 CPU2.DMA CPU2.XINT CPU2.CLA1 CPU2.BGCRC CPU2.ERAD CPU2.CM Message RAMs	
PLLSYSCLK	CPU1.NMIWD	CPU2.NMIWD	GS0 - GS15 RAMs GPIO Input Sync and Qual IPC CPU1 & CPU2 MSG RAMs XBARS EMIF1 AnalogSubsys EPWM System Control Registers
PERx.SYSCLK	USB		ADC CMPSS DAC ePWM & HRPWM eCAP eQEP I2C McBSP SDFM FSI PMBUS HRCAL SPI SCI DCC CAN
PERx.LSPCLK			McBSP SCI SPI
CAN Bit Clock			CAN
AUXPLLCLK	USB		
WDCLK (INTOSC1)	CPU1.Watchdog	CPU2.Watchdog	

3.7.6 PLL/AUXPLL

The PLL/AUXPLL is responsible for synthesizing an output frequency from the input clock (from the oscillator); Figure 3-10 shows a simple block diagram of the PLL/AUXPLL. The PLL/AUXPLL divides the reference input for a lower frequency input into the PLL/AUXPLL by (REFDIV+1). Then multiplies this internal frequency by IMULT to get the VCO output clock. The PLL/AUXPLL output is divided by (ODIV+1) to generate PLLRAWCLK/ AUXPLLRAWCLK which is further divided by SYSCLKDIVSEL.PLLSYSCLKDIV/AUXCLKDIVSEL.AUXPLLDIV to generate PLLSYSCLK/AUXCLK

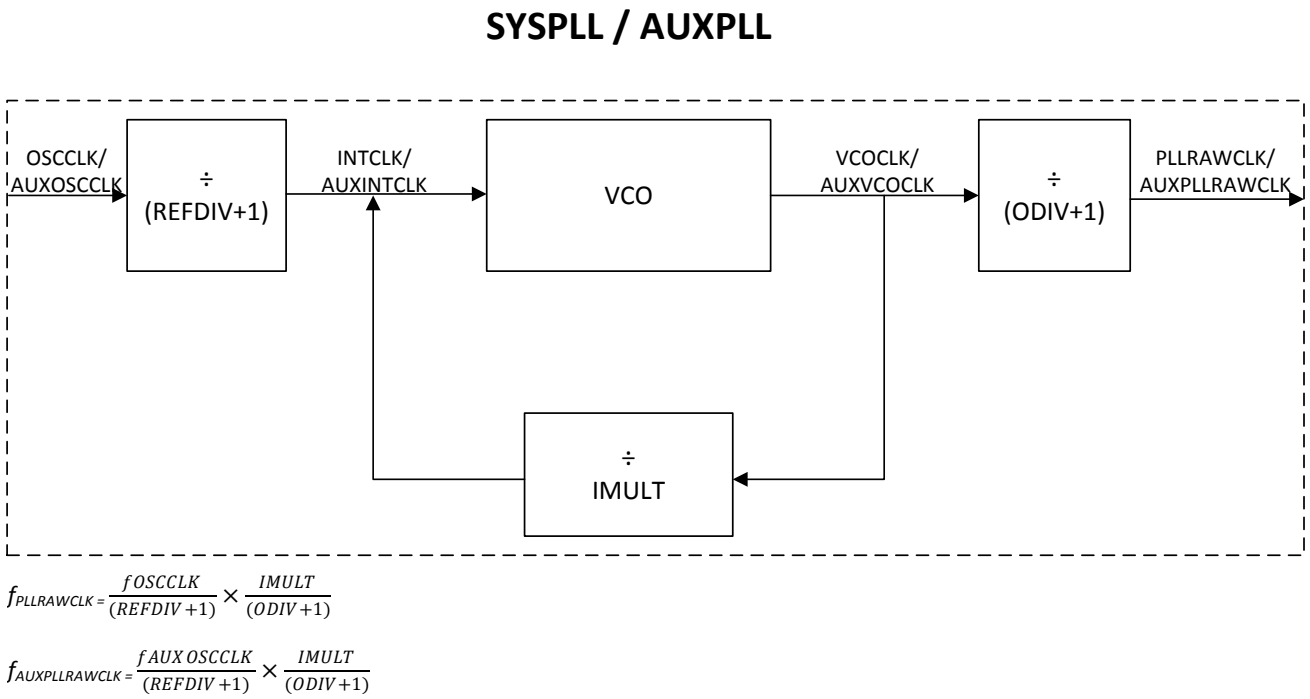


Figure 3-10. PLL/AUXPLL

3.7.6.1 Choosing PLL Settings

There are two PLLs (SYSPLL and AUXPLL) and equations shown in Figure 3-10 should be used to configure respective PLL

IMULT is the integer value of the multiplier.

REFDIV is the reference divider for the OSCCLK/AUXOSCCLK.

ODIV is the output divider of the PLLRAWCLK/AUXPLLRAWCLK.

PLLSYSCLKDIV is the system clock divider.

AUXPLLDIV is the auxiliary clock divider.

For the permissible values of the multipliers and dividers, see the documentation for their respective registers.

Many combinations of multiplier and divider can produce the same output frequency. However, the product of the reference clock frequency and the multiplier (known as the VCO frequency) must be in the range specified in the data manual.

Note

The system clock frequency (PLLSYSCLK) may not exceed the limit specified in the datasheet. This limit does not allow for oscillator tolerance.

The clock source and PLL configuration registers are shared between the two CPUs (CPU1 and CPU2). Register access is controlled by way of a semaphore, which is described in the *Interprocessor Communication (IPC)* chapter.

3.7.6.2 System Clock Setup

Once the application requirements are understood, a specific clock configuration can be determined. The default configuration is for INTOSC2 to be used as the system clock (PLLSYSCLK) with a divider of 1. The following procedure should be used to set up the desired application configuration:

Refer to your device SysCtl_setClock() function inside C2000Ware installation for an example.

Recommended sequence to set up the system PLL:

1. Bypass the PLL by clearing SYSPLLCTL1[PLLCLKEN] and wait for at least 120 CPU clock cycles by adding 120 NOP instructions.
2. Power down the PLL by writing to SYSPLLCTL1.PLLEN = 0 and wait for at least 60 CPU clock cycles by adding 60 NOP instructions.
3. Select the reference clock source (OSCCLK) by writing to CLKSRCCTL1.OSCCLKSRCSEL and wait for at least 300 CPU clock cycles by adding 300 NOP instructions.
4. Set the system clock divider to "/1" to ensure the fastest PLL configuration by clearing SYSCLKDIVSEL[PLLSYSCLKDIV].
5. Set the IMULT, REFDIV & ODIV simultaneously by writing 32-bit value in SYSPLLMULT at once. This will automatically enable the PLL. Be sure the settings for multiplier and dividers do not violate the frequency specifications as defined in the data sheet.
6. Wait for PLL to lock by polling for lock status bit to go high, that is, SYSPLLSTS.LOCKS = 1
7. Configure DCC with reference clock as OSCCLK and clock under measurement as PLLRAWCLK, and verify the frequency of the PLL. If the frequency is out of range, do not enable PLLRAWCLK as SYSCLK, stop here and troubleshoot. Refer to DCC chapter for more information on its configuration and usage.
8. If the PLLRAWCLK is within the valid range, then set the system clock divider one setting higher than the final desired value. For example ClkCfgRegs.SYSCLKDIVSEL.bit.PLLSYSCLKDIV = divsel + 1. This limits the current increase when switching to the PLL.
9. Switch to the PLL as the system clock by setting SYSPLLCTL1[PLLCLKEN] and wait for 200 PLLSYSCLK cycles for current to stabilize by adding 200 NOP instructions.
10. Change the system clock divider (PLLSYSCLKDIV) to the appropriate value.

Note

1. SYSPLL must be bypassed and powered down manually before changing the OSCCLK source.
2. At least 120 CPU clock cycles delay is needed after bypassing PLL, that is, SYSPLLCTL1.PLLCLKEN=0.
3. At least 60 CPU clock cycles delay is needed after PLL is powered down, that is, SYSPLLCTL1.PLLEN=0.
4. At least 300 CPU clock cycles delay is needed after OSSCLK source is changed.
5. PLL SLIP bit is not supported. DCC should be used to check the validity of the PLL clock. This feature is included as part of SysCtl_setClock() function inside C2000Ware.

3.7.6.3 USB Auxiliary Clock Setup

Refer to your device SysCtl_setAuxClock() function inside C2000Ware installation for an example.

If USB functionality is needed, the auxiliary clock (AUXPLLCLK) must be configured to produce 60 MHz. The procedure is similar to the system clock setup:

1. Bypass the PLL by clearing AUXPLLCTL1[PLLCLKEN] and wait for at least 120 CPU clock cycles by adding 120 NOP instructions..
2. Power down the PLL by writing to AUXPLLCTL1.PLLEN=0 and wait for at least 60 CPU clock cycles by adding 60 NOP instructions.
3. Select the reference clock source (AUXOSCCLK) by writing to CLKSRCCTL2.AUXOSCCLKSRCSEL and wait for at least 60 CPU clock cycles by adding 60 NOP instructions.
4. Set the IMULT, REFDIV & ODIV simultaneously by writing 32-bit value in AUXPLLMULT at once. This will automatically enable the PLL. Be sure the settings for multiplier and dividers do not violate the frequency specifications as defined in the data sheet.
5. Wait for PLL to lock by polling for lock status bit to go high (AUXPLLSTS.LOCKS = 1)
6. Configure DCC with reference clock as AUXOSCCLK and clock under measurement as AUXPLLRAWCLK, and verify the frequency of the PLL. If the frequency is out of range, stop here and troubleshoot. Refer to DCC chapter for more information on its configuration and usage.
7. Connect the auxiliary PLL output clock (AUXPLLRAWCLK) to AUXPLLCLK by writing a 1 to AUXPLLCTL1.PLLCLKEN.

The auxiliary clock configuration can be changed at run time. Changing the AUXOSCCLK source will automatically bypass the PLL and set the multiplier to zero. Changing the multiplier from one non-zero value to another will temporarily bypass the PLL until it re-locks.

Note

1. AUXPLL must be bypassed and powered down manually before changing the AUXOSCCLK source.
2. At least 120 CPU clock cycles delay is needed after bypassing PLL, that is, AUXPLLCTL1.PLLCLKEN = 0.
3. At least 60 CPU clock cycles delay is needed after PLL is powered down, that is, AUXPLLCTL1.PLLEN = 0.
4. At least 60 CPU clock cycles delay is needed after AUXOSSCLK source is changed.
5. AUXPLL SLIP bit is not supported. DCC should be used to check the validity of the AUXPLL clock. This feature is included as part of SysCtl_setAuxClock() function inside C2000Ware.

3.7.6.4 SYS PLL / AUX PLL Bypass

If the application requires the PLL clock to be bypassed from the system, then the application needs to configure SYSPLLCTL1.PLLCLKEN=0 or AUXPLLCTL1.PLLCLKEN=0. It takes up to 120 CPU clock cycles before the bypass is effective. In the meantime, if PLLSYSCLKDIV / AUXPLLDIV is reduced to a lower value (for example, from /2 to /1 or /4 to /2), the device may be clocked above the maximum rated frequency and can lead to unpredictable device behavior. Hence, a delay of 120 CPU clock cycles is required after bypassing the PLL from the enable state, that is, going from PLLCLKEN=1 to PLLCLKEN=0.

3.7.7 Clock (OSCCLK) Failure Detection

To achieve safety diagnostic, Missing Clock Detection (MCD) can be used.

Table 3-7 lists the details.

Table 3-7. Clock Source (OSCCLK) Failure Detection

Clock Failure Detection Circuitry	Clocks Detected	Time for Detection (in Cycles)	Limitations
Missing Clock Detection (MCD)	INTOSC2, XTAL/X1	8192 INTOSC1 cycles	Cannot detect INTOSC1 clock failure.

3.7.7.1 Missing Clock Detection Logic

The missing clock detect (MCD) logic detects OSCCLK failure, using INTOSC1 as the reference clock source. This circuit only detects complete loss of OSCCLK and doesn't do any detection of frequency drift on the OSCCLK.

This circuit monitors the OSCLK (primary clock) using the 10 MHz clock provided by the INTOSC1 (secondary clock) as a backup clock. This circuit functions as:

1. The primary clock (OSCCLK) clock keeps ticking a 7-bit counter (named as MCDPCNT). This counter is asynchronously reset with XRSn.
2. The secondary clock (INTOSC1) clock keeps ticking a 13-bit counter (named as MCDSCNT). This counter is asynchronously reset with XRSn.
3. Each time MCDPCNT overflows, the MCDSCNT counter is reset. Thus, if OSCCLK is present or not slower than INTOSC1 by a factor of 64, MCDSCNT never overflows.
4. If OSCCLK stops for some reason or is slower than INTOSC1 by at least a factor of 64, the MCDSCNT overflows and a missing clock condition is detected on OSCCLK.
5. The above check is continuously active, unless the MCD is disabled using MCDCR register (by making the MCLKOFF bit 1).
6. If the circuit ever detects a missing OSCCLK, the following occurs:
   - The MCDSTS flag is set.
   - The MCDSCNT counter is frozen to prevent further missing clock detection.
   - The CLOCKFAIL signal goes high, which generates TRIP events to PWM modules and fires NMIs to CPU1.NMIWDand CPU2.NMIWD.
   - PLL is forcefully bypassed and OSCCLK source is switched to INTOSC1 (System Clock Frequency = INTOSC1 Freq (10MHz)/SYSDIV). PLLMULT is zeroed out automatically in this case.
   - While the MCDSTS bit is set, the OSCCLKSRCSEL bits have no effect and OSCCLK is forcefully connected to INTOSC1.
   - PLLRAWCLK going to the system is switched to INTOSC1 automatically.
7. If the MCLKCLR bit is written (this is a W=1 bit), MCDSTS bit is cleared and OSCCLK source is decided by the OSCCLKSRCSEL bits. Writing to MCLKCLR also clears the MCDPCNT and MCDSCNT counters to allow the circuit re-evaluate missing clock detection. If the user wants to lock the PLL after missing clock detection, switch the clock source to INTOSC1 (using OSCCLKSRCSEL register), do a MCLKCLR, and re-lock the PLL.
8. The MCD is enabled at power up.

Figure 3-11 shows the missing clock logic functional flow.

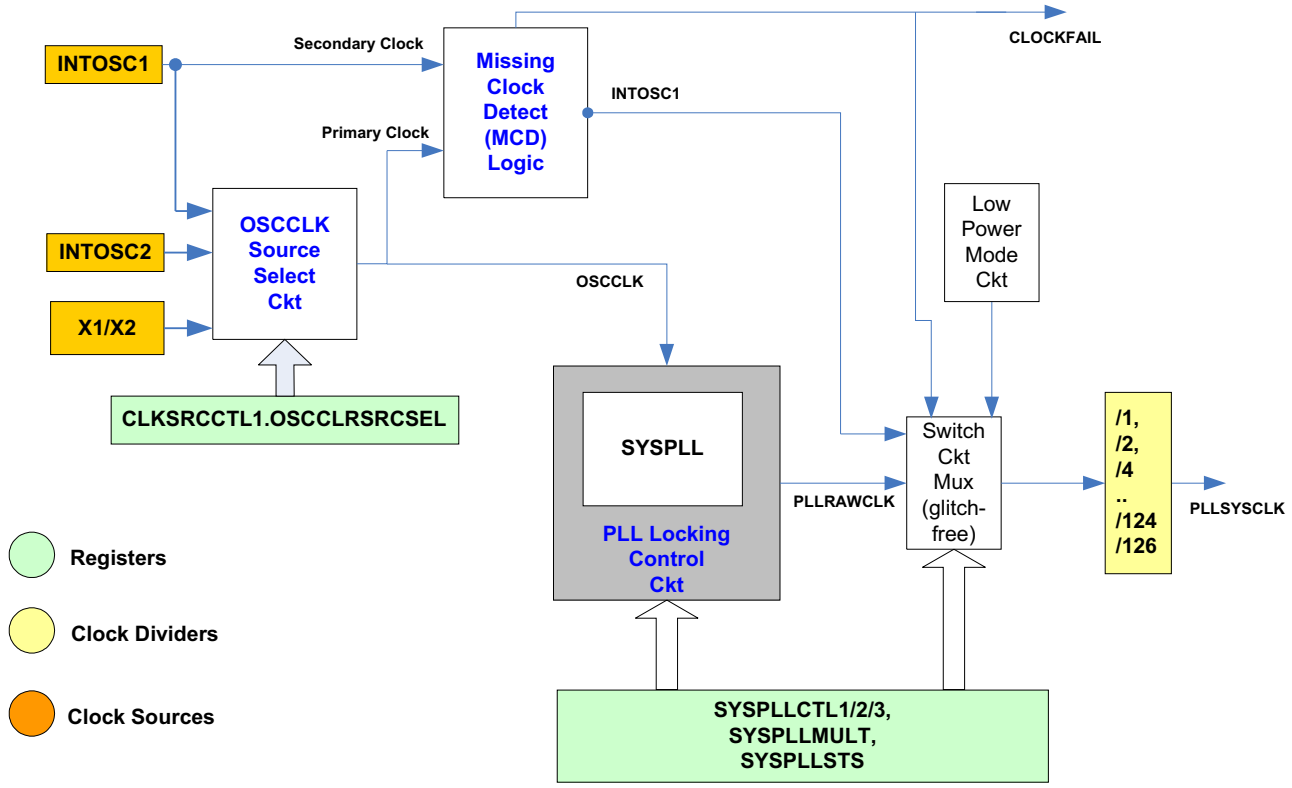


Figure 3-11. Missing Clock Detection Logic

Note

On a complete clock failure when OSCCLK is dead, it may take a maximum time of 8192 INTOSC1 cycles (that is, 0.8192 ms) before CLOCKFAIL signal goes high, after which:

- NMI is generated
- OSCCLK is switched to INTOSC1
- PWM Trip happens

3.8 Clock Configuration Semaphore

Both CPUs can access the PLL and peripheral clock configuration registers. The clock configuration semaphore allows one CPU to access the registers without being interrupted by the other CPU.

The clock configuration semaphore is implemented as a two-bit field in a register with special write protections. This register requires a key field to be written at the same time as the semaphore bits. The possible semaphore states are:

00 or 11	Either CPU writes to the semaphore. CPU1 has control of the clock configuration registers by default. 00 is the reset state.
01	CPU2 has exclusive control of the clock configuration registers and exclusive write access to the semaphore.
10	CPU1 has exclusive control of the clock configuration registers and exclusive write access to the semaphore.

Each CPU is only allowed to take control of the clock configuration registers for itself. However, CPU1 can force both semaphores into the default state (00) at any time by putting CPU2 into reset. Figure 3-12 shows the allowed states and state transitions.

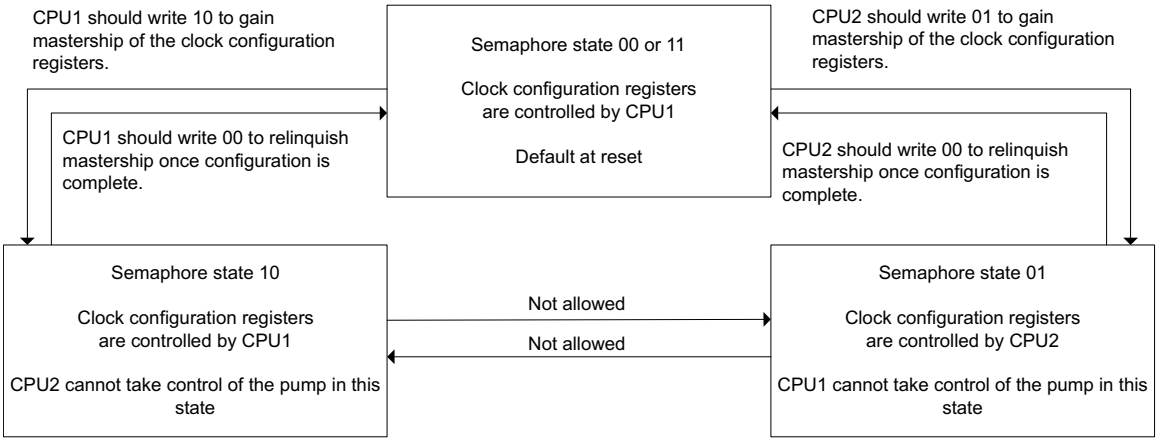


Figure 3-12. Clock Configuration Semaphore (CLKSEM) State Transitions

3.9 32-Bit CPU Timers 0/1/2

This section describes the three 32-bit CPU-Timers (TIMER0/1/2) shown in Figure 3-13.

CPU-Timer2 is reserved for real-time operating system uses (for example, TI-RTOS), but if CPU-Timer2 is not used by real-time operating systems then, CPU-Timer2 can also be used for other applications. The CPU-Timer0 and CPU-Timer1 run off of SYSCLK. The CPU-Timer2 normally runs off of SYSCLK, but can also use INTOSC1, INTOSC2, XTAL, and AUXPLLCLK. The CPU-Timer interrupt signals (TINT0, TINT1, TINT2) are connected as shown in Figure 3-14.

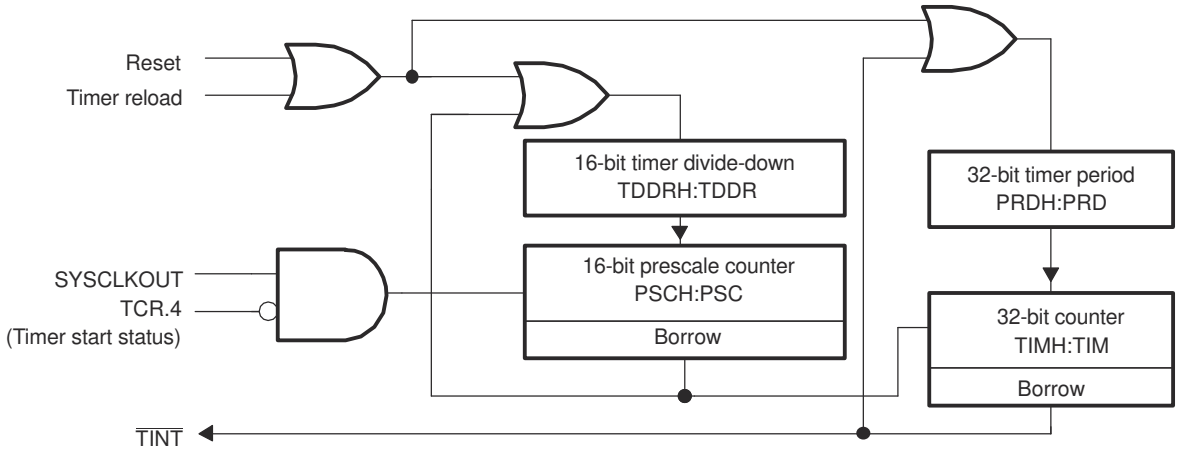
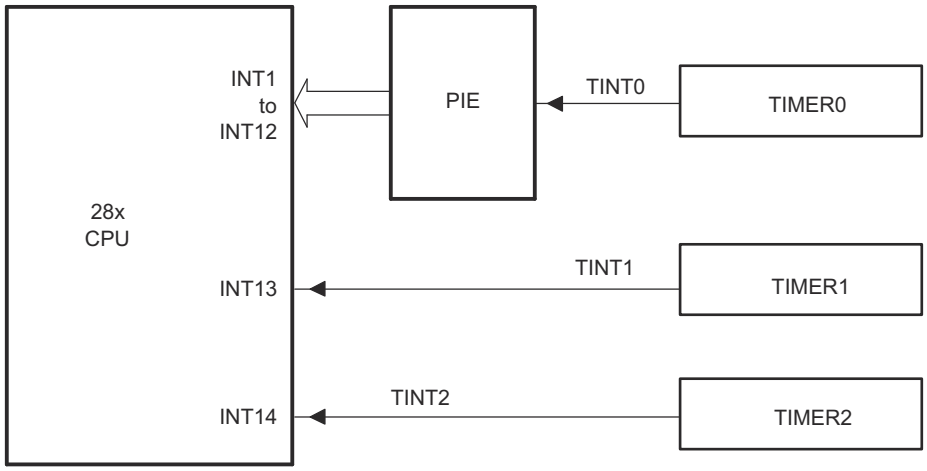


Figure 3-13. CPU-Timers



A. The timer registers are connected to the memory bus of the C28x processor.
B. The CPU Timers are synchronized to SYSCLKOUT.

Figure 3-14. CPU-Timer Interrupts Signals and Output Signal

The general operation of the CPU-Timer is as follows:

- The 32-bit counter register, TIMH:TIM, is loaded with the value in the period register PRDH:PRD.
- The counter decrements once every (TPR[TDDRH:TDDR]+1) SYSCLKOUT cycles, where TDDRH:TDDR is the timer divider.
- When the counter reaches 0, a timer interrupt output signal generates an interrupt pulse.

The registers listed in Section 3.16 are used to configure the timers.

3.10 Watchdog Timers

The watchdog module generates an output pulse 512 watchdog clocks (WDCLKs) wide whenever the 8-bit watchdog up counter has reached its maximum value. The watchdog clock source is INTOSC1. Software must periodically write a 0x55 + 0xAA sequence into the watchdog key register to reset the watchdog counter. The counter can also be disabled. Figure 3-15 shows the various functional blocks within the watchdog module.

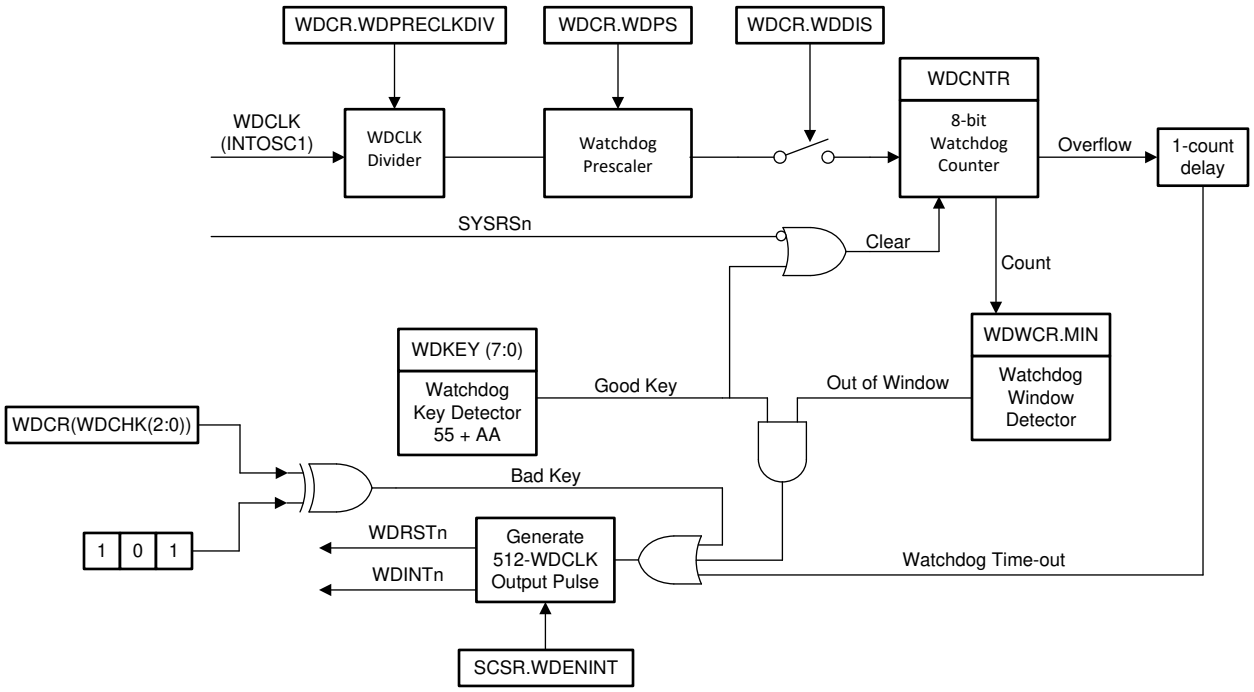


Figure 3-15. CPU Watchdog Timer Module

3.10.1 Servicing the Watchdog Timer

The watchdog counter (WDCNTR) is reset when the proper sequence is written to the WDKEY register before the 8-bit watchdog counter overflows. The WDCNTR is reset-enabled when a value of 0x55 is written to the WDKEY. When the next value written to the WDKEY register is 0xAA, then the WDCNTR is reset. Any value written to the WDKEY other than 0x55 or 0xAA causes no action. Any sequence of 0x55 and 0xAA values can be written to the WDKEY without causing a system reset; only a write of 0x55 followed by a write of 0xAA to the WDKEY resets the WDCNTR.

Table 3-8. Example Watchdog Key Sequences

Step	Value Written to WDKEY	Result
1	0xAA	No action
2	0xAA	No action
3	0x55	WDCNTR is enabled to be reset if next value is 0xAA.
4	0x55	WDCNTR is enabled to be reset if next value is 0xAA.
5	0x55	WDCNTR is enabled to be reset if next value is 0xAA.
6	0xAA	WDCNTR is reset.
7	0xAA	No action
8	0x55	WDCNTR is enabled to be reset if next value is 0xAA.
9	0xAA	WDCNTR is reset.
10	0x55	WDCNTR is enabled to be reset if next value is 0xAA.
11	0x32	Improper value written to WDKEY. No action, WDCNTR no longer enabled to be reset by next 0xAA.
12	0xAA	No action due to previous invalid value.
13	0x55	WDCNTR is enabled to be reset if next value is 0xAA.
14	0xAA	WDCNTR is reset.

Step 3 in Table 3-8 is the first action that enables the WDCNTR to be reset. The WDCNTR is not actually reset until step 6. Step 8 again re-enables the WDCNTR to be reset and step 9 resets the WDCNTR. Step 10 again re-enables the WDCNTR to be reset. Writing the wrong key value to the WDKEY in step 11 causes no action, however the WDCNTR is no longer enabled to be reset and the 0xAA in step 12 now has no effect.

If the watchdog is configured to reset the device, then a WDCNTR overflow or writing the incorrect value to the WDCR[WDCHK] bits resets the device and sets the watchdog flag (WDRSn) in the reset cause register (RESC). After a reset, the program can read the state of this flag to determine whether the reset was caused by the watchdog. After doing this, the program must clear WDRSn to allow subsequent watchdog resets to be detected. Watchdog resets are not prevented when the flag is set.

3.10.2 Minimum Window Check

To complement the timeout mechanism, the watchdog also contains an optional "windowing" feature that requires a minimum delay between counter resets. This can help protect against error conditions that bypass large parts of the normal program flow but still include watchdog handling.

To set the window minimum, write the desired minimum watchdog count to the WDWCR register. This value will take effect after the next WDKEY sequence. From then on, any attempt to service the watchdog when WDCNTR is less than WDWCR will trigger a watchdog interrupt or reset. When WDCNTR is greater than or equal to WDWCR, the watchdog can be serviced normally.

At reset, the window minimum is zero, which disables the windowing feature.

3.10.3 Watchdog Reset or Watchdog Interrupt Mode

The watchdog can be configured in the SCSR register to either reset the device ($\overline{\text{WDRST}}$) or assert an interrupt ($\overline{\text{WDINT}}$), if the watchdog counter reaches the maximum value. The behavior of each condition is:

- **Reset mode:**

  If the watchdog is configured to reset the device, then the $\overline{\text{WDRST}}$ signal pulls the device reset ($\overline{\text{XRS}}$) pin low for 512 OSCCLK cycles when the watchdog counter reaches the maximum value.
- **Interrupt mode:**

  When the watchdog counter expires, the watchdog counter asserts an interrupt by driving the $\overline{\text{WDINT}}$ signal low for 512 OSCCLK cycles. The falling edge of $\overline{\text{WDINT}}$ triggers a WAKEINT interrupt in the PIE, if the interrupt is enabled. Because the PIE is edge-triggered, re-enabling the WAKEINT while $\overline{\text{WDINT}}$ is active does not produce a duplicate interrupt.

  To avoid unexpected behavior, software must not change the configuration of the watchdog while $\overline{\text{WDINT}}$ is active. For example, changing from interrupt mode to reset mode while $\overline{\text{WDINT}}$ is active immediately resets the device. Disabling the watchdog while $\overline{\text{WDINT}}$ is active causes a duplicate interrupt if the watchdog is later re-enabled. If a debug reset is issued while $\overline{\text{WDINT}}$ is active, the reset cause register (RESC) shows a watchdog reset. The WDINTS bit in the SCSR register can be read to determine the current state of $\overline{\text{WDINT}}$.

3.10.4 Watchdog Operation in Low Power Modes

In IDLE mode, the watchdog interrupt ($\overline{\text{WDINT}}$) signal can generate an interrupt to the CPU to take the CPU out of IDLE mode. As with any other peripheral, the watchdog interrupt will trigger a WAKEINT interrupt in the PIE during IDLE mode. User software must determine which peripheral caused the interrupt.

In STANDBY mode, all of the clocks to the peripherals are turned off within the CPU subsystem. The only peripheral that remains functional is the watchdog since the watchdog module runs off the oscillator clock (OSCCLK). The $\overline{\text{WDINT}}$ signal is fed to the Low Power Modes (LPM) block so that it can be used to wake the CPU from STANDBY low power mode. This feature is enabled by setting LPMCR.WDINTE = 1. See Section 3.11 for details.

Note: If the watchdog interrupt is used to wake-up from an IDLE or STANDBY low power mode condition, software must make sure that the $\overline{\text{WDINT}}$ signal goes back high before attempting to reenter the IDLE or STANDBY mode. The $\overline{\text{WDINT}}$ signal will be held low for 512 OSCCLK cycles when the watchdog interrupt is generated. The current state of $\overline{\text{WDINT}}$ can be determined by reading the watchdog interrupt status bit (WDINTS) bit in the SCSR register. WDINTS follows the state of $\overline{\text{WDINT}}$ by two SYSCLKOUT cycles.

3.10.5 Emulation Considerations

The watchdog module behaves as follows under various debug conditions:

CPU Suspended:	When the CPU is suspended, the watchdog clock (WDCLK) is suspended
Run-Free Mode:	When the CPU is placed in run-free mode, then the watchdog module resumes operation as normal.
Real-Time Single-Step Mode:	When the CPU is in real-time single-step mode, the watchdog clock (WDCLK) is suspended. The watchdog remains suspended even within real-time interrupts.
Real-Time Run-Free Mode:	When the CPU is in real-time run-free mode, the watchdog operates as normal.

3.11 Low Power Modes

This device has two clock-gating, low-power modes. All low-power modes are entered by setting the LPMCR register and executing the IDLE instruction. More information about this instruction can be found in the *TMS320C28x CPU and Instruction Set Reference Guide*.

Low-power modes should not be entered into while a Flash program or erase is ongoing.

The application should verify the following before entering STANDBY:

1. Check the value of the GPIODAT register of the pin selected for STANDBY(GPIOLPMSEL0/1) prior to entering the Low-Power mode to ensure that the wake event has not already been asserted.
2. The LPMCR.QUALSTDBY register should be set to a value greater than the ratio of INTOSC1/PLLSYSCLK to ensure proper wake up.

3.11.1 IDLE

IDLE is a standard feature of the C28x CPU. In this mode, the CPU clock is gated while all peripheral clocks are left running. IDLE can thus be used to conserve power while a CPU is waiting for peripheral events. When one CPU is in IDLE, there is no effect on the other CPU subsystem.

Any enabled interrupt wakes up the CPU from IDLE mode.

To enter IDLE mode, set LPMCR.LPM to 0x0 and execute the IDLE instruction.

3.11.2 STANDBY

STANDBY is a more aggressive low-power mode that gates both the CPU clock and any peripheral clocks derived from the CPU SYSCLK. The watchdog, however, is left active. Like IDLE, this mode affects only one CPU subsystem. The other CPU subsystem and all of the peripherals are unaffected. STANDBY is best used for an application where the wake-up signal comes from an external system (or CPU subsystem) rather than a peripheral input.

An NMI (or optionally) a watchdog interrupt or a configured GPIO can wake the CPU from STANDBY mode. Each GPIO from GPIO0-63 can be configured to wake the CPU when the GPIO are driven active low. Upon wakeup, the CPU receives the WAKEINT interrupt if configured.

IPC interrupt 1 (flag 0), an NMI fired to the other CPU, or (optionally) a watchdog interrupt, wakes up the CPU subsystem from STANDBY mode. Any of GPIO0-63 can also be configured to wake up the subsystem when the GPIO are driven active low. Upon wake up, the CPU receives a WAKEINT interrupt, even if the CPU was woken by an IPCINT1 signal.

To enter STANDBY mode:

1. Set LPMCR.LPM to 0x1.
2. Enable the WAKEINT interrupt in the PIE.
3. For watchdog interrupt wakeup, set LPMCR.WDINTE to 1 and configure the watchdog to generate interrupts.
4. For GPIO wakeup, set GPIOLPMSEL0 and GPIOLPMSEL1 to connect the chosen GPIOs to the LPM module, and set LPMCR.QUALSTDBY to select the number of OSCCLK cycles for input qualification.
5. Execute the IDLE instruction to enter STANDBY.

To wake up from Standby mode:

1. Configure the desired GPIO to trigger the wakeup.
2. Drive the selected GPIO signal low; the signal must remain low for the number of OSCCLK cycles specified in the QUALSTDBY bits in the LPMCR register. If the signal is sampled high during this period, the count restarts.

At the end of the qualification period, the PLL enables the CLKIN to the CPU and the WAKEINT interrupt is latched in the PIE block. The WAKEINT interrupt can also triggered by IPCINT1 sent from the other CPU and a watchdog interrupt.

The CPU is now out of STANDBY mode and can resume normal execution.

If CPU2 is in STANDBY mode, writing a 1 to the RESET bit of the CPU2RESCTL register has no effect. CPU2 can be reset by any chip-level reset (POR, XRSn, CPU1.WDRSn, or CPU1.NMIWDRSn). Alternately, CPU2 can be woken up by any configured wake-up event.

If CPU2 is in STANDBY mode and the debugger is connected, executing a debug reset on CPU2 has no effect. To wake up the CPU2 with the debugger, Click Run, Single Step, or Step over in the Debug toolbar. CCS IDE prompts the user requesting to bring the CPU out of the low-power mode. Click Yes. This wakes up CPU2 from STANDBY and continues execution.

3.12 Memory Controller Module

This device has CPU1 subsystem, CPU2 subsystem, and CM subsystem. This section describes the memory controller used for CPU1 and CPU2 subsystem.

The different RAMs available on CPU1 and CPU2 subsystem have different characteristics. Some are:

- Dedicated to each CPU (M0, M1, and Dx RAMs),
- Shared between the CPU and a CLA (LSx RAM),
- Shared between the CPU and DMA of both subsystems (GSx RAM), and
- Used to send and receive messages between processors (MSGRAM).
- Used to exchange data between CLA and DMA

All these RAMs are highly configurable to achieve control for write access and fetch access from different masters. There are also RAMs - called IPC MSGRAMs - that are used for interprocessor communication. All RAMs are enabled with the ECC or parity feature (both data and address). Some of the dedicated memories are secure memory as well. Refer to Chapter 6 for more details. Each RAM has a controller that takes care of the access protection/security related checks and ECC/Parity features for that RAM. Figure 3-16 shows the configuration of these RAMs.

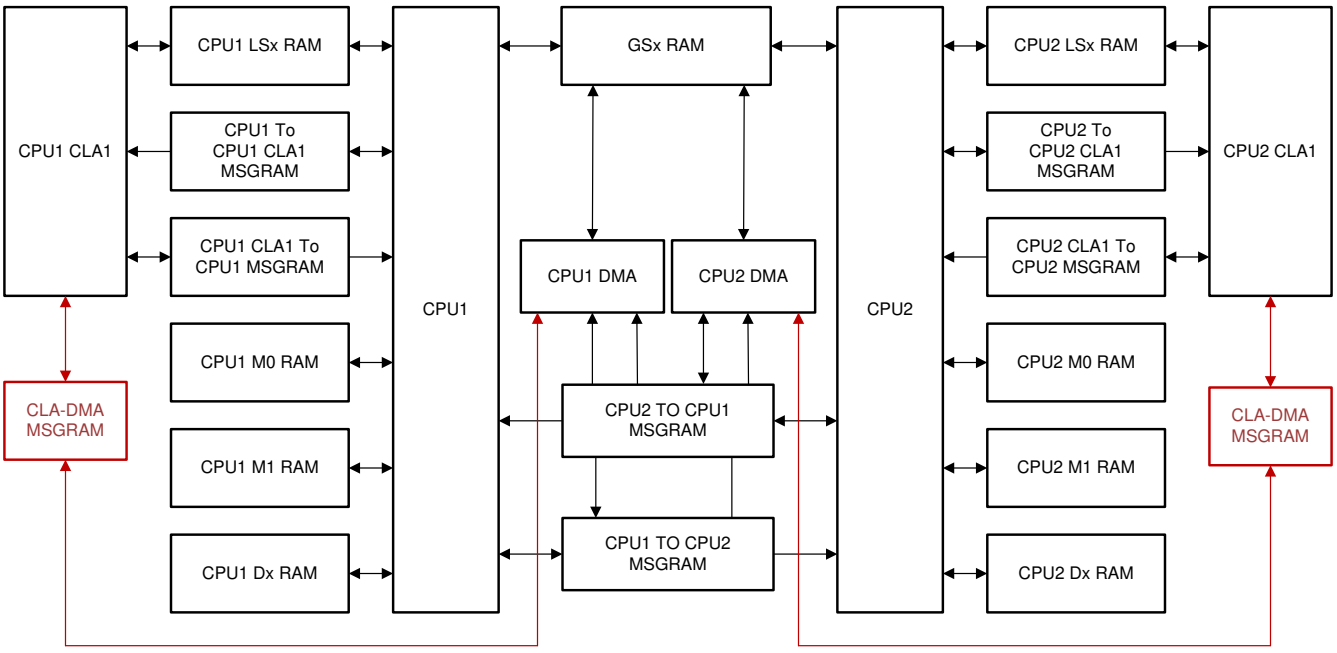


Figure 3-16. Memory Architecture

Note

All RAMs on these devices are SRAMs.

3.12.1 Functional Description

This section further defines and discusses the dedicated RAMs, shared RAMs, and MSG RAMs on this device.

3.12.1.1 Dedicated RAM (Dx RAM)

Each CPU subsystem has four dedicated RAM blocks: M0, M1, D0, and D1. M0/M1 memories are small blocks of memory that are tightly coupled with the CPU. Only the CPU has access to these memories. No other masters (including DMA) have any access to these memories.

All dedicated RAMs have the ECC feature. All dedicated memories are secure memory (except for M0/M1) and have the access protection (CPU write protection/CPU fetch protection) feature. Each type of access protection for each RAM block can be enabled/disabled by configuring the specific bit in the access protection register, allocated to each subsystem (DxACCPROT).

3.12.1.2 Local Shared RAM (LSx RAM)

RAM blocks that are dedicated to each subsystem and are accessible to the respective CPU and CLA only, are called local shared RAMs (LSx RAMs). All such memories are secure memory and have the ECC feature. By default, these memories are dedicated to the CPU only, and the user can choose to share these memories with the CLA by appropriately configuring the MSEL_LSx bit field in the LSxMSEL register. Further, when these memories are shared between the CPU and CLA, the user can choose to use these memories as CLA program memory by configuring the CLAPGM_LSx bit field in the LSxCLAPGM registers. CPU access to all memory blocks, which are programmed as CLA program memory, are blocked.

All these RAMs have the access protection (CPU write and CPU fetch) feature. Each type of access protection for each RAM block can be enabled or disabled by configuring the specific bit in the local shared RAM access protection registers, mapped to each CPU subsystem. Table 3-9 shows the LSx RAM features.

Table 3-9. Local Shared RAM

MSEL_LSx	CLAPGM_LSx	CPUx Allowed Access	CPUx.CLA1 Allowed Access	Comment
00	X	All	-	LSx memory is configured as CPU dedicated RAM
01	0	All	Data Read Data Write	LSx memory is shared between CPU and CLA1
01	1	Emulation Read Emulation Write	Fetch Only Emulation Read Emulation Write	LSx memory is CLA1 program memory

3.12.1.3 Global Shared RAM (GSx RAM)

RAM blocks that are accessible from both the CPU and their respective DMA are called global shared RAMs (GSx RAMs). Each shared RAM can be owned by either CPU subsystem based on the configuration of their respective bits (one bit for each GSx memory) in the GSxMSEL register. When a particular GSx RAM block is owned by the CPU1 subsystem, CPU1 and CPU1.DMA have full access to that RAM block, whereas CPU2 and CPU2.DMA have only read access to that RAM block (no fetch/write access). Similarly, when a particular GSx RAM block is owned by the CPU2 subsystem, CPU1 and CPU1.DMA has only read access (no fetch/write access) to that RAM block, whereas CPU2 and CPU2.DMA has full access to that RAM block. Table 3-10 shows the features of the GSx RAM.

Table 3-10. Global Shared RAM

GSxMSEL	CPU1	CPU1	CPU1	CPU1.DMA	CPU1.DMA	CPU2	CPU2	CPU2	CPU2.DMA	CPU2.DMA
	Fetch	Read	Write	Read	Write	Fetch	Read	Write	Read	Write
0	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes	No
1	No	Yes	No	Yes	No	Yes	Yes	Yes	Yes	Yes

Note

Emulation/Debugger access is allowed from both the CPUs, irrespective of the GSxMSEL setting.

Like other shared RAMs, these RAMs also have different levels of access protection that can be enabled or disabled by configuring specific bits in the GSxACCPROT registers mapped in each subsystem.

Master select and access protection configuration for each GSx RAM block can be individually locked by the user to prevent further update to these bit fields. The user can also choose to permanently lock the configuration to individual bit fields by setting the specific bit fields in the GSxCOMMIT register (refer to the register description for more details). Once configuration is committed for a particular GSx RAM block, it can not be changed further until CPUx.$\overline{\text{SYSRS}}$ is issued. Only the CPU1 software can change the master select configuration by writing into the GSxMSEL register, mapped on the CPU1. The GSxMSEL register, which is mapped to the CPU2 subsystem, is a status register that can only be used by CPU2 software to know the master ownership for each GSx RAM block.

3.12.1.4 CPU Message RAM (CPU MSG RAM)

These RAM blocks can be used to share data between CPU1 and CPU2. Since these RAMs are used for interprocessor communication, they are also called IPC RAMs. The CPU MSGRAMs have CPU and DMA read and write access from its own CPU subsystem, and has CPU and DMA read only access from the other subsystem.

This RAM has parity.

3.12.1.5 CLA Message RAM (CLA MSGRAM)

These RAM blocks can be used to share data between the CPU and CLA. The CLA has read and write access to the "CLA to CPU MSGRAM." The CPU has read and write access to the "CPU to CLA MSGRAM." The CPU and CLA both have read access to both MSGRAMs.

This RAM has parity.

3.12.1.6 CLA-DMA MSG RAM

These RAMs blocks can be used to share data between CLA and DMA. The CLA has read and write access to the "CLA to DMA MSGRAM." The DMA has read and write access to the "DMA to CLA MSGRAM." The CLA and DMA both have read access to both MSGRAMs.

3.12.1.7 Access Arbitration

For a shared RAM, multiple accesses can happen at a given time. The maximum number of accesses to any shared RAM at any given time depends on the type of shared RAM. On this device, a combination of a fixed and round robin scheme is followed to arbitrate multiple access at any given time. Accesses from the same masters are arbitrated in a fixed priority manner, but the accesses from different masters are arbitrated using the round robin scheme.

The following is the order of fixed priority for CPU accesses:

1. Data Write/Program Write
2. Data Read
3. Program Read/Program Fetch

The following is the order of fixed priority for CLA accesses:

1. Data Write
2. Data Read/Program Fetch

Figure 3-17 represents the arbitration scheme on global shared memories:

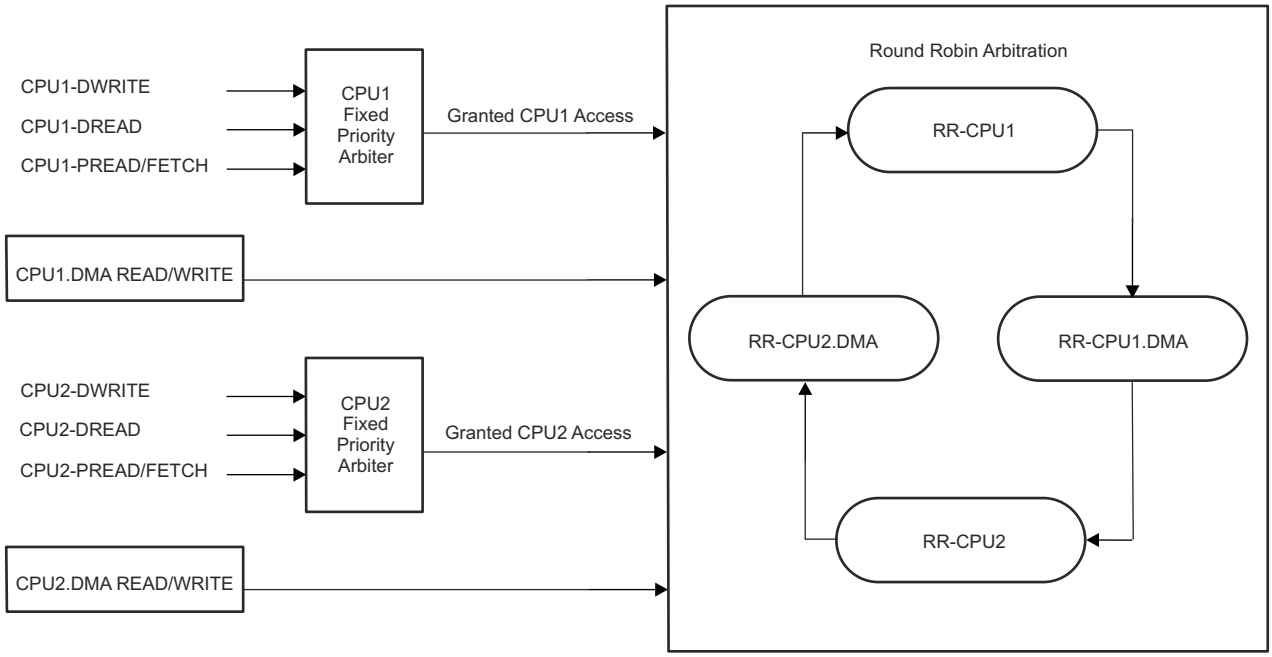


Figure 3-17. Arbitration Scheme on Global Shared Memories

Figure 3-18 represents the arbitration scheme on local shared memories.

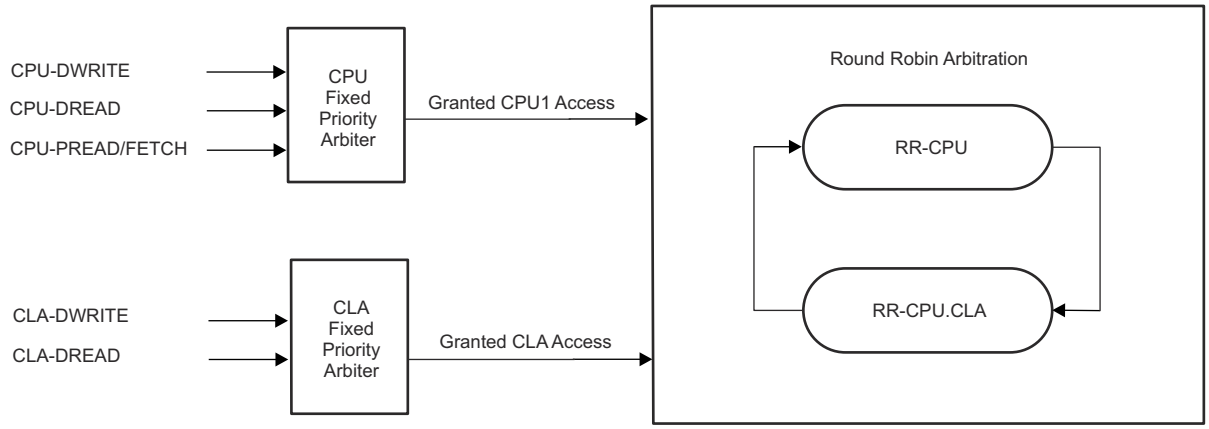


Figure 3-18. Arbitration Scheme on Local Shared Memories

3.12.1.8 Access Protection

All RAM blocks on both subsystems have different levels of protection. This feature allows the user to enable or disable specific access to individual RAM blocks from individual masters. There is no protection for read accesses, hence reads are always allowed from all the masters which have access to that RAM block.

The following sections describe the different kinds of protection available for RAM blocks on this device.

Note: For debug accesses, all the protections are disabled.

3.12.1.8.1 CPU Fetch Protection

A CPU has execution permission from a memory, only if that memory is dedicated to that CPU or the respective subsystem is master for that memory (in case of GSx memory). When fetch accesses are allowed based on the mastership, fetch accesses can be further protected by setting the FETCHPROTx bit of the specific register to 1. If fetch access is done by the CPU to a memory where CPU fetch protection is enabled, a fetch protection violation occurs.

There are two types of fetch protection violations:

- Non-master CPU fetch protection violation
- Master CPU fetch protection violation

If a fetch access is made to a memory by a non-master CPU, the fetch access is called a non-master fetch protection violation. If a fetch access is made to a dedicated or shared memory by the master CPU, and FETCHPROTx is set to 1 for that memory, the violation is called a master CPU fetch protection violation.

If a fetch protection violation occurs, the violation results in an ITRAP for CPU. A flag gets set into the appropriate access violation flag register, and the memory address for which the access violation occurred, get latched into the appropriate CPU fetch access violation address register.

3.12.1.8.2 CPU Write Protection

A CPU has write permission to a memory only if that memory is dedicated to that CPU, or if the respective subsystem is the master for that memory (in case of GSx memory). When write accesses are allowed based on the mastership, write accesses can be further protected by setting the CPUWRPROTx bit of the specific register to 1. If write access is done by a CPU to memory where the write access is protected, a write protection violation occurs.

There are two types of CPU write protection violations:

- Non-master CPU write protection violation
- Master CPU write protection violation
  If a write access is made to a dedicated or shared memory by the master CPU and CPUWRPROTx is set to 1 for that memory, the violation is called a master CPU write protection violation.

If a write protection violation occurs, write gets ignored, a flag gets set into the appropriate access violation flag register, and the memory address for which the access violation occurred, gets latched into the appropriate CPU write access violation address register. Also, an access violation interrupt is generated if enabled in the interrupt enable register.

3.12.1.8.3 CPU Read Protection

For local shared RAM, if memory is shared between the CPU and the CLA, the CPU only has access if the memory is configured as data RAM for the CLA. If the memory is programmed as program RAM, all the access from the CPU, including a read, is blocked and the violation is considered as a non-master access violation.

If a read protection violation occurs, a flag gets set into the appropriate access violation flag register, and the memory address for which the access violation occurred, gets latched into the appropriate CPU read access violation address register. Also, if enabled in the interrupt enable register, an access violation interrupt is generated.

3.12.1.8.4 CLA Fetch Protection

If local shared RAM is configured as dedicated RAM for the CPU, or if the RAM is configured as data RAM for the CLA, any fetch access from the CLA to that particular LSx RAM results in a CLA fetch protection violation, which is a non-master access violation.

If a CLA fetch protection violation occurs, the violation results in a MSTOP, a flag gets set into the appropriate access violation flag register, and the memory address for which the access violation occurred, gets latched into the appropriate CLA fetch access violation address register. Also, an access violation interrupt is generated to the master CPU if enabled in the interrupt enable register.

3.12.1.8.5 CLA Write Protection

If local shared RAM is configured as dedicated RAM for the CPU, or if the RAM is configured as program RAM for the CLA, any data write access from the CLA to that particular LSx RAM results in a CLA write protection violation, which is a non-master access violation. Similarly any data write access from CLA to CPUTOCLA or DMATOCLA MSGRAM results in a CLA write protection violation, which is a non-master access violation.

If a CLA write protection violation occurs, write gets ignored, a flag gets set into the appropriate access violation flag register, and the memory address for which the access violation occurred, gets latched into the appropriate CLA write access violation address register. Also, an access violation interrupt is generated to the master CPU if enabled in the interrupt enable register.

3.12.1.8.6 CLA Read Protection

If local shared RAM is configured as dedicated RAM for the CPU, or if the RAM is configured as program RAM for the CLA, any data read access from the CLA to that particular LSx RAM results in a CLA read protection violation, which is a non-master access violation.

If a CLA read protection violation occurs, a flag gets set into the appropriate access violation flag register, and the memory address for which the access violation occurred, gets latched into the appropriate CLA read access violation address register. Also, an access violation interrupt is generated to the master CPU if enabled in the interrupt enable register.

3.12.1.8.7 DMA Write Protection

The CPU1 or CPU2 DMA has write permission to a GSx memory only if the respective subsystem is master for that memory. When write accesses from a DMA are allowed based on the mastership, the write access can be further protected by setting the DMAWRPROTx bit of a specific register to 1. If write access is done by the DMA to protected memory, a write protection violation occurs.

There are two types of DMA write protection violations:

- Non-master DMA write protection violation (applicable to GSx RAMs and DMATOCLA MSGRAM)
- Master DMA write protection violation

If a write access is made to GSx memory by a non-master DMA, the write is called a non-master write protection violation. If a write access is made to a dedicated or shared memory by a master DMA, and DMAWRPROTx is set to 1 for that memory, the write is called a master DMA write protection violation.

If a DMA write protection violation occurs, the write gets ignored, a flag gets set into the appropriate access violation flag register, and the memory address for which the access violation occurred gets latched into the appropriate DMA write access violation address register. Also, an access violation interrupt is generated to the respective CPU, if enabled in the interrupt enable register. These are dedicated registers for each subsystem.

Note 1: All access protections are ignored during debug accesses. Write access to a protected memory go through when the write access is done by way of the debugger, irrespective of the write protection configuration for that memory.

Note 2: In the case of local shared RAM, if memory is shared between the CPU and the CLA, the CPU only has access if the memory is configured as data RAM for the CLA. If the memory is programmed as program RAM, all the access from the CPU (including read) and data access from the CLA is blocked, and the violation is considered as a non-master access violation. If the memory is configured as dedicated to the CPU, all access from the CLA is blocked and the violation is considered a non-master access violation.

3.12.1.9 Memory Error Detection, Correction and Error Handling

These devices have memory error detection and correction features to satisfy safety standards requirements. These requirements warrant the addition of detection mechanisms for finite dangerous failures.

In this device, all dedicated RAMs and LSx RAMs support error correction code (ECC) protection and other shared RAMs have parity protection. The ECC scheme used is Single Error Correction Double Error Detection (SECDED). The parity scheme used is even parity. ECC/Parity will cover the data bits stored in memory as well as address.

ECC/Parity calculation is done inside the memory controller module and then calculated. ECC/Parity is written into the memory along with the data. ECC/Parity is computed for 16-bit data; hence, for each 32-bit data, there will be three 7-bit ECC codes (or 3-bit parity), two of which are for data and a third one for the address.

3.12.1.9.1 Error Detection and Correction

Error detection is done while reading the data from memory. The error detection is performed for data as well as address. For parity memory, only a single-bit error gets detected, whereas in the case of ECC memory, along with a single-bit error, a double-bit error also gets detected. These errors are called correctable error and uncorrectable errors. The following are characteristics of these errors:

- Parity errors are always uncorrectable errors
- Single-bit ECC errors are correctable errors
- Double-bit ECC errors are uncorrectable errors
- Address ECC errors are also uncorrectable errors

Correctable errors get corrected by the memory controller module and then correct data is given back as read data to the master. It is also written back into the memory to prevent double-bit error due to another single-bit error at the same memory address.

Note

ECC/Parity for address is calculated for address offset only (based on RAM block size) of corresponding 32-bit aligned address. In case of LSx RAM which are 4KB RAM block, only 11 LSB of 32-bit aligned address are used. So if address is 0x8F8F, address ECC (or Parity) will be calculated for address 0x78E (11-bit offset of 32-bit aligned address). Similarly for 8KB RAM block,12-bit address offset will be used.

3.12.1.9.2 Error Handling

For each correctable error, the count in the correctable error count register increments by one. When the value in this count register becomes equal to the value configured into the correctable error threshold register, an interrupt is generated to the respective CPU, that is, if the interrupt is enabled in the correctable interrupt enable register. The user needs to configure the correctable error threshold register based on the system requirements. Also, the address for which the error occurred, gets latched into the master-specific status register and a flag gets set. Each of these registers are dedicated for each CPU subsystem.

If there are uncorrectable errors, an NMI gets generated for the respective CPU. In this case, the address for which the error occurred, also gets latched into the master-specific address status register, and a flag gets set.

Table 3-11 summarizes different error situations that can arise. These need to be handled appropriately in the software, using the status and interrupt indications provided.

Table 3-11. Error Handling in Different Scenarios

Access Type	Error Found In	Error Type	Status Indication	Error Notification
Reads	Data read from memory	Uncorrectable Error (Single-bit error for Parity RAMs OR Double bit Error for ECC RAMs)	Yes -CPUx/CPUx.DMA/CPUx.CLA1 CPU/DMA/CLA Read Error Address Register Data returned to CPUx/ CPUx.DMA/CPUx.CLA1 is incorrect	NMI for CPUx access NMI for CPUx.DMA access NMI to CPU for CPUx.CLA1 access
Reads	Data read from memory	Single-bit error for ECC RAMs	Yes - CPUx/CPUx.DMA CPU/DMA Read Error Address Register Increment single error counter	Interrupt when error counter reaches the user programmable threshold for single errors
Reads	Address	Address error	Yes - CPUx/CPUx.DMA/CPUx.CLA1 CPU/DMA/CLA Read Address Error Register Data returned to CPUx/ CPUx.DMA/CPUx.CLA1 is incorrect	NMI to CPU for CPUx access NMI to CPU for CPUx.DMA access NMI to CPU for CPUx.CLA1 access

Note

In the case of an uncorrectable error during fetch on the CPU, there is the possibility of getting an ITRAP before an NMI exception, since garbage instructions enter into the CPU pipeline before the NMI gets generated.

During debug accesses, correctable as well as uncorrectable errors are masked.

3.12.1.10 Application Test Hooks for Error Detection and Correction

Since error detection and correction logic is part of safety critical logic, safety applications need to make sure that the logic is always working fine (during run time also). To enable this, different test modes are provided to generate ecc/parity error in RAM locations. These test modes can be configured in RAM Test registers of different RAM blocks. (for example, for D0 RAM, TEST_D0 bit in DxTEST register). Different test modes are for different usage. In test mode user can modify the data bits (without modifying the ECC/Parity bits) or ECC/Parity bits directly. Using this feature, an ECC/Parity error can be injected into data. Since an uncorrectable error generates NMI, you can avoid generating this during test mode as one of the test modes (11) is provided where NMI generation gets disabled. This mode is just like functional mode except NMI generation on uncorrectable error.

Note

The memory-map for ECC/Parity bits and data bits are the same. Choose a different test mode (10) to access ECC/Parity bits.

The following tables show the bit mapping for the ECC and Parity bits when the bits are read in RAMTEST mode using their respective addresses.

Table 3-12. Mapping of ECC Bits in Read Data from ECC/Parity Address Map

Data Bits Location in Read Data	Content (ECC Memory)
6:0	ECC Code for lower 16 bits of data
7	Not Used
14:8	ECC Code for upper 16 bits of data
15	Not Used
22:16	ECC Code for address
31:23	Not Used

Table 3-13. Mapping of Parity Bits in Read Data from ECC/Parity Address Map

Data Bits Location in Read Data	Content (Parity Memory)
0	Parity for lower 16 bits of data
7:1	Not Used
8	Parity for upper 16 bits of data
15:9	Not Used
16	Parity for address
31:17	Not Used

Following is the sequence that must be followed to test the ecc/parity logic.

- Set the test mode to 01 or 10, depending on whether data field or ecc/parity field needs to be written.
- Write the data pattern into the memory.
- Set the test mode to 11 to read from memory and exercise the ecc/parity logic.
- Check the test log registers to verify the correctness of the logic.
- The above sequence can be repeated for any number of data patterns.
- Once the test is complete, re-initialize the locations used in test, and set the test mode to 00 that changes the RAM block back into functional mode.

3.12.1.11 ROM Test

ROMs are read only memory, unlike RAMs, data or parity bits cannot be modified to introduce errors for diagnostic coverage of parity checking logic. Following method is used to check health of parity checking logic in ROMs.

- Added duplicate Parity check logic and feed the same data into duplicate parity checker
- Generate uncorrectable error if parity check status of these two separate parity checkers do not match

Probability of both circuits having fault is unlikely hence Parity Errors are certainly detected.

To generate the error, a test bit FORCE_ERROR is added. When the FORCE_ERROR bit is set, the parity bit going to one of the party checkers is inverted thereby introducing an uncorrectable error. An uncorrectable error is generated only if there is an error on all parity checkers that is: address, data [15:0], and data [31:16]. This makes sure that all three parity checkers are working as expected.

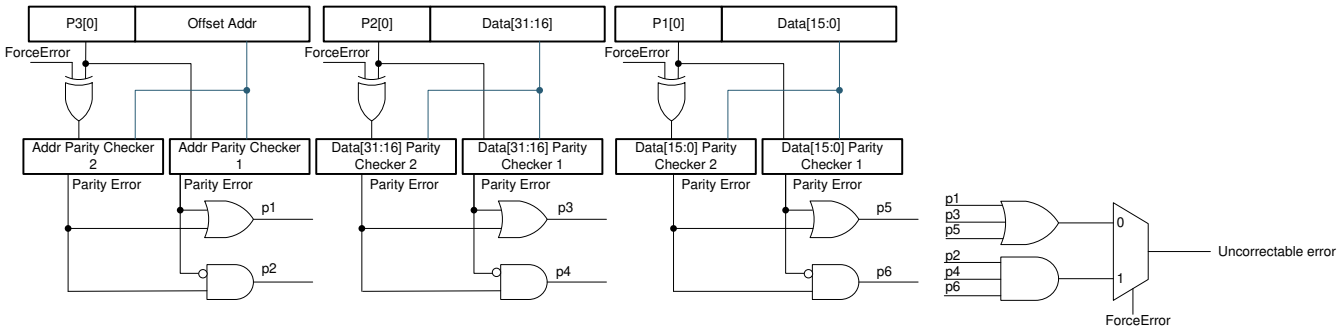


Figure 3-19. ROM Parity Checking Logic

3.12.1.12 RAM Initialization

To make sure that read/fetch from uninitialized RAM locations do not cause ECC or parity errors, the RAM_INIT feature is provided for each memory block. Using this feature, any RAM block can be initialized with 0x0 data and respective ECC/Parity bits accordingly. This can be initiated by setting the INIT bit to 1 for the specific RAM block in INIT registers. To check the status of RAM initialization, software must poll for the INITDONE bit for that RAM block in the INITDONE register to be set. Unless this bit gets set, no access must be made to that RAM memory block.

In the case of GSx memory, only the CPU of the subsystem that is configured as the master for the particular GSx RAM block can initiate the RAM initialization.

Note

The INIT bit must not be set again until the INITDONE bit for the corresponding RAM block has been polled to be set.

None of the masters must access the memory while initialization is taking place. If memory is accessed before RAMINITDONE is set, the memory read/write as well as initialization does not happen correctly.

3.13 JTAG

Gel files perform certain initialization tasks. This helps the users in a debug environment. However, when executed standalone (without the emulator connected) the application could potentially not work as expected, since there is no gel file to perform those initializations. For example, gel file disables watchdog. If user code does not service the watchdog in the application (or fails to disable the watchdog), there would be a difference in how the application behaves with the debugger and without the debugger.

Common tasks performed by the gel files (but not boot-ROM):

- On Reset:
  - Disable Flash ECC on some devices.
    - Disabling ECC only when using Flash API functions, see the Flash API User Guide for details. Otherwise, TI suggests to always program ECC and enable ECC-check.
  - Disable Watchdog
  - Enable CLA clock
  - Select real-time mode or C28x mode
- On Restart:
  - Select real-time mode or C28x mode
  - Clear IER and IFR
- On Target Connect:
  - Select real-time mode or C28x mode

For more information, see *C2000 MCU JTAG Connectivity Debug*.

3.13.1 JTAG Noise and TAP_STATUS

The TAP_STATUS register reflects the status of the JTAG TAP at any given time. Normally when no JTAG is connected to the device, the status can be IDLE. In some cases with excessive PCB noise, there can be unwanted TMS and TCK toggles that take JTAG out of the IDLE state. When persistent, this can ultimately lead to unwanted activation of the JTAG Boundary Scan or some other JTAG mode that can interfere with the intended application. To avoid this scenario, place strong enough pull resistors on the board to prevent noise from activating JTAG. As a debug tool, the TAP_STATUS register can be polled by the application code to detect if this is a cause of device disturbance. The SOFTPRES40[JTAG_nTRST] register can also be used to reset the JTAG TAP through software. Use this reset register with caution, as this prevents connecting a debugger unless the code qualifies writes to this register with some other GPIO state or other means to distinguish between noise and debugger accesses.

3.14 System Control Register Configuration Restrictions

Memory-mapped registers in the System Control operate on INTOSC1 clock domain; hence, any CPU writes to these registers requires a delay between subsequent writes otherwise a second write can be lost. The application needs to take this into consideration and add a delay in terms of the number of NOP instructions after every write to these registers that are mentioned in Table 3-14. The formula to compute delay between subsequent writes:

Delay (in SYSCLK cycles) = 3 × (F_{SYSCLK} ÷ $F_{INTOSC1}$) + 9

For Example - For SYSCLK = 100200 MHz

Delay (in SYSCLK cycles) = 3 × (100 MHz ÷ 10 MHz) + 9 = 39 SYSCLK cycles

Table 3-14. System Control Registers Impacted

Registers requiring delay after every write
CLBCLKCTL
PERCLKDIVSEL
SYSCLKDIVSEL
SYSPLLCTL1
SYSPLLMULT
WDCR
XCLKOUTDIVSEL
XTALCR
CLKSRCCTL1
CLKSRCCTL2
CLKSRCCTL3
CPU1TMR2CTL (TMR2CLKCTL)

3.15 Software

3.15.1 SYSCTL Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/sysctl

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

3.15.1.1 Missing clock detection (MCD)

FILE: sysctl_ex1_missing_clock_detection.c

This example demonstrates the missing clock detection functionality and the way to handle it. Once the MCD is simulated by disconnecting the OSCCLK to the MCD module an NMI would be generated. This NMI determines that an MCD was generated due to a clock failure which is handled in the ISR.

Before an MCD the clock frequency would be as per device initialization (200Mhz). Post MCD the frequency would move to 10Mhz or INTOSC1.

The example also shows how we can lock the PLL after missing clock, detection, by first explicitly switching the clock source to INTOSC1, resetting the missing clock detect circuit and then re-locking the PLL. Post a re-lock the clock frequency would be 200Mhz but using the INTOSC1 as clock source.

External Connections

- None.

Watch Variables

- *fail* - Indicates that a missing clock was either not detected or was not handled correctly.
- *mcd_clkfail_isr* - Indicates that the missing clock failure caused an NMI to be triggered and called an the ISR to handle it.
- *mcd_detect* - Indicates that a missing clock was detected.
- *result* - Status of a successful handling of missing clock detection

3.15.1.2 XCLKOUT (External Clock Output) Configuration

FILE: sysctl_ex2_xclkout_config.c

This example demonstrates how to configure the XCLKOUT pin for observing internal clocks through an external pin, for debugging and testing purposes.

In this example, we are using INTOSC1 as the XCLKOUT clock source and configuring the divider as 8. Expected frequency of XCLKOUT = (INTOSC1 freq)/8 = 10/8 = 1.25MHz

View the XCLKOUT on GPIO73 using an oscilloscope.

3.15.2 MEMCFG Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/memcfg

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

3.15.2.1 Correctable & Uncorrectable Memory Error Handling

FILE: memcfg_ex1_error_handling.c

This example demonstrates error handling in case of various erroneous memory read/write operations. Error handling in case of CPU read/write violations, correctable & uncorrectable memory errors has been demonstrated. Correctable memory errors & violations can generate SYS_INT interrupt to CPU while uncorrectable errors lead to NMI generation.

External Connections

- None

Watch Variables

- *testStatusGlobal* - Equivalent to *TEST_PASS* if test finished correctly, else the value is set to *TEST_FAIL*
- *errCountGlobal* - Error counter

3.15.2.2 Shared RAM Management (CPU1) - C28X_DUAL

FILE: memcfg_ex1_ram_management_cpu1.c

This example shows how to assign shared RAM for use by both the CPU2 and CPU1 core. Shared RAM regions are defined in both the CPU2 and CPU1 linker files. In this example GS0 and GS14 are assigned to/owned by CPU2. The remaining shared RAM regions are owned by CPU1.

In this example, a pattern is written to cpu1RWArray and then an IPC flag is sent to notify CPU2 that data is ready to be read. CPU2 then reads the data from cpu2RArray and writes a modified pattern to cpu2RWArray. Once CPU2 acknowledges the IPC flag, CPU1 reads the data from cpu1RArray and compares with expected result.

A timer ISR is also serviced in both CPUs. The ISRs are copied into the shared RAM region owned by the respective CPUs. Each ISR toggles a GPIO. Watch the GPIOs on an oscilloscope, or if using the controlCARD, watch LED1 and LED2 blink at different rates.

Following are the memory allocation details of CPU Timer interrupt ISRs & read(R)/read write(RW) arrays in CPU1 & CPU2 as configured in the example.

- cpu1RWArray[] is mapped to shared RAM GS1
- cpu1RArray[] is mapped to shared RAM GS0
- cpu2RArray[] is mapped to shared RAM GS1
- cpu2RWArray[] is mapped to shared RAM GS0
- cpuTimer0ISR in CPU2 is copied to shared RAM GS14, toggles LED1
- cpuTimer0ISR in CPU1 is copied to shared RAM GS15, toggles LED2

Watch Variables

- error Indicates that the data written is not correctly received by the other CPU.

3.15.2.3 Shared RAM Management (CPU2) - C28X_DUAL

FILE: memcfg_ex1_ram_management_cpu2.c

This example shows how to assign shared RAM for use by both the CPU2 and CPU1 core. Shared RAM regions are defined in both the CPU2 and CPU1 linker files. In this example GS0 and GS14 are assigned to/owned by CPU2. The remaining shared RAM regions are owned by CPU1.

In this example, a pattern is written to cpu1RWArray and then an IPC flag is sent to notify CPU2 that data is ready to be read. CPU2 then reads the data from cpu2RArray and writes a modified pattern to cpu2RWArray. Once CPU2 acknowledges the IPC flag, CPU1 reads the data from cpu1RArray and compares with expected result.

A timer ISR is also serviced in both CPUs. The ISRs are copied into the shared RAM region owned by the respective CPUs. Each ISR toggles a GPIO. Watch the GPIOs on an oscilloscope, or if using the controlCARD, watch LED1 and LED2 blink at different rates.

Following are the memory allocation details of CPU Timer interrupt ISRs & read(R)/read write(RW) arrays in CPU1 & CPU2 as configured in the example.

- cpu1RWArray[] is mapped to shared RAM GS1
- cpu1RArray[] is mapped to shared RAM GS0
- cpu2RArray[] is mapped to shared RAM GS1
- cpu2RWArray[] is mapped to shared RAM GS0
- cpuTimer0ISR in CPU2 is copied to shared RAM GS14, toggles LED1
- cpuTimer0ISR in CPU1 is copied to shared RAM GS15, toggles LED2

3.15.2.4 Demonstrate memconfig diagnostics and error handling. - CM

FILE: memcfg_ex1_error_handling.c

This example demonstrates how to configure the diagnostic mode and induce ECC errors. This example induces single and two bit ECC errors in E0RAM and tries to read the corrupted location in diagnostic and functional mode. cm_common_config_c28x example needs to be run on C28x1 before running this example on the CM core

External Connections

- None

Watch Variables

- *testStatus* - Equivalent to *TEST_PASS* if test finished correctly, else the value is set to *TEST_FAIL*
- *errorGlobalCount* - Error counter
- *retx* - Individual test status

3.15.2.5 Shared RAM Management (CPU1) - C28X_DUAL

FILE: memcfg_ex2_ram_sysconfig_cpu1.c

This example shows how to assign shared RAM for use by both the CPU2 and CPU1 core. Shared RAM regions are defined in both the CPU2 and CPU1 linker files. In this example GS0 and GS14 are assigned to/owned by CPU2. The remaining shared RAM regions are owned by CPU1.

In this example, a pattern is written to cpu1RWArray and then an IPC flag is sent to notify CPU2 that data is ready to be read. CPU2 then reads the data from cpu2RArray and writes a modified pattern to cpu2RWArray. Once CPU2 acknowledges the IPC flag, CPU1 reads the data from cpu1RArray and compares with expected result.

A timer ISR is also serviced in both CPUs. The ISRs are copied into the shared RAM region owned by the respective CPUs. Each ISR toggles a GPIO. Watch the GPIOs on an oscilloscope, or if using the controlCARD, watch LED1 and LED2 blink at different rates.

Following are the memory allocation details of CPU Timer interrupt ISRs & read(R)/read write(RW) arrays in CPU1 & CPU2 as configured in the example.

- cpu1RWArray[] is mapped to shared RAM GS1
- cpu1RArray[] is mapped to shared RAM GS0
- cpu2RArray[] is mapped to shared RAM GS1
- cpu2RWArray[] is mapped to shared RAM GS0
- cpuTimer0ISR in CPU2 is copied to shared RAM GS14, toggles LED1
- cpuTimer0ISR in CPU1 is copied to shared RAM GS15, toggles LED2

Watch Variables

- error Indicates that the data written is not correctly received by the other CPU.

3.15.2.6 Shared RAM Management (CPU2) - C28X_DUAL

FILE: memcfg_ex2_ram_sysconfig_cpu2.c

This example shows how to assign shared RAM for use by both the CPU2 and CPU1 core. Shared RAM regions are defined in both the CPU2 and CPU1 linker files. In this example GS0 and GS14 are assigned to/owned by CPU2. The remaining shared RAM regions are owned by CPU1.

In this example, a pattern is written to cpu1RWArray and then an IPC flag is sent to notify CPU2 that data is ready to be read. CPU2 then reads the data from cpu2RArray and writes a modified pattern to cpu2RWArray. Once CPU2 acknowledges the IPC flag, CPU1 reads the data from cpu1RArray and compares with expected result.

A timer ISR is also serviced in both CPUs. The ISRs are copied into the shared RAM region owned by the respective CPUs. Each ISR toggles a GPIO. Watch the GPIOs on an oscilloscope, or if using the controlCARD, watch LED1 and LED2 blink at different rates.

Following are the memory allocation details of CPU Timer interrupt ISRs & read(R)/read write(RW) arrays in CPU1 & CPU2 as configured in the example.

- cpu1RWArray[] is mapped to shared RAM GS1
- cpu1RArray[] is mapped to shared RAM GS0
- cpu2RArray[] is mapped to shared RAM GS1
- cpu2RWArray[] is mapped to shared RAM GS0
- cpuTimer0ISR in CPU2 is copied to shared RAM GS14, toggles LED1
- cpuTimer0ISR in CPU1 is copied to shared RAM GS15, toggles LED2

3.15.3 NMI Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/nmi

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

3.15.3.1 NMI handling - C28X_DUAL

FILE: nmi_ex1_cpu1handling.c

This example demonstrates how to handle an NMI.

The watchdog of CPU2 is configured to reset the core once the watchdog overflows and in the CPU1 the NMI is triggered. The NMI status is read and is verified to be due to CPU2 Watchdog reset. The NMI ISR reboots the CPU2 core and the process is repeated.

Watch Variables

- nmi_isr_count Indicates the number of times the NMI ISR was hit because of CPU2 watchdog reset.

3.15.3.2 Watchdog Reset - C28X_DUAL

FILE: nmi_ex1_cpu2wdreset.c

This example shows how to configure the watchdog to reset CPU2 which will trigger an NMI in CPU1. LED1 is toggled at the start of main indicating CPU reset.

External Connections

- None.

Watch Variables

- loopCount - The number of loops performed while not in ISR

3.15.3.3 NMI handling - C28X_DUAL

FILE: nmi_ex2_sysconfig_cpu1.c

This example demonstrates how to handle an NMI.

The watchdog of CPU2 is configured to reset the core once the watchdog overflows and in the CPU1 the NMI is triggered. The NMI status is read and is verified to be due to CPU2 Watchdog reset. The NMI ISR reboots the CPU2 core and the process is repeated.

Watch Variables

- nmi_isr_count Indicates the number of times the NMI ISR was hit because of CPU2 watchdog reset.

3.15.3.4 Watchdog Reset - C28X_DUAL

FILE: nmi_ex2_sysconfig_cpu2.c

This example shows how to configure the watchdog to reset CPU2 which will trigger an NMI in CPU1. LED1 is toggled at the start of main indicating CPU reset.

External Connections

- None.

Watch Variables

- loopCount - The number of loops performed while not in ISR

3.15.4 TIMER Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/timer

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

3.15.4.1 CPU Timers

FILE: timer_ex1_cputimers.c

This example configures CPU Timer0, 1, and 2 and increments a counter each time the timer asserts an interrupt.

External Connections

- None

Watch Variables

- cpuTimer0IntCount
- cpuTimer1IntCount
- cpuTimer2IntCount

3.15.4.2 CPU Timers - CM

FILE: timer_ex1_cputimers.c

This example configures CPU Timer0, 1, and 2 and increments a counter each time the timer asserts an interrupt.

Before running this example, please run the cm_common_config_c28x Example from the c28x folder. It will initialize the clock and configure the GPIOs.

External Connections

- None

Watch Variables

- cpuTimer0IntCount
- cpuTimer1IntCount
- cpuTimer2IntCount

3.15.4.3 CPU Timers

FILE: timer_ex1_cputimers_sysconfig.c

This example configures CPU Timer0, 1, and 2 and increments a counter each time the timer asserts an interrupt.

External Connections

- None

Watch Variables

- cpuTimer0IntCount
- cpuTimer1IntCount
- cpuTimer2IntCount

3.15.5 WATCHDOG Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/watchdog

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

3.15.5.1 Watchdog

FILE: watchdog_ex1_service.c

This example shows how to service the watchdog or generate a wakeup interrupt using the watchdog. By default the example will generate a Wake interrupt. To service the watchdog and not generate the interrupt, uncomment the SysCtl_serviceWatchdog() line in the main for loop.

External Connections

- None.

Watch Variables

- wakeCount - The number of times entered into the watchdog ISR
- loopCount - The number of loops performed while not in ISR

3.15.5.2 Windowed watchdog expiry with NMI handling - CM

FILE: watchdog_ex1_nmi_handling.c

This program demonstrates an NMI generation to the CM4 core when the Windowed watchdog (WWD) expires.

A delay is provided after enabling the WWD to make the watchdog count up from 0 to 0xFF. Once 0 is reached, an NMI is triggered. Currently on triggering an NMI, a status flag is set indicating if the NMI was handled after the WWD expired.

External Connections

- None

Watch Variables

- *wdstatus* - Indicates if the WWD caused an NMI on expiry.
- *cmnmi* - Indicates if the NMI was handled after the WWD expired
- *fail* - Status if the Windowed watchdog expired generating an NMI with proper NMI handling

3.16 System Control Registers

This section describes the various System Control Registers.

3.16.1 SYSCTRL Base Address Table (C28)

Table 3-15. SYSCTRL Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
AccessProtectionRegs	ACCESS_PROTECTION_REGS	ACCESSPROTECTION_BASE	0x0005_F500	YES	YES	-	-	YES
ClkCfgRegs	CLK_CFG_REGS	CLKCFG_BASE	0x0005_D200	YES	YES	-	-	YES
CmConfRegs	CM_CONF_REGS	CMCONF_BASE	0x0005_DC00	YES	-	-	-	YES
CpuIdRegs	CPU_ID_REGS	CPUID_BASE	0x0007_0223	YES	YES	-	-	-
CpuSysRegs	CPU_SYS_REGS	CPUSYS_BASE	0x0005_D300	YES	YES	-	-	YES
CpuTimer0Regs	CPUTIMER_REGS	CPUTIMER0_BASE	0x0000_0C00	YES	YES	-	-	-
CpuTimer1Regs	CPUTIMER_REGS	CPUTIMER1_BASE	0x0000_0C08	YES	YES	-	-	-
CpuTimer2Regs	CPUTIMER_REGS	CPUTIMER2_BASE	0x0000_0C10	YES	YES	-	-	-
DevCfgRegs	DEV_CFG_REGS	DEVCFG_BASE	0x0005_D000	YES	-	-	-	YES
DmaClaSrcSelRegs	DMA_CLA_SRC_SEL_REGS	DMACLASRCSEL_BASE	0x0000_7980	YES	YES	-	-	YES
MemCfgRegs	MEM_CFG_REGS	MEMCFG_BASE	0x0005_F400	YES	YES	-	-	YES
MemoryErrorRegs	MEMORY_ERROR_REGS	MEMORYERROR_BASE	0x0005_F540	YES	YES	-	-	YES
NmiIntruptRegs	NMI_INTRUPT_REGS	NMI_BASE	0x0000_7060	YES	YES	-	-	YES
PieCtrlRegs	PIE_CTRL_REGS	PIECTRL_BASE	0x0000_0CE0	YES	YES	-	-	-
PieVectTable	PIE_VECT_TABLE	PIEVECTTABLE_BASE	0x0000_0D00	YES	YES	-	-	-
RomPrefetchRegs	ROM_PREFETCH_REGS	ROMPREFETCH_BASE	0x0005_F588	YES	YES	-	-	YES
RomWaitStateRegs	ROM_WAIT_STATE_REGS	ROMWAITSTATE_BASE	0x0005_F580	YES	YES	-	-	YES
SyncSocRegs	SYNC_SOC_REGS	SYNCSOC_BASE	0x0000_7940	YES	-	-	-	YES
SysPeriphAcRegs	CPU1_PERIPH_AC_REGS	PERIPHAC_BASE	0x0005_D500	YES	-	-	-	YES
SysPeriphAcRegs	CPU2_PERIPH_AC_REGS	PERIPHAC_BASE	0x0005_D500	-	YES	-	-	YES
SysStatusRegs	SYS_STATUS_REGS	SYSSTAT_BASE	0x0005_D400	YES	YES	-	-	YES
TestErrorRegs	TEST_ERROR_REGS	TESTERROR_BASE	0x0005_F590	YES	YES	-	-	YES
UidRegs	UID_REGS	UID_BASE	0x0007_0200	YES	YES	-	-	-
WdRegs	WD_REGS	WD_BASE	0x0000_7000	YES	YES	-	-	YES
XintRegs	XINT_REGS	XINT_BASE	0x0000_7070	YES	YES	-	-	YES

3.16.2 ACCESS_PROTECTION_REGS Registers

Table 3-16 lists the memory-mapped registers for the ACCESS_PROTECTION_REGS registers. All register offset addresses not listed in Table 3-16 should be considered as reserved locations and the register contents should not be modified.

Table 3-16. ACCESS_PROTECTION_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	NMAVFLG	Non-Master Access Violation Flag Register		Go
2h	NMAVSET	Non-Master Access Violation Flag Set Register	EALLOW	Go
4h	NMAVCLR	Non-Master Access Violation Flag Clear Register	EALLOW	Go
6h	NMAVINTEN	Non-Master Access Violation Interrupt Enable Register	EALLOW	Go
8h	NMCPURDAVADDR	Non-Master CPU Read Access Violation Address		Go
Ah	NMCPUWRAVADDR	Non-Master CPU Write Access Violation Address		Go
Ch	NMCPUFAVADDR	Non-Master CPU Fetch Access Violation Address		Go
Eh	NMDMAWRAVADDR	Non-Master DMA Write Access Violation Address		Go
10h	NMCLA1RDAVADDR	Non-Master CLA1 Read Access Violation Address		Go
12h	NMCLA1WRAVADDR	Non-Master CLA1 Write Access Violation Address		Go
14h	NMCLA1FAVADDR	Non-Master CLA1 Fetch Access Violation Address		Go
1Ch	NMDMARDAVADDR	Non-Master DMA Read Access Violation Address		Go
20h	MAVFLG	Master Access Violation Flag Register		Go
22h	MAVSET	Master Access Violation Flag Set Register	EALLOW	Go
24h	MAVCLR	Master Access Violation Flag Clear Register	EALLOW	Go
26h	MAVINTEN	Master Access Violation Interrupt Enable Register	EALLOW	Go
28h	MCPUFAVADDR	Master CPU Fetch Access Violation Address		Go
2Ah	MCPUWRAVADDR	Master CPU Write Access Violation Address		Go
2Ch	MDMAWRAVADDR	Master DMA Write Access Violation Address		Go

Complex bit access types are encoded to fit into small table cells. Table 3-17 shows the codes that are used for access types in this section.

Table 3-17. ACCESS_PROTECTION_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 3-17. ACCESS_PROTECTION_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.2.1 NMAVFLG Register (Offset = 0h) [Reset = 0h]

NMAVFLG is shown in Figure 3-20 and described in Table 3-18.

Return to the Summary Table.

Non-Master Access Violation Flag Register

Figure 3-20. NMAVFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					DMAREAD	RESERVED	RESERVED
R-0h					R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	CLA1FETCH	CLA1WRITE	CLA1READ	DMAWRITE	CPUFETCH	CPUWRITE	CPUREAD
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 3-18. NMAVFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-11	RESERVED	R	0h	Reserved
10	DMAREAD	R	0h	Non Master DMA Read Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	RESERVED	R	0h	Reserved
6	CLA1FETCH	R	0h	Non Master CLA1 Fetch Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
5	CLA1WRITE	R	0h	Non Master CLA1 Write Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
4	CLA1READ	R	0h	Non Master CLA1 Read Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
3	DMAWRITE	R	0h	Non Master DMA Write Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
2	CPUFETCH	R	0h	Non Master CPU Fetch Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn

Table 3-18. NMAVFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	CPUWRITE	R	0h	Non Master CPU Write Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
0	CPUREAD	R	0h	Non Master CPU Read Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn

3.16.2.2 NMAVSET Register (Offset = 2h) [Reset = 0h]

NMAVSET is shown in Figure 3-21 and described in Table 3-19.

Return to the Summary Table.

Non-Master Access Violation Flag Set Register

Figure 3-21. NMAVSET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					DMAREAD	RESERVED	RESERVED
R-0h					R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
RESERVED	CLA1FETCH	CLA1WRITE	CLA1READ	DMAWRITE	CPUFETCH	CPUWRITE	CPUREAD
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-19. NMAVSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-11	RESERVED	R	0h	Reserved
10	DMAREAD	R-0/W1S	0h	0: No action. 1: DMA Read Access Violation Flag in NMAVFLG register will be set and interrupt will be generated if enabled.. Reset type: SYSRSn
9	RESERVED	R-0/W1S	0h	Reserved
8	RESERVED	R-0/W1S	0h	Reserved
7	RESERVED	R-0/W1S	0h	Reserved
6	CLA1FETCH	R-0/W1S	0h	0: No action. 1: CLA1 Fetch Access Violation Flag in NMAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
5	CLA1WRITE	R-0/W1S	0h	0: No action. 1: CLA1 Write Access Violation Flag in NMAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
4	CLA1READ	R-0/W1S	0h	0: No action. 1: CLA1 Read Access Violation Flag in NMAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
3	DMAWRITE	R-0/W1S	0h	0: No action. 1: DMA Write Access Violation Flag in NMAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
2	CPUFETCH	R-0/W1S	0h	0: No action. 1: CPU Fetch Access Violation Flag in NMAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn

Table 3-19. NMAVSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	CPUWRITE	R-0/W1S	0h	0: No action. 1: CPU Write Access Violation Flag in NMAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
0	CPUREAD	R-0/W1S	0h	0: No action. 1: CPU Read Access Violation Flag in NMAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn

3.16.2.3 NMAVCLR Register (Offset = 4h) [Reset = 0h]

NMAVCLR is shown in Figure 3-22 and described in Table 3-20.

Return to the Summary Table.

Non-Master Access Violation Flag Clear Register

Figure 3-22. NMAVCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					DMAREAD	RESERVED	RESERVED
R-0h					R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
RESERVED	CLA1FETCH	CLA1WRITE	CLA1READ	DMAWRITE	CPUFETCH	CPUWRITE	CPUREAD
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-20. NMAVCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-11	RESERVED	R	0h	Reserved
10	DMAREAD	R-0/W1S	0h	0: No action. 1: DMA Read Access Violation Flag in NMAVFLG register will be cleared. Reset type: SYSRSn
9	RESERVED	R-0/W1S	0h	Reserved
8	RESERVED	R-0/W1S	0h	Reserved
7	RESERVED	R-0/W1S	0h	Reserved
6	CLA1FETCH	R-0/W1S	0h	0: No action. 1: CLA1 Fetch Access Violation Flag in NMAVFLG register will be cleared. Reset type: SYSRSn
5	CLA1WRITE	R-0/W1S	0h	0: No action. 1: CLA1 Write Access Violation Flag in NMAVFLG register will be cleared. Reset type: SYSRSn
4	CLA1READ	R-0/W1S	0h	0: No action. 1: CLA1 Read Access Violation Flag in NMAVFLG register will be cleared. Reset type: SYSRSn
3	DMAWRITE	R-0/W1S	0h	0: No action. 1: DMA Write Access Violation Flag in NMAVFLG register will be cleared. Reset type: SYSRSn
2	CPUFETCH	R-0/W1S	0h	0: No action. 1: CPU Fetch Access Violation Flag in NMAVFLG register will be cleared. Reset type: SYSRSn

Table 3-20. NMAVCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	CPUWRITE	R-0/W1S	0h	0: No action. 1: CPU Write Access Violation Flag in NMAVFLG register will be cleared. Reset type: SYSRSn
0	CPUREAD	R-0/W1S	0h	0: No action. 1: CPU Read Access Violation Flag in NMAVFLG register will be cleared. Reset type: SYSRSn

3.16.2.4 NMAVINTEN Register (Offset = 6h) [Reset = 0h]

NMAVINTEN is shown in Figure 3-23 and described in Table 3-21.

Return to the Summary Table.

Non-Master Access Violation Interrupt Enable Register

Figure 3-23. NMAVINTEN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					DMAREAD	RESERVED	RESERVED
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	CLA1FETCH	CLA1WRITE	CLA1READ	DMAWRITE	CPUFETCH	CPUWRITE	CPUREAD
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-21. NMAVINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-11	RESERVED	R	0h	Reserved
10	DMAREAD	R/W	0h	0: DMA Non Master Read Access Violation Interrupt is disabled. 1: DMA Non Master Read Access Violation Interrupt is enabled. Reset type: SYSRSn
9	RESERVED	R/W	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	CLA1FETCH	R/W	0h	0: CLA1 Non Master Fetch Access Violation Interrupt is disabled. 1: CLA1 Non Master Fetch Access Violation Interrupt is enabled. Reset type: SYSRSn
5	CLA1WRITE	R/W	0h	0: CLA1 Non Master Write Access Violation Interrupt is disabled. 1: CLA1 Non Master Write Access Violation Interrupt is enabled. Reset type: SYSRSn
4	CLA1READ	R/W	0h	0: CLA1 Non Master Read Access Violation Interrupt is disabled. 1: CLA1 Non Master Read Access Violation Interrupt is enabled. Reset type: SYSRSn
3	DMAWRITE	R/W	0h	0: DMA Non Master Write Access Violation Interrupt is disabled. 1: DMA Non Master Write Access Violation Interrupt is enabled. Reset type: SYSRSn
2	CPUFETCH	R/W	0h	0: CPU Non Master Fetch Access Violation Interrupt is disabled. 1: CPU Non Master Fetch Access Violation Interrupt is enabled. Reset type: SYSRSn
1	CPUWRITE	R/W	0h	0: CPU Non Master Write Access Violation Interrupt is disabled. 1: CPU Non Master Write Access Violation Interrupt is enabled. Reset type: SYSRSn
0	CPUREAD	R/W	0h	0: CPU Non Master Read Access Violation Interrupt is disabled. 1: CPU Non Master Read Access Violation Interrupt is enabled. Reset type: SYSRSn

3.16.2.5 NMCPURDAVADDR Register (Offset = 8h) [Reset = 0h]

NMCPURDAVADDR is shown in Figure 3-24 and described in Table 3-22.

Return to the Summary Table.

Non-Master CPU Read Access Violation Address

Figure 3-24. NMCPURDAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMCPURDAVADDR																															
R-0h																															

Table 3-22. NMCPURDAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NMCPURDAVADDR	R	0h	This register captures the address location for which non master CPU read access vaiolation occurred. Reset type: SYSRSn

3.16.2.6 NMCPUWRAVADDR Register (Offset = Ah) [Reset = 0h]

NMCPUWRAVADDR is shown in Figure 3-25 and described in Table 3-23.

Return to the Summary Table.

Non-Master CPU Write Access Violation Address

Figure 3-25. NMCPUWRAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMCPUWRAVADDR																															
R-0h																															

Table 3-23. NMCPUWRAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NMCPUWRAVADDR	R	0h	This register captures the address location for which non master CPU write access vaiolation occurred. Reset type: SYSRSn

3.16.2.7 NMCPUFAVADDR Register (Offset = Ch) [Reset = 0h]

NMCPUFAVADDR is shown in Figure 3-26 and described in Table 3-24.

Return to the Summary Table.

Non-Master CPU Fetch Access Violation Address

Figure 3-26. NMCPUFAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMCPUFAVADDR																															
R-0h																															

Table 3-24. NMCPUFAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NMCPUFAVADDR	R	0h	This register captures the address location for which non master CPU fetch access vaiolation occurred. Reset type: SYSRSn

3.16.2.8 NMDMAWRAVADDR Register (Offset = Eh) [Reset = 0h]

NMDMAWRAVADDR is shown in Figure 3-27 and described in Table 3-25.

Return to the Summary Table.

Non-Master DMA Write Access Violation Address

Figure 3-27. NMDMAWRAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMDMAWRAVADDR																															
R-0h																															

Table 3-25. NMDMAWRAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NMDMAWRAVADDR	R	0h	This register captures the address location for which non master DMA write access vaiolation occurred. Reset type: SYSRSn

3.16.2.9 NMCLA1RDAVADDR Register (Offset = 10h) [Reset = 0h]

NMCLA1RDAVADDR is shown in Figure 3-28 and described in Table 3-26.

Return to the Summary Table.

Non-Master CLA1 Read Access Violation Address

Figure 3-28. NMCLA1RDAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMCLA1RDAVADDR																															
R-0h																															

Table 3-26. NMCLA1RDAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NMCLA1RDAVADDR	R	0h	This register captures the address location for which non master CLA1 read access vaiolation occurred. Reset type: SYSRSn

3.16.2.10 NMCLA1WRAVADDR Register (Offset = 12h) [Reset = 0h]

NMCLA1WRAVADDR is shown in Figure 3-29 and described in Table 3-27.

Return to the Summary Table.

Non-Master CLA1 Write Access Violation Address

Figure 3-29. NMCLA1WRAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMCLA1WRAVADDR																															
R-0h																															

Table 3-27. NMCLA1WRAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NMCLA1WRAVADDR	R	0h	This register captures the address location for which non master CLA1 write access vaiolation occurred. Reset type: SYSRSn

3.16.2.11 NMCLA1FAVADDR Register (Offset = 14h) [Reset = 0h]

NMCLA1FAVADDR is shown in Figure 3-30 and described in Table 3-28.

Return to the Summary Table.

Non-Master CLA1 Fetch Access Violation Address

Figure 3-30. NMCLA1FAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMCLA1FAVADDR																															
R-0h																															

Table 3-28. NMCLA1FAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NMCLA1FAVADDR	R	0h	This register captures the address location for which non master CLA1 fetch access vaiolation occurred. Reset type: SYSRSn

3.16.2.12 NMDMARDAVADDR Register (Offset = 1Ch) [Reset = 0h]

NMDMARDAVADDR is shown in Figure 3-31 and described in Table 3-29.

Return to the Summary Table.

Non-Master DMA Read Access Violation Address

Figure 3-31. NMDMARDAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMDMARDAVADDR																															
R-0h																															

Table 3-29. NMDMARDAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NMDMARDAVADDR	R	0h	This register captures the address location for which non master DMA read access vaiolation occurred. Reset type: SYSRSn

3.16.2.13 MAVFLG Register (Offset = 20h) [Reset = 0h]

MAVFLG is shown in Figure 3-32 and described in Table 3-30.

Return to the Summary Table.

Master Access Violation Flag Register

Figure 3-32. MAVFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					DMAWRITE	CPUWRITE	CPUFETCH
R-0h					R-0h	R-0h	R-0h

Table 3-30. MAVFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-3	RESERVED	R	0h	Reserved
2	DMAWRITE	R	0h	Master DMA Write Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
1	CPUWRITE	R	0h	Master CPU Write Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
0	CPUFETCH	R	0h	Master CPU Fetch Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn

3.16.2.14 MAVSET Register (Offset = 22h) [Reset = 0h]

MAVSET is shown in Figure 3-33 and described in Table 3-31.

Return to the Summary Table.

Master Access Violation Flag Set Register

Figure 3-33. MAVSET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					DMAWRITE	CPUWRITE	CPUFETCH
R-0h					R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-31. MAVSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-3	RESERVED	R	0h	Reserved
2	DMAWRITE	R-0/W1S	0h	0: No action. 1: DMA Write Access Violation Flag in MAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
1	CPUWRITE	R-0/W1S	0h	0: No action. 1: CPU Write Access Violation Flag in MAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
0	CPUFETCH	R-0/W1S	0h	0: No action. 1: CPU Fetch Access Violation Flag in MAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn

3.16.2.15 MAVCLR Register (Offset = 24h) [Reset = 0h]

MAVCLR is shown in Figure 3-34 and described in Table 3-32.

Return to the Summary Table.

Master Access Violation Flag Clear Register

Figure 3-34. MAVCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					DMAWRITE	CPUWRITE	CPUFETCH
R-0h					R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-32. MAVCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-3	RESERVED	R	0h	Reserved
2	DMAWRITE	R-0/W1S	0h	0: No action. 1: DMA Write Access Violation Flag in MAVFLG register will be cleared. Reset type: SYSRSn
1	CPUWRITE	R-0/W1S	0h	0: No action. 1: CPU Write Access Violation Flag in MAVFLG register will be cleared . Reset type: SYSRSn
0	CPUFETCH	R-0/W1S	0h	0: No action. 1: CPU Fetch Access Violation Flag in MAVFLG register will be cleared. Reset type: SYSRSn

3.16.2.16 MAVINTEN Register (Offset = 26h) [Reset = 0h]

MAVINTEN is shown in Figure 3-35 and described in Table 3-33.

Return to the Summary Table.

Master Access Violation Interrupt Enable Register

Figure 3-35. MAVINTEN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					DMAWRITE	CPUWRITE	CPUFETCH
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 3-33. MAVINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-3	RESERVED	R	0h	Reserved
2	DMAWRITE	R/W	0h	0: DMA Write Access Violation Interrupt is disabled. 1: DMA Write Access Violation Interrupt is enabled. Reset type: SYSRSn
1	CPUWRITE	R/W	0h	0: CPU Write Access Violation Interrupt is disabled. 1: CPU Write Access Violation Interrupt is enabled. Reset type: SYSRSn
0	CPUFETCH	R/W	0h	0: CPU Fetch Access Violation Interrupt is disabled. 1: CPU Fetch Access Violation Interrupt is enabled. Reset type: SYSRSn

3.16.2.17 MCPUFAVADDR Register (Offset = 28h) [Reset = 0h]

MCPUFAVADDR is shown in Figure 3-36 and described in Table 3-34.

Return to the Summary Table.

Master CPU Fetch Access Violation Address

Figure 3-36. MCPUFAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCPUFAVADDR																															
R-0h																															

Table 3-34. MCPUFAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MCPUFAVADDR	R	0h	This register captures the address location for which master CPU fetch access vaiolation occurred. Reset type: SYSRSn

3.16.2.18 MCPUWRAVADDR Register (Offset = 2Ah) [Reset = 0h]

MCPUWRAVADDR is shown in Figure 3-37 and described in Table 3-35.

Return to the Summary Table.

Master CPU Write Access Violation Address

Figure 3-37. MCPUWRAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCPUWRAVADDR																															
R-0h																															

Table 3-35. MCPUWRAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MCPUWRAVADDR	R	0h	This register captures the address location for which master CPU write access vaiolation occurred. Reset type: SYSRSn

3.16.2.19 MDMAWRAVADDR Register (Offset = 2Ch) [Reset = 0h]

MDMAWRAVADDR is shown in Figure 3-38 and described in Table 3-36.

Return to the Summary Table.

Master DMA Write Access Violation Address

Figure 3-38. MDMAWRAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MDMAWRAVADDR																															
R-0h																															

Table 3-36. MDMAWRAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MDMAWRAVADDR	R	0h	This register captures the address location for which master DMA write access vaiolation occurred. Reset type: SYSRSn

3.16.3 CLK_CFG_REGS Registers

Table 3-37 lists the memory-mapped registers for the CLK_CFG_REGS registers. All register offset addresses not listed in Table 3-37 should be considered as reserved locations and the register contents should not be modified.

Table 3-37. CLK_CFG_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CLKSEM	Clock Control Semaphore Register	EALLOW	Go
2h	CLKCFGLOCK1	Lock bit for CLKCFG registers	EALLOW	Go
8h	CLKSRCCTL1	Clock Source Control register-1	EALLOW	Go
Ah	CLKSRCCTL2	Clock Source Control register-2	EALLOW	Go
Ch	CLKSRCCTL3	Clock Source Control register-3	EALLOW	Go
Eh	SYSPLLCTL1	SYSPLL Control register-1	EALLOW	Go
14h	SYSPLLMULT	SYSPLL Multiplier register	EALLOW	Go
16h	SYSPLLSTS	SYSPLL Status register		Go
18h	AUXPLLCTL1	AUXPLL Control register-1	EALLOW	Go
1Eh	AUXPLLMULT	AUXPLL Multiplier register	EALLOW	Go
20h	AUXPLLSTS	AUXPLL Status register		Go
22h	SYSCLKDIVSEL	System Clock Divider Select register	EALLOW	Go
24h	AUXCLKDIVSEL	Auxillary Clock Divider Select register	EALLOW	Go
26h	PERCLKDIVSEL	Peripheral Clock Divider Selet register	EALLOW	Go
28h	XCLKOUTDIVSEL	XCLKOUT Divider Select register	EALLOW	Go
2Ah	CLBCLKCTL	CLB Clocking Control Register	EALLOW	Go
2Ch	LOSPCP	Low Speed Clock Source Prescalar	EALLOW	Go
2Eh	MCDCR	Missing Clock Detect Control Register	EALLOW	Go
30h	X1CNT	10-bit Counter on X1 Clock		Go
32h	XTALCR	XTAL Control Register	EALLOW	Go
36h	ETHERCATCLKCTL	ETHERCATCLKCTL	EALLOW	Go
38h	CMCLKCTL	CMCLKCTL	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-38 shows the codes that are used for access types in this section.

Table 3-38. CLK_CFG_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 3-38. CLK_CFG_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.3.1 CLKSEM Register (Offset = 0h) [Reset = 0h]

CLKSEM is shown in Figure 3-39 and described in Table 3-39.

Return to the Summary Table.

Clock Control Semaphore Register

Figure 3-39. CLKSEM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
R-0/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														SEM	
R-0-0h														R/W-0h	

Table 3-39. CLKSEM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Writing the value 0xa5a5 will allow the writing of the SEM bits, else writes are ignored. Reads will return 0. Note: [1] Due to this KEY, only 32-bit writes will succeed (provided the KEY matches). 16-bit writes to the upper or lower half of this register will be ignored Reset type: CPU1.SYSRSn
15-2	RESERVED	R-0	0h	Reserved
1-0	SEM	R/W	0h	This register provides a mechanism to acquire all the CLKCFG registers (except this register) by CPU1 or CPU2. A CPU can perform read/writes to any of the CLKCFG registers (except this register) only if it owns the semaphore. Otherwise, writes are ignored and reads will return 0x0. Semaphore State Transitions: A value of 00, 10, 11 gives ownership to CPU1 A value of 01 gives ownership to CPU2. The following are the only state transitions allowed on these bits. 00,11 <-> 01 (allowed by CPU2) 00,11 <-> 10 (allowed by CPU1) If a CPU doesn't own the CLK_CFG_REGS set of registers (as defined by the state of this semaphore), reads from that CPU to all those registers return 0x0 and writes are ignore. Note that this is not true of CLKSEM register. The CLKSEM register's reads and writes are always allowed from both CPU1 and CPU2. Reset type: CPU1.SYSRSn

3.16.3.2 CLKCFGLOCK1 Register (Offset = 2h) [Reset = 0h]

CLKCFGLOCK1 is shown in Figure 3-40 and described in Table 3-40.

Return to the Summary Table.

Lock bit for CLKCFG registers
Notes:
[1] Any bit in this register, once set can only be cleared through a CPU1.SYSRSn. Write of 0 to any bit of this register has no effect
[2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed

Figure 3-40. CLKCFGLOCK1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED					CMCLKCTL	ETHERCATCLKCTL	XTALCR
R-0-0h					R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
15	14	13	12	11	10	9	8
LOSPCP	CLBCLKCTL	PERCLKDIVSEL	AUXCLKDIVSEL	SYSCLKDIVSEL	AUXPLLMULT	RESERVED	RESERVED
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R-0-0h	R-0-0h
7	6	5	4	3	2	1	0
AUXPLLCTL1	SYSPLLMULT	SYSPLLCTL3	SYSPLLCTL2	SYSPLLCTL1	CLKSRCCTL3	CLKSRCCTL2	CLKSRCCTL1
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 3-40. CLKCFGLOCK1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R-0	0h	Reserved
18	CMCLKCTL	R/WSonce	0h	Lock bit for CMCLKCTL register: 0: Respective register is not locked 1: Respective register is locked. Reset type: CPU1.SYSRSn
17	ETHERCATCLKCTL	R/WSonce	0h	Lock bit for ETHERCATCLKCTL register: 0: Respective register is not locked 1: Respective register is locked. Reset type: CPU1.SYSRSn
16	XTALCR	R/WSonce	0h	Lock bit for XTALCR register: 0: Respective register is not locked 1: Respective register is locked. Reset type: CPU1.SYSRSn
15	LOSPCP	R/WSonce	0h	Lock bit for LOSPCP register: 0: Respective register is not locked 1: Respective register is locked. Reset type: CPU1.SYSRSn
14	CLBCLKCTL	R/WSonce	0h	Lock bit for CLBCLKCTL register: 0: Respective register is not locked 1: Respective register is locked. Reset type: CPU1.SYSRSn
13	PERCLKDIVSEL	R/WSonce	0h	Lock bit for PERCLKDIVSEL register: 0: Respective register is not locked 1: Respective register is locked. Reset type: CPU1.SYSRSn

Table 3-40. CLKCFGLOCK1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	AUXCLKDIVSEL	R/WSonce	0h	Lock bit for AUXCLKDIVSEL register: 0: Respective register is not locked 1: Respective register is locked. Reset type: CPU1.SYSRSn
11	SYSCLKDIVSEL	R/WSonce	0h	Lock bit for SYSCLKDIVSEL register: 0: Respective register is not locked 1: Respective register is locked. Reset type: CPU1.SYSRSn
10	AUXPLLMULT	R/WSonce	0h	Lock bit for AUXPLLMULT register: 0: Respective register is not locked 1: Respective register is locked. Reset type: CPU1.SYSRSn
9	RESERVED	R-0	0h	Reserved
8	RESERVED	R-0	0h	Reserved
7	AUXPLLCTL1	R/WSonce	0h	Lock bit for AUXPLLCTL1 register: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any bit in this register, once set can only be cleared through a CPU1.SYSRSn. Write of 0 to any bit of this register has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: CPU1.SYSRSn
6	SYSPLLMULT	R/WSonce	0h	Lock bit for SYSPLLMULT register: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any bit in this register, once set can only be cleared through a CPU1.SYSRSn. Write of 0 to any bit of this register has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: CPU1.SYSRSn
5	SYSPLLCTL3	R/WSonce	0h	Lock bit for SYSPLLCTL3 register: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any bit in this register, once set can only be cleared through a CPU1.SYSRSn. Write of 0 to any bit of this register has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: CPU1.SYSRSn
4	SYSPLLCTL2	R/WSonce	0h	Lock bit for SYSPLLCTL2 register: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any bit in this register, once set can only be cleared through a CPU1.SYSRSn. Write of 0 to any bit of this register has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: CPU1.SYSRSn
3	SYSPLLCTL1	R/WSonce	0h	Lock bit for SYSPLLCTL1 register: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any bit in this register, once set can only be cleared through a CPU1.SYSRSn. Write of 0 to any bit of this register has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: CPU1.SYSRSn

Table 3-40. CLKCFGLOCK1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	CLKSRCCTL3	R/WSonce	0h	Lock bit for CLKSRCCTL3 register: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any bit in this register, once set can only be cleared through a CPU1.SYSRSn. Write of 0 to any bit of this register has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: CPU1.SYSRSn
1	CLKSRCCTL2	R/WSonce	0h	Lock bit for CLKSRCCTL2 register: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any bit in this register, once set can only be cleared through a CPU1.SYSRSn. Write of 0 to any bit of this register has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: CPU1.SYSRSn
0	CLKSRCCTL1	R/WSonce	0h	Lock bit for CLKSRCCTL1 register: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any bit in this register, once set can only be cleared through a CPU1.SYSRSn. Write of 0 to any bit of this register has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: CPU1.SYSRSn

3.16.3.3 CLKSRCCTL1 Register (Offset = 8h) [Reset = 0h]

CLKSRCCTL1 is shown in Figure 3-41 and described in Table 3-41.

Return to the Summary Table.

Clock Source Control register-1
This memory mapped register requires a delay of 69 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 69 NOP instructions.

Figure 3-41. CLKSRCCTL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		RESERVED	XTALOFF	INTOSC2OFF_NOTSUPPORTED	RESERVED	OSCCLKSRCSEL	
R-0-0h		R/W-0h	R/W-0h	R/W-0h	R-0-0h	R/W-0h	

Table 3-41. CLKSRCCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-6	RESERVED	R-0	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	XTALOFF	R/W	0h	Crystal (External) Oscillator Off Bit: This bit turns external oscillator off: 0 = Crystal (External) Oscillator On (default on reset) 1 = Crystal (External) Oscillator Off NOTE: Ensure no resources are using a clock source prior to disabling it. For example OSCCLKSRCSEL (SYSPLL), AUXOSCCLKSRCSEL (AUXPLL), CANxBCLKSEL (CAN Clock), TMR2CLKSRCSEL (CPUTIMER2) and XCLKOUTSEL(XCLKOUT). Reset type: XRSn
3	INTOSC2OFF_NOTSUPPORTED	R/W	0h	RESERVED: This bit is not supported any more, and should not be set to 1. Note: If this bit is set to 1 it will turn OFF INTOSC2 and lead to PLL failure Reset type: XRSn
2	RESERVED	R-0	0h	Reserved

Table 3-41. CLKSRCCTL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	OSCCLKSRCSEL	R/W	0h	Oscillator Clock Source Select Bit: This bit selects the source for OSCCLK. 00 = INTOSC2 (default on reset) 01 = External Oscillator (XTAL) 10 = INTOSC1 11 = reserved (default to INTOSC1) At power-up or after an XRSn, INTOSC2 is selected by default. Whenever the user changes the clock source using these bits, the SYSPLLMULT register will be forced to zero and the PLL will be bypassed and powered down. This prevents potential PLL overshoot. The user will then have to write to the SYSPLLMULT register to configure the appropriate multiplier. Notes: [1] Reserved selection defaults to 00 configuration [2] INTOSC1 is recommended to be used only after missing clock detection. If user wants to re-lock the PLL with INTOSC1 (the back-up clock source) after missing clock is detected, he can do a MCLKCLR and lock the PLL. [3] Any writes to this bit must be followed with at least 300 CPU Cycles of wait time by adding at least 300 NOP instructions. [4] Changing the OSCCLKSRC while PLL is running and used by system (i.e. PLLCLKEN=1), can lead to dead System Clock. User needs to first bypass the PLL clock from the system by PLLCLKEN=0, and then change the OSCCLK source. Reset type: XRSn

3.16.3.4 CLKSRCCTL2 Register (Offset = Ah) [Reset = 0h]

CLKSRCCTL2 is shown in Figure 3-42 and described in Table 3-42.

Return to the Summary Table.

Clock Source Control register-2
This memory mapped register requires a delay of 69 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 69 NOP instructions.

Figure 3-42. CLKSRCCTL2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED				MCANABITCLKSEL		RESERVED	
R-0-0h				R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		CANBBCLKSEL		CANABCLKSEL		AUXOSCCLKSRCSEL	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 3-42. CLKSRCCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-12	RESERVED	R-0	0h	Reserved
11-10	MCANABITCLKSEL	R/W	0h	MCAN Bit Clock Source Select Bit: 00 = CM.Perx.SYSCLK or CPU1.PERx.SYSCLK based on PALLOCATE.MCAN_A setting. 01 = AUXPLLRAWCLK 10 = AUXCLKIN 11 = Rsvd If bit timing clock source has to change, then PCLKCR bit for the corresponding MCAN instance has to be cleared to '0' first before updating the values in thies field. Once the value is updated, the corresponding PCLKCR bit of this instance can be set back to '1'. Reset type: XRSn
9-8	RESERVED	R/W	0h	Reserved
7-6	RESERVED	R/W	0h	Reserved
5-4	CANBBCLKSEL	R/W	0h	CANB Bit Clock Source Select Bit: 00 = CPUx.PERx.SYSCLK (default on reset), if PALLOCATE0.CANB is 0. Else CM.Perx.SYSCLK. 01 = External Oscillator (XTAL) 10 = AUXCLKIN (from GPIO) 11 = Reserved Missing clock detect circuit doesnt have any impact on these bits. If bit timing clock source has to change, then PCLKCR bit for the corresponding DCAN instance has to be cleared to '0' first before updating the values in thies field. Once the value is updated, the corresponding PCLKCR bit of this instance can be set back to '1'. Reset type: XRSn

Table 3-42. CLKSRCCTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	CANABCLKSEL	R/W	0h	CANA Bit Clock Source Select Bit: 00 = PERx.SYSCLK (default on reset),if PALLOCATE0.CANA is 0 else CMCLK. 01 = External Oscillator (XTAL) 10 = AUXCLKIN (from GPIO) 11 = Reserved Missing clock detect circuit doesnt have any impact on these bits. If bit timing clock source has to change, then PCLKCR bit for the corresponding DCAN instance has to be cleared to '0' first before updating the values in thies field. Once the value is updated, the corresponding PCLKCR bit of this instance can be set back to '1'. Reset type: XRSn
1-0	AUXOSCCLKSRCSEL	R/W	0h	Oscillator Clock Source Select Bit: This bit selects the source for AUXOSCCLK: 00 = INTOSC2 (default on reset) 01 = External Oscillator (XTAL) 10 = AUXCLKIN (from GPIO) 11 = Reserved(default to INTOSC2) Whenever the user changes the clock source using these bits, the AUXPLLMULT register will be forced to zero. The user will then have to write to the AUXPLLMULT register to configure the appropriate multiplier. The user must wait 10 OSCCLK cycles before writing to AUXPLLMULT or disabling the previous clock source to allow the change to complete. The missing clock detection circuit does not affect these bits. Notes: [1] Changing the AUXOSCCLKSRC while PLL is running and AUXPLLCLKEN=1, can lead to dead AUXPLLCLK. User needs to first bypass the AUXPLL by AUXPLLCLKEN=0, and then change the AUXOSCCLK source. Reset type: XRSn

3.16.3.5 CLKSRCCTL3 Register (Offset = Ch) [Reset = 0h]

CLKSRCCTL3 is shown in Figure 3-43 and described in Table 3-43.

Return to the Summary Table.

Clock Source Control register-3
This memory mapped register requires a delay of 69 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 69 NOP instructions.

Figure 3-43. CLKSRCCTL3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												XCLKOUTSEL			
R-0-0h												R/W-0h			

Table 3-43. CLKSRCCTL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-4	RESERVED	R-0	0h	Reserved
3-0	XCLKOUTSEL	R/W	0h	XCLKOUT Source Select Bit: This bit selects the source for XCLKOUT: 0000 = PLLSYSCLK (default on reset) 0001 = SYSPLLCLK 0010 = CPU1.SYSCLK 0011 = CPU2.SYSCLK 0100 = AUXPLLCLK 0101 = INTOSC1 0110 = INTOSC2 0111 = XTAL OSC o/p clock 1000 = CMCLK 1100 = PLLRAWCLK 1101 = AUXPLLRAWCLK others = Reserved Reset type: CPU1.SYSRSn

3.16.3.6 SYSPLLCTL1 Register (Offset = Eh) [Reset = 0h]

SYSPLLCTL1 is shown in Figure 3-44 and described in Table 3-44.

Return to the Summary Table.

SYSPLL Control register-1

Figure 3-44. SYSPLLCTL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	RESERVED	RESERVED	PLLCLKEN	PLLEN
R-0-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-44. SYSPLLCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-5	RESERVED	R-0	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	PLLCLKEN	R/W	0h	SYSPLL bypassed or included in the PLLSYSCLK path: This bit decides if the SYSPLL is bypassed when PLLSYSCLK is generated 1 = PLLSYSCLK is fed from the SYSPLL clock output. Users need to make sure that the PLL is locked before enabling this clock to the system. 0 = SYSPLL is bypassed. Clock to system is direct feed from OSCCLK Note: Any writes to this bit must be followed with at least 120 CPU Cycles of wait time by adding at least 120 NOP instructions. Reset type: XRSn
0	PLLEN	R/W	0h	SYSPLL enabled or disabled: This bit decides if the SYSPLL is enabled or not 1 = SYSPLL is enabled 0 = SYSPLL is powered off. Clock to system is direct feed from OSCCLK Note: Any writes to this bit must be followed with at least 60 CPU Cycles of wait time by adding at least 60 NOP instructions. Reset type: XRSn

3.16.3.7 SYSPLLMULT Register (Offset = 14h) [Reset = 0h]

SYSPLLMULT is shown in Figure 3-45 and described in Table 3-45.

Return to the Summary Table.

SYSPLL Multiplier register
NOTE:
IMULT and REFDIV fields in this register must be written at the same time and ONLY when SYSPLLCTL1.PLLCLKEN=0 for correct PLL operation. If IMULT or REFDIV values are changed after SYSPLLCTL1.PLLCLKEN=1 then it will disrupt PLL operation and cause system hangup.
This memory mapped register requires a delay of 69 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 69 NOP instructions.

Figure 3-45. SYSPLLMULT Register

31	30	29	28	27	26	25	24
RESERVED			REFDIV				
R-0-0h			R/W-0h				
23	22	21	20	19	18	17	16
RESERVED			ODIV				
R-0-0h			R/W-0h				
15	14	13	12	11	10	9	8
RESERVED		RESERVED		RESERVED		RESERVED	
R-0-0h		R/W-0h		R-0-0h		R/W-0h	
7	6	5	4	3	2	1	0
IMULT							
R/W-0h							

Table 3-45. SYSPLLMULT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R-0	0h	Reserved
28-24	REFDIV	R/W	0h	SYSPLL Reference Clock Divider PLL Reference Divider = REFDIV + 1 NOTE: IMULT and REFDIV fields in this register must be written at the same time and ONLY when SYSPLLCTL1.PLLCLKEN=0 for correct PLL operation. If IMULT or REFDIV values are changed after SYSPLLCTL1.PLLCLKEN=1 then it will disrupt PLL operation and cause system hangup. Reset type: XRSn
23-21	RESERVED	R-0	0h	Reserved
20-16	ODIV	R/W	0h	SYSPLL Output Clock Divider PLL Output Divider = ODIV + 1 ODIV should be set to at least 1 to ensure the PLL output meets system duty cycle requirements. NOTE: If PLL is powered when SYSPLLCTL1.PLLCLKEN=0, then it is recommended to write IMULT, REFDIV and ODIV at the same time. This field can ALSO be programmed after SYSPLLCTL1.PLLCLKEN=1 if application desires to change the output divider of PLL clock, but proper care must be taken to make sure values of IMULT and REFDIV are left unchanged when SYSPLLCTL1.PLLCLKEN=1, if values of IMULT or REFDIV are change after SYSPLLCTL1.PLLCLKEN=1 then it will disrupt PLL operation and cause system hangup. Reset type: XRSn
15-14	RESERVED	R-0	0h	Reserved

Table 3-45. SYSPLLMULT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13-12	RESERVED	R/W	0h	Reserved
11-10	RESERVED	R-0	0h	Reserved
9-8	RESERVED	R/W	0h	Reserved
7-0	IMULT	R/W	0h	SYSPLL Integer Multiplier: For 0000000 Fout = Fref (PLLBYPASS) Integer Multiplier = 1 0000001 Integer Multiplier = 1 0000010 Integer Multiplier = 2 0000011 Integer Multiplier = 3 ..... 1111111 Integer Multipler = 127 NOTE: IMULT and REFDIV fields in this register must be written at the same time and ONLY when SYSPLLCTL1.PLLCLKEN=0 for correct PLL operation. If IMULT or REFDIV values are changed after SYSPLLCTL1.PLLCLKEN=1 then it will disrupt PLL opeartion and cause system hangup. Reset type: XRSn

3.16.3.8 SYSPLLSTS Register (Offset = 16h) [Reset = 0h]

SYSPLLSTS is shown in Figure 3-46 and described in Table 3-46.

Return to the Summary Table.

SYSPLL Status register

Figure 3-46. SYSPLLSTS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	SLIPS_NOTSUPPORTED	LOCKS
R-0-0h				R-0h	R-0h	R-0h	R-0h

Table 3-46. SYSPLLSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-4	RESERVED	R-0	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	SLIPS_NOTSUPPORTED	R	0h	RESERVED: This bit is reserved and the value read should be ignored. TI recommends using DCC to evaluate SYSPLL Slip status. Refer to InitSysPll() or SysCtl_setClock() functions inside the latest example software from C2000Ware for checking SYSPLL Slip status using DCC. Reset type: XRSn
0	LOCKS	R	0h	SYSPLL Lock Status Bit: This bit indicates whether the SYSPLL is locked or not 0 = SYSPLL is not yet locked 1 = SYSPLL is locked Reset type: XRSn

3.16.3.9 AUXPLLCTL1 Register (Offset = 18h) [Reset = 0h]

AUXPLLCTL1 is shown in Figure 3-47 and described in Table 3-47.

Return to the Summary Table.

AUXPLL Control register-1

Figure 3-47. AUXPLLCTL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	RESERVED	RESERVED	PLLCLKEN	PLLEN
R-0-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-47. AUXPLLCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-5	RESERVED	R-0	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	PLLCLKEN	R/W	0h	AUXPLL bypassed or included in the AUXPLLCLK path: This bit decides if the AUXPLL is bypassed when AUXPLLCLK is generated 1 = AUXPLLCLK is fed from the AUXPLL clock output. Users need to make sure that the PLL is locked before enabling this clock to the AUXPLLCLK connected modules. 0 = AUXPLL is bypassed. Clock to modules connected to AUXPLLCLK is direct feed from AUXOSCCLK Reset type: XRSn
0	PLLEN	R/W	0h	AUXPLL enabled or disabled: This bit decides if the AUXPLL is enabled or not 1 = AUXPLL is enabled 0 = AUXPLL is powered off. Clock to system is direct feed from AUXOSCCLK Reset type: XRSn

3.16.3.10 AUXPLLMULT Register (Offset = 1Eh) [Reset = 0h]

AUXPLLMULT is shown in Figure 3-48 and described in Table 3-48.

Return to the Summary Table.

AUXPLL Multiplier register
NOTE:
IMULT and REFDIV fields in this register must be written at the same time and ONLY when AUXPLLCTL1.PLLCLKEN=0 for correct PLL operation. If IMULT or REFDIV values are changed after AUXPLLCTL1.PLLCLKEN=1 then it will disrupt PLL operation.
This memory mapped register requires a delay of 69 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 69 NOP instructions.

Figure 3-48. AUXPLLMULT Register

31	30	29	28	27	26	25	24
RESERVED			REFDIV				
R-0-0h			R/W-0h				
23	22	21	20	19	18	17	16
RESERVED			ODIV				
R-0-0h			R/W-0h				
15	14	13	12	11	10	9	8
RESERVED		RESERVED		RESERVED		RESERVED	
R-0-0h		R/W-0h		R-0-0h		R/W-0h	
7	6	5	4	3	2	1	0
IMULT							
R/W-0h							

Table 3-48. AUXPLLMULT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R-0	0h	Reserved
28-24	REFDIV	R/W	0h	AUXPLL Reference Clock Divider PLL Reference Divider = REFDIV + 1 NOTE: IMULT and REFDIV fields in this register must be written at the same time and ONLY when AUXPLLCTL1.PLLCLKEN=0 for correct PLL operation. If IMULT or REFDIV values are changed after AUXPLLCTL1.PLLCLKEN=1 then it will disrupt PLL operation. Reset type: XRSn
23-21	RESERVED	R-0	0h	Reserved
20-16	ODIV	R/W	0h	AUXPLL Output Clock Divider PLL Output Divider = ODIV + 1 ODIV should be set to at least 1 to ensure the PLL output meets system duty cycle requirements. NOTE: If PLL is powered when AUXPLLCTL1.PLLCLKEN=0, then it is recommended to write IMULT, REFDIV and ODIV at the same time. This field can ALSO be programmed after AUXPLLCTL1.PLLCLKEN=1 if application desires to change the output divider of PLL clock, but proper care must be taken to make sure values of IMULT and REFDIV are left unchanged when AUXPLLCTL1.PLLCLKEN=1, if values of IMULT or REFDIV are change after AUXPLLCTL1.PLLCLKEN=1 then it will disrupt PLL operation. Reset type: XRSn
15-14	RESERVED	R-0	0h	Reserved
13-12	RESERVED	R/W	0h	Reserved

Table 3-48. AUXPLLMULT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11-10	RESERVED	R-0	0h	Reserved
9-8	RESERVED	R/W	0h	Reserved
7-0	IMULT	R/W	0h	AUXPLL Integer Multiplier: For 0000000 Fout = Fref (PLLBYPASS) Integer Multiplier = 1 0000001 Integer Multiplier = 1 0000010 Integer Multiplier = 2 0000011 Integer Multiplier = 3 ..... 1111111 Integer Multipler = 127 NOTE: IMULT and REFDIV fields in this register must be written at the same time and ONLY when AUXPLLCTL1.PLLCLKEN=0 for correct PLL operation. If IMULT or REFDIV values are changed after AUXPLLCTL1.PLLCLKEN=1 then it will disrupt PLL operation. Reset type: XRSn

3.16.3.11 AUXPLLSTS Register (Offset = 20h) [Reset = 0h]

AUXPLLSTS is shown in Figure 3-49 and described in Table 3-49.

Return to the Summary Table.

AUXPLL Status register

Figure 3-49. AUXPLLSTS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	SLIPS_NOTSUPPORTED	LOCKS
R-0-0h				R-0h	R-0h	R-0h	R-0h

Table 3-49. AUXPLLSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-4	RESERVED	R-0	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	SLIPS_NOTSUPPORTED	R	0h	RESERVED: This bit is reserved and the value read should be ignored. TI recommends using DCC to evaluate AUXPLL Slip status. Refer to InitAuxPll() or SysCtl_setAuxClock() functions inside the latest example software from C2000Ware for checking AUXPLL Slip status using DCC. Reset type: XRSn
0	LOCKS	R	0h	AUXPLL Lock Status Bit: This bit indicates whether the AUXPLL is locked or not 0 = AUXPLL is not yet locked 1 = AUXPLL is locked Reset type: XRSn

3.16.3.12 SYSCLKDIVSEL Register (Offset = 22h) [Reset = 0h]

SYSCLKDIVSEL is shown in Figure 3-50 and described in Table 3-50.

Return to the Summary Table.

System Clock Divider Select register
This memory mapped register requires a delay of 69 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 69 NOP instructions.

Figure 3-50. SYSCLKDIVSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED										PLLSYSCLKDIV					
R-0-0h										R/W-0h					

Table 3-50. SYSCLKDIVSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-6	RESERVED	R-0	0h	Reserved
5-0	PLLSYSCLKDIV	R/W	0h	PLLSYSCLK Divide Select: This bit selects the divider setting for the PLLSYSCLK. 0000 = /1 0001 = /2 0010 = /4 0011 = /6 0100 = /8 0101 = /10 0110 = /12 0111 = /14 1000 = /16 Others: Reserved Reset type: XRSn

3.16.3.13 AUXCLKDIVSEL Register (Offset = 24h) [Reset = 1301h]

AUXCLKDIVSEL is shown in Figure 3-51 and described in Table 3-51.

Return to the Summary Table.

Auxillary Clock Divider Select register
This memory mapped register requires a delay of 69 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 69 NOP instructions.

Figure 3-51. AUXCLKDIVSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			MCANCLKDIV					RESERVED					AUXPLLDIV		
R-0-0h			R/W-13h					R-0-0h					R/W-1h		

Table 3-51. AUXCLKDIVSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R-0	0h	Reserved
12-8	MCANCLKDIV	R/W	13h	This bit-field divides the source clock (chosen by the CLKSRCCTL2.MCANABITCLKSEL bit-field) before feeding it as the bit-clock to the MCAN module 00000 = /1 00001 = /2 ... 10010 = /19 10011 = /20 101xx = Rsvd 11xxx = Rsvd Reset type: XRSn
7-3	RESERVED	R-0	0h	Reserved
2-0	AUXPLLDIV	R/W	1h	AUXPLLCLK Divide Select: This bit selects the divider setting for the AUXPLLCK. 000 = /1 001 = /2 (default on reset) 010 = /4 011 = /8 100 = /3 101 = /5 110 = /6 111 = /7 Reset type: XRSn

3.16.3.14 PERCLKDIVSEL Register (Offset = 26h) [Reset = 51h]

PERCLKDIVSEL is shown in Figure 3-52 and described in Table 3-52.

Return to the Summary Table.

Peripheral Clock Divider Selet register
This memory mapped register requires a delay of 69 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 69 NOP instructions.

Figure 3-52. PERCLKDIVSEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED	EMIF2CLKDIV	RESERVED	EMIF1CLKDIV	RESERVED		EPWMCLKDIV	
R-0-0h	R/W-1h	R-0-0h	R/W-1h	R/W-0h		R/W-1h	

Table 3-52. PERCLKDIVSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-7	RESERVED	R-0	0h	Reserved
6	EMIF2CLKDIV	R/W	1h	EMIF2 Clock Divide Select: This bit selects whether the EMIF2 module run with a /1 or /2 clock. 0: /1 of CPU1.SYSCLK is selected 1: /2 of CPU1.SYSCLK is selected Reset type: CPU1.SYSRSn
5	RESERVED	R-0	0h	Reserved
4	EMIF1CLKDIV	R/W	1h	EMIF1 Clock Divide Select: This bit selects whether the EMIF1 module run with a /1 or /2 clock. For single core device 0: /1 of CPU1.SYSCLK is selected 1: /2 of CPU1.SYSCLK is selected For Dual core device 0: /1 of PLLSYSCLK is selected 1: /2 of PLLSYSCLK is selected Reset type: CPU1.SYSRSn
3-2	RESERVED	R/W	0h	Reserved
1-0	EPWMCLKDIV	R/W	1h	EPWM Clock Divide Select: This bit selects whether the EPWM modules run with a /1 or /2 clock. This divider sits in front of the PLLSYSCLK x0 = /1 of PLLSYSCLK x1 = /2 of PLLSYSLCK (default on reset) Note: Refer to the EPWM User Guide for maximum EPWM Frequency Reset type: CPU1.SYSRSn

3.16.3.15 XCLKOUTDIVSEL Register (Offset = 28h) [Reset = 3h]

XCLKOUTDIVSEL is shown in Figure 3-53 and described in Table 3-53.

Return to the Summary Table.

XCLKOUT Divider Select register
This memory mapped register requires a delay of 69 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 69 NOP instructions.

Figure 3-53. XCLKOUTDIVSEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						XCLKOUTDIV	
R-0-0h						R/W-3h	

Table 3-53. XCLKOUTDIVSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-2	RESERVED	R-0	0h	Reserved
1-0	XCLKOUTDIV	R/W	3h	XCLKOUT Divide Select: This bit selects the divider setting for the XCLKOUT. 00 = /1 01 = /2 10 = /4 11 = /8 (default on reset) Reset type: CPU1.SYSRSn

3.16.3.16 CLBCLKCTL Register (Offset = 2Ah) [Reset = 7h]

CLBCLKCTL is shown in Figure 3-54 and described in Table 3-54.

Return to the Summary Table.

CLB Clocking Control Register
This memory mapped register requires a delay of 69 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 69 NOP instructions.

Figure 3-54. CLBCLKCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
CLKMODECLB8	CLKMODECLB7	CLKMODECLB6	CLKMODECLB5	CLKMODECLB4	CLKMODECLB3	CLKMODECLB2	CLKMODECLB1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED			TILECLKDIV	RESERVED	CLBCLKDIV		
R-0-0h			R/W-0h	R-0-0h	R/W-7h		

Table 3-54. CLBCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R-0	0h	Reserved
23	CLKMODECLB8	R/W	0h	0 : CLB8 is synchronous to SYSCLK 1 : CLB8 runs of asynchronous clock Reset type: SYSRSn
22	CLKMODECLB7	R/W	0h	0 : CLB7 is synchronous to SYSCLK 1 : CLB7 runs of asynchronous clock Reset type: SYSRSn
21	CLKMODECLB6	R/W	0h	0 : CLB6 is synchronous to SYSCLK 1 : CLB6 runs of asynchronous clock Reset type: SYSRSn
20	CLKMODECLB5	R/W	0h	0 : CLB5 is synchronous to SYSCLK 1 : CLB5 runs of asynchronous clock Reset type: SYSRSn
19	CLKMODECLB4	R/W	0h	0 : CLB4 is synchronous to SYSCLK 1 : CLB4 runs of asynchronous clock Reset type: SYSRSn
18	CLKMODECLB3	R/W	0h	0 : CLB3 is synchronous to SYSCLK 1 : CLB3 runs of asynchronous clock Reset type: SYSRSn
17	CLKMODECLB2	R/W	0h	0 : CLB2 is synchronous to SYSCLK 1 : CLB2 runs of asynchronous clock Reset type: SYSRSn
16	CLKMODECLB1	R/W	0h	0 : CLB1 is synchronous to SYSCLK 1 : CLB1 runs of asynchronous clock Reset type: SYSRSn
15-5	RESERVED	R-0	0h	Reserved
4	TILECLKDIV	R/W	0h	0: /1 1: /2 Reset type: SYSRSn

Table 3-54. CLBCLKCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	RESERVED	R-0	0h	Reserved
2-0	CLBCLKDIV	R/W	7h	000: /1 001: /2 010: /3 011: /4 100: /5 101: /6 110: /7 111: /8 Reset type: SYSRSn

3.16.3.17 LOSPCP Register (Offset = 2Ch) [Reset = 2h]

LOSPCP is shown in Figure 3-55 and described in Table 3-55.

Return to the Summary Table.

Low Speed Clock Source Prescalar

Figure 3-55. LOSPCP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													LSPCLKDIV		
R-0-0h													R/W-2h		

Table 3-55. LOSPCP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-3	RESERVED	R-0	0h	Reserved
2-0	LSPCLKDIV	R/W	2h	These bits configure the low-speed peripheral clock (LSPCLK) rate relative to SYSCLK of CPU1 and CPU2. 000,LSPCLK = / 1 001,LSPCLK = / 2 010,LSPCLK = / 4 (default on reset) 011,LSPCLK = / 6 100,LSPCLK = / 8 101,LSPCLK = / 10 110,LSPCLK = / 12 111,LSPCLK = / 14 Note: [1] This clock is used as strobe for the SCI and SPI modules. Reset type: CPU1.SYSRSn

3.16.3.18 MCDCR Register (Offset = 2Eh) [Reset = 0h]

MCDCR is shown in Figure 3-56 and described in Table 3-56.

Return to the Summary Table.

Missing Clock Detect Control Register

Figure 3-56. MCDCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				OSCOFF	MCLKOFF	MCLKCLR	MCLKSTS
R-0-0h				R/W-0h	R/W-0h	R-0/W1S-0h	R-0h

Table 3-56. MCDCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-4	RESERVED	R-0	0h	Reserved
3	OSCOFF	R/W	0h	Oscillator Clock Disconnect from MCD Bit: 0 = OSCCLK Connected to OSCCLK Counter in MCD module 1 = OSCCLK Disconnected to OSCCLK Counter in MCD module Reset type: XRSn
2	MCLKOFF	R/W	0h	Missing Clock Detect Off Bit: 0 = Missing Clock Detect Circuit Enabled 1 = Missing Clock Detect Circuit Disabled Reset type: XRSn
1	MCLKCLR	R-0/W1S	0h	Missing Clock Clear Bit: Write 1" to this bit to clear MCLKSTS bit and reset the missing clock detect circuit." Reset type: XRSn
0	MCLKSTS	R	0h	Missing Clock Status Bit: 0 = OSCCLK Is OK 1 = OSCCLK Detected Missing, CLOCKFAILn Generated Reset type: XRSn

3.16.3.19 X1CNT Register (Offset = 30h) [Reset = 0h]

X1CNT is shown in Figure 3-57 and described in Table 3-57.

Return to the Summary Table.

10-bit Counter on X1 Clock

Figure 3-57. X1CNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															CLR
R-0-0h															R/W-0h
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						X1CNT									
R-0-0h						R-0h									

Table 3-57. X1CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R-0	0h	Reserved
16	CLR	R/W	0h	X1 Counter clear: 1: X1CNT clear is asserted 0: X1CNT clear is de-asserted Reset type: XRSn
15-10	RESERVED	R-0	0h	Reserved
9-0	X1CNT	R	0h	X1 Counter: - This counter increments on every X1 CLOCKs positive-edge. - Once it reaches the values of 0x3ff, it freezes - Before switching from INTOSC2 to X1, application must check this counter and make sure that it has saturated. This will ensure that the Crystal connected to X1/X2 is oscillating. Note: Before switching the OSCCCLKSRC to X1, X1CNT register needs to be read at least 3 times for 0x3FF value. Refer C2000Ware function "SysCtl_pollX1Counter" for SW implementation. Reset type: XRSn

3.16.3.20 XTALCR Register (Offset = 32h) [Reset = 4h]

XTALCR is shown in Figure 3-58 and described in Table 3-58.

Return to the Summary Table.

XTAL Control Register
This memory mapped register requires a delay of 69 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 69 NOP instructions.

Figure 3-58. XTALCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED					RESERVED	SE	OSCOFF
R-0-0h					R/W-1h	R/W-0h	R/W-0h

Table 3-58. XTALCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R-0	0h	Reserved
2	RESERVED	R/W	1h	Reserved
1	SE	R/W	0h	Configures XTAL oscillator in single-ended or Crystal mode when XTAL oscillator is powered (OFF = 0) 0 XTAL oscillator in Crystal mode 1 XTAL oscilator in single0ended mode (through X1) Reset type: XRSn
0	OSCOFF	R/W	0h	This bit if '1', powers-down the XTAL oscillator macro and hence doesn't let X2 to be driven by the XTAL oscillator. If a crystal is connected to X1/X2, user needs to first clear this bit, wait for the oscillator to power up (using X1CNT) and then only switch the clock source to X1/X2 Reset type: XRSn

3.16.3.21 ETHERCATCLKCTL Register (Offset = 36h) [Reset = Eh]

ETHERCATCLKCTL is shown in Figure 3-59 and described in Table 3-59.

Return to the Summary Table.

ETHERCAT clock control register.
This memory mapped register requires a delay of 69 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 69 NOP instructions.

Figure 3-59. ETHERCATCLKCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							PHYCLKEN
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED				ECATDIV			DIVSRCSEL
R-0-0h				R/W-7h			R/W-0h

Table 3-59. ETHERCATCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	PHYCLKEN	R/W	0h	0 : etherCAT phy clock disabled 1 : etherCAT phy clock enabled Reset type: XRSn
7-4	RESERVED	R-0	0h	Reserved
3-1	ECATDIV	R/W	7h	000: /1 001: /2 010: /3 011: /4 100: /5 101: /6 110: /7 111: /8 Reset type: XRSn
0	DIVSRCSEL	R/W	0h	0: Auxillary PLL is the source for the etherCAT clock divider. 1: System PLL is the source for etherCAT clock divider. Reset type: XRSn

3.16.3.22 CMCLKCTL Register (Offset = 38h) [Reset = E6h]

CMCLKCTL is shown in Figure 3-60 and described in Table 3-60.

Return to the Summary Table.

CM Clock control register
This memory mapped register requires a delay of 69 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 69 NOP instructions.

Figure 3-60. CMCLKCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
ETHDIV			ETHDIVSRCSEL	CMCLKDIV			CMDIVSRCSEL
R/W-7h			R/W-0h	R/W-3h			R/W-0h

Table 3-60. CMCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-5	ETHDIV	R/W	7h	000: /1 001: /2 010: /3 011: /4 100: /5 101: /6 110: /7 111: /8 Reset type: XRSn
4	ETHDIVSRCSEL	R/W	0h	0: Auxillary PLL is the source for the etherNET clock divider. 1: System PLL is the source for etherNET clock divider. Reset type: XRSn
3-1	CMCLKDIV	R/W	3h	000: /1 001: /2 010: /3 011: /4 100: /5 101: /6 110: /7 111: /8 Note: CMCLKDIV should be configured prior or along with CMCLKDIV configuration. If CMCLKDIV is configured after CMDIVSRCSEL in the next cycle, the writes to this field gets ignored. Reset type: XRSn
0	CMDIVSRCSEL	R/W	0h	0: Auxillary PLL is the source for the CM clock divider. 1: System PLL is the source for CM clock divider. Reset type: XRSn

3.16.4 CM_CONF_REGS Registers

Table 3-61 lists the memory-mapped registers for the CM_CONF_REGS registers. All register offset addresses not listed in Table 3-61 should be considered as reserved locations and the register contents should not be modified.

Table 3-61. CM_CONF_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CMRESCTL	CM Reset Control Register	EALLOW	Go
2h	CMTOCPU1NMICTL	CM To CPU1 NMI Control register	EALLOW	Go
4h	CMTOCPU1INTCTL	CM To CPU1 interrupt Control register	EALLOW	Go
20h	PALLOCATE0	CM Peripheral Allocation Register.	EALLOW	Go
3FEh	CM_CONF_REGS_LOCK	CM Configuration Registers Lock	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-62 shows the codes that are used for access types in this section.

Table 3-62. CM_CONF_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.4.1 CMRESCTL Register (Offset = 0h) [Reset = 1h]

CMRESCTL is shown in Figure 3-61 and described in Table 3-63.

Return to the Summary Table.

Software reset of CM.

Figure 3-61. CMRESCTL Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESETSTS	RESET
R-0h						R-0h	R/W-1h

Table 3-63. CMRESCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to this register succeeds only if this field is written with a value of 0xa5a5 Note: [1] Due to this KEY, only 32-bit writes will succeed (provided the KEY matches). 16-bit writes to the upper or lower half of this register will be ignored Reset type: CPU1.SYSRSn
15-2	RESERVED	R	0h	Reserved
1	RESETSTS	R	0h	0: CM is under reset 1: CM is out of reset. Reset type: CPU1.SYSRSn
0	RESET	R/W	1h	1: Asserts reset to CM. 0: De-asserts reset to CM. Software Note: This bit should be kept high until RESETSTS bit of CMRSTCTL register goes low. Reset type: CPU1.SYSRSn

3.16.4.2 CMTOCPU1NMICTL Register (Offset = 2h) [Reset = 0h]

CMTOCPU1NMICTL is shown in Figure 3-62 and described in Table 3-64.

Return to the Summary Table.

CM To CPU1 NMI Control register

Figure 3-62. CMTOCPU1NMICTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					CMNMIWDRST	RESERVED	RESERVED
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 3-64. CMTOCPU1NMICTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2	CMNMIWDRST	R/W	0h	0: NMI to CPU1 is not fired on a CMNMIWDRST to CM4. 1: NMI to CPU1 is fired on a CMNMIWDRST to CM4. Reset type: XRSn
1	RESERVED	R/W	0h	Reserved
0	RESERVED	R/W	0h	Reserved

3.16.4.3 CMTOCPU1INTCTL Register (Offset = 4h) [Reset = 0h]

CMTOCPU1INTCTL is shown in Figure 3-63 and described in Table 3-65.

Return to the Summary Table.

CM To CPU1 interrupt Control register

Figure 3-63. CMTOCPU1INTCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					CMNMIWDRST	SYSRESETREQ	VECTRESET
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 3-65. CMTOCPU1INTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2	CMNMIWDRST	R/W	0h	0: Interrupt to CPU1 is not fired on a CMNMIWDRST to CM4. 1: Interrupt to CPU1 is fired on a CMNMIWDRST to CM4. Reset type: XRSn
1	SYSRESETREQ	R/W	0h	0: Interrupt to CPU1 is not fired on a SYSRESETREQ to CM4. 1: Interrupt to CPU1 is fired on a SYSRESETREQ to CM4. Reset type: XRSn
0	VECTRESET	R/W	0h	0: Interrupt to CPU1 is not fired on a VECTRESET to CM4. 1: Interrupt to CPU1 is fired on a VECTRESET to CM4. Reset type: XRSn

3.16.4.4 PALLOCATE0 Register (Offset = 20h) [Reset = 10h]

PALLOCATE0 is shown in Figure 3-64 and described in Table 3-66.

Return to the Summary Table.

CM Peripheral Allocation Register for shared peripherals.
This register must be configured prior to enabling the peripheral clocks.
The clock for each peripheral is derived from the selected CPU subsystem. The clock mux controlled by this register is not glitch-free, therefore the PALLOCATEx register must be configured before the PCLKCRx register.

Figure 3-64. PALLOCATE0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			MCAN_A	CAN_B	CAN_A	ETHERCAT	USB_A
R-0h			R/W-1h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-66. PALLOCATE0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4	MCAN_A	R/W	1h	0: MCAN_A is allocated to C28x CPU1, CM accesses to MCAN_A will be ignored and interrupts from MCAN_A will not be generated to CM. 1: MCAN_A is allocated to CM, C28x CPU1 accesses to MCAN_A will be ignored and interrupts from MCAN_A will not be generated to C28x CPU1. Note:CPU2 does not have access to MCAN_A. Reset type: XRSn
3	CAN_B	R/W	0h	0: CAN_B is allocated to C28x CPU1 or CPU2 (Mapping to CPU1 or CPU2 is determined by CPUSELx.CAN_B bit setting), CM accesses to CAN_B will be ignored and interrupts from CAN_B will not be generated to CM. 1: CAN_B is allocated to CM, C28x CPU1/2 accesses to CAN_B will be ignored and interrupts from CAN_B will not be generated to C28x CPU1/2. Reset type: XRSn
2	CAN_A	R/W	0h	0: CAN_A is allocated to C28x CPU1 or CPU2 (Mapping to CPU1 or CPU2 is determined by CPUSELx.CAN_A bit setting), CM accesses to CAN_A will be ignored and interrupts from CAN_A will not be generated to CM. 1: CAN_A is allocated to CM, C28x CPU1/2 accesses to CAN_A will be ignored and interrupts from CAN_A will not be generated to C28x CPU1/2. Reset type: XRSn

Table 3-66. PALLOCATE0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	ETHERCAT	R/W	0h	0: ETHERCAT is allocated to C28x CPU1, CM accesses to ETHERCAT will be ignored and interrupts from ETHERCAT will not be generated to CM. 1: ETHERCAT is allocated to CM, C28x CPU1 accesses to ETHERCAT will be ignored and interrupts from ETHERCAT will not be generated to C28x CPU1. Note:CPU2 does not have access to ETHERCAT. Reset type: XRSn
0	USB_A	R/W	0h	0: USB_A is allocated to C28x CPU1, CM accesses to USB_A will be ignored and interrupts from USB_A will not be generated to CM. 1: USB_A is allocated to CM, C28x CPU1 accesses to USB_A will be ignored and interrupts from USB_A will not be generated to C28x CPU1. Note:CPU2 does not have access to USB_A. Reset type: XRSn

3.16.4.5 CM_CONF_REGS_LOCK Register (Offset = 3FEh) [Reset = 0h]

CM_CONF_REGS_LOCK is shown in Figure 3-65 and described in Table 3-67.

Return to the Summary Table.

CM Configuration Registers Lock

Figure 3-65. CM_CONF_REGS_LOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK
R-0h							R/WSonce-0h

Table 3-67. CM_CONF_REGS_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	LOCK	R/WSonce	0h	0: Allows write to the following registers 1: Writes to following registers are ignored. 1. PALLOCATE0 2. RAMALLOCATE 3. CMTOCPU1NMICTL 4. CMTOCPU1INTCTL Reset type: XRSn

3.16.5 CPU_SYS_REGS Registers

Table 3-68 lists the memory-mapped registers for the CPU_SYS_REGS registers. All register offset addresses not listed in Table 3-68 should be considered as reserved locations and the register contents should not be modified.

Table 3-68. CPU_SYS_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CPUSYSLOCK1	Lock bit for CPUSYS registers	EALLOW	Go
2h	CPUSYSLOCK2	Lock bit for CPUSYS registers	EALLOW	Go
Ah	PIEVERRADDR	PIE Vector Fetch Error Address register	EALLOW	Go
22h	PCLKCR0	Peripheral Clock Gating Registers	EALLOW	Go
24h	PCLKCR1	Peripheral Clock Gating Registers	EALLOW	Go
26h	PCLKCR2	Peripheral Clock Gating Registers	EALLOW	Go
28h	PCLKCR3	Peripheral Clock Gating Registers	EALLOW	Go
2Ah	PCLKCR4	Peripheral Clock Gating Registers	EALLOW	Go
2Eh	PCLKCR6	Peripheral Clock Gating Registers	EALLOW	Go
30h	PCLKCR7	Peripheral Clock Gating Registers	EALLOW	Go
32h	PCLKCR8	Peripheral Clock Gating Registers	EALLOW	Go
34h	PCLKCR9	Peripheral Clock Gating Registers	EALLOW	Go
36h	PCLKCR10	Peripheral Clock Gating Registers	EALLOW	Go
38h	PCLKCR11	Peripheral Clock Gating Registers	EALLOW	Go
3Ch	PCLKCR13	Peripheral Clock Gating Registers	EALLOW	Go
3Eh	PCLKCR14	Peripheral Clock Gating Registers	EALLOW	Go
42h	PCLKCR16	Peripheral Clock Gating Registers	EALLOW	Go
44h	PCLKCR17	Peripheral Clock Gating Registers	EALLOW	Go
46h	PCLKCR18	Peripheral Clock Gating Registers	EALLOW	Go
4Ah	PCLKCR20	Peripheral Clock Gating Registers	EALLOW	Go
4Ch	PCLKCR21	Peripheral Clock Gating Registers	EALLOW	Go
4Eh	PCLKCR22	Peripheral Clock Gating Registers	EALLOW	Go
50h	PCLKCR23	Peripheral Clock Gating Registers	EALLOW	Go
70h	SIMRESET	Simulated Reset Register		Go
76h	LPMCR	LPM Control Register	EALLOW	Go
78h	GPIOLPMSEL0	GPIO LPM Wakeup select registers	EALLOW	Go
7Ah	GPIOLPMSEL1	GPIO LPM Wakeup select registers	EALLOW	Go
7Ch	TMR2CLKCTL	Timer2 Clock Measurement functionality control register	EALLOW	Go
7Eh	RESCCLR	Reset Cause Clear Register		Go
80h	RESC	Reset Cause register		Go
98h	MCANWAKESTATUS	MCAN Wake Status Register		Go
9Ah	MCANWAKESTATUSCLR	MCAN Wake Status Clear Register		Go

Complex bit access types are encoded to fit into small table cells. Table 3-69 shows the codes that are used for access types in this section.

Table 3-69. CPU_SYS_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read

Table 3-69. CPU_SYS_REGS Access Type Codes (continued)

Access Type	Code	Description
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.5.1 CPUSYSLOCK1 Register (Offset = 0h) [Reset = 0h]

CPUSYSLOCK1 is shown in Figure 3-66 and described in Table 3-70.

Return to the Summary Table.

Lock bit for CPUSYS registers
Notes:
[1] Any bit in this register, once set can only be cleared through a CPU1.SYSRSn. Write of 0 to any bit of this register has no effect
[2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed

Figure 3-66. CPUSYSLOCK1 Register

31	30	29	28	27	26	25	24
RESERVED	PCLKCR23	PCLKCR22	PCLKCR21	PCLKCR20	RESERVED	PCLKCR18	PCLKCR17
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
23	22	21	20	19	18	17	16
GPIOLPMSEL1	GPIOLPMSEL0	LPMCR	RESERVED	PCLKCR16	RESERVED	PCLKCR14	PCLKCR13
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
15	14	13	12	11	10	9	8
RESERVED	PCLKCR11	PCLKCR10	PCLKCR9	PCLKCR8	PCLKCR7	PCLKCR6	RESERVED
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
7	6	5	4	3	2	1	0
PCLKCR4	PCLKCR3	PCLKCR2	PCLKCR1	PCLKCR0	PIEVERRADDR	RESERVED	RESERVED
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 3-70. CPUSYSLOCK1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/WSonce	0h	Reserved
30	PCLKCR23	R/WSonce	0h	Lock bit for PCLKCR23 Register 0: Respective register is not locked 1: Respective register is locked. Note: This bit will be reserved for CPU2. Reset type: SYSRSn
29	PCLKCR22	R/WSonce	0h	Lock bit for PCLKCR22 Register 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
28	PCLKCR21	R/WSonce	0h	Lock bit for PCLKCR21 Register 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
27	PCLKCR20	R/WSonce	0h	Lock bit for PCLKCR20 Register 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
26	RESERVED	R/WSonce	0h	Reserved
25	PCLKCR18	R/WSonce	0h	Lock bit for PCLKCR18 Register 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
24	PCLKCR17	R/WSonce	0h	Lock bit for PCLKCR17 Register 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn

Table 3-70. CPUSYSLOCK1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	GPIOLPMSEL1	R/WSonce	0h	Lock bit for GPIOLPMSEL1 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
22	GPIOLPMSEL0	R/WSonce	0h	Lock bit for GPIOLPMSEL0 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
21	LPMCR	R/WSonce	0h	Lock bit for LPMCR Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
20	RESERVED	R/WSonce	0h	Reserved
19	PCLKCR16	R/WSonce	0h	Lock bit for PCLKCR16 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
18	RESERVED	R/WSonce	0h	Reserved
17	PCLKCR14	R/WSonce	0h	Lock bit for PCLKCR14 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
16	PCLKCR13	R/WSonce	0h	Lock bit for PCLKCR13 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
15	RESERVED	R/WSonce	0h	Reserved
14	PCLKCR11	R/WSonce	0h	Lock bit for PCLKCR11 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
13	PCLKCR10	R/WSonce	0h	Lock bit for PCLKCR10 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
12	PCLKCR9	R/WSonce	0h	Lock bit for PCLKCR9 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
11	PCLKCR8	R/WSonce	0h	Lock bit for PCLKCR8 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
10	PCLKCR7	R/WSonce	0h	Lock bit for PCLKCR7 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
9	PCLKCR6	R/WSonce	0h	Lock bit for PCLKCR6 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
8	RESERVED	R/WSonce	0h	Reserved
7	PCLKCR4	R/WSonce	0h	Lock bit for PCLKCR4 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn

Table 3-70. CPUSYSLOCK1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	PCLKCR3	R/WSonce	0h	Lock bit for PCLKCR3 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
5	PCLKCR2	R/WSonce	0h	Lock bit for PCLKCR2 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
4	PCLKCR1	R/WSonce	0h	Lock bit for PCLKCR1 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
3	PCLKCR0	R/WSonce	0h	Lock bit for PCLKCR0 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
2	PIEVERRADDR	R/WSonce	0h	Lock bit for PIEVERRADDR Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
1	RESERVED	R/WSonce	0h	Reserved
0	RESERVED	R/WSonce	0h	Reserved

3.16.5.2 CPUSYSLOCK2 Register (Offset = 2h) [Reset = 0h]

CPUSYSLOCK2 is shown in Figure 3-67 and described in Table 3-71.

Return to the Summary Table.

Lock bit for CPUSYS registers
Notes:
[1] Any bit in this register, once set can only be cleared through a CPU1.SYSRSn. Write of 0 to any bit of this register has no effect
[2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed

Figure 3-67. CPUSYSLOCK2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ETHERCATCTL
R-0h							R/WSonce-0h

Table 3-71. CPUSYSLOCK2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	ETHERCATCTL	R/WSonce	0h	Lock bit for ETHERCATCTL register: 0: Respective register is not locked 1: Respective register is locked. Notes: 1 Any bit in this register, once set can only be cleared through a CPU1.SYSRSn. Write of 0 to any bit of this register has no effect 2 The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed 3 This bit is reserved for CPU2 Reset type: SYSRSn

3.16.5.3 PIEVERRADDR Register (Offset = Ah) [Reset = 003FFFFFh]

PIEVERRADDR is shown in Figure 3-68 and described in Table 3-72.

Return to the Summary Table.

PIE Vector Fetch Error Address register

Figure 3-68. PIEVERRADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED										ADDR																					
R-0-0h										R/W-003FFFFFh																					

Table 3-72. PIEVERRADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R-0	0h	Reserved
21-0	ADDR	R/W	003FFFFFh	This register defines the address of the PIE Vector Fetch Error handler routine. Its the responsibility of user to initialize this register. If this register is not initialized, a default error handler at address 0x3fffbe will get executed. Reset type: XRSn

3.16.5.4 PCLKCR0 Register (Offset = 22h) [Reset = 38h]

PCLKCR0 is shown in Figure 3-69 and described in Table 3-73.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-69. PCLKCR0 Register

31	30	29	28	27	26	25	24
RESERVED							ERAD
R-0-0h							R/W-0h
23	22	21	20	19	18	17	16
RESERVED				GTBCLKSYNC	TBCLKSYNC	RESERVED	HRCAL
R-0-0h				R/W-0h	R/W-0h	R-0-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	CLA1BGCRC	CPUBGCRC	RESERVED				
R/W-0h	R/W-0h	R/W-0h	R-0-0h				
7	6	5	4	3	2	1	0
RESERVED		CPUTIMER2	CPUTIMER1	CPUTIMER0	DMA	RESERVED	CLA1
R-0-0h		R/W-1h	R/W-1h	R/W-1h	R/W-0h	R/W-0h	R/W-0h

Table 3-73. PCLKCR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R-0	0h	Reserved
24	ERAD	R/W	0h	ERAD Clock Enable Bit: When set, this enables the clock to the HRPWM module 1: ERAD clock is enabled 0: ERAD clock is disabled Reset type: SYSRSn
23-20	RESERVED	R-0	0h	Reserved
19	GTBCLKSYNC	R/W	0h	EPWM Time Base Clock Global sync: When set by CPU1, PWM time bases of all modules start counting. The effect of this bit is seen on all the EPMW modules irrespective of their partitioning based on CPUSEL Notes: 1. This bit on the CPU2.PCLKCR0 register has no effect. 2. Writing '1' to this bit overrides the effect of write '1' to the TBCLKSYNC bit at the same time Reset type: SYSRSn
18	TBCLKSYNC	R/W	0h	EPWM Time Base Clock sync: When set PWM time bases of all the PWM modules belonging to the same CPU-Subsystem (as partitioned using their CPUSEL bits) start counting Notes: 1. This bit from CPU1.PCLKCR0 or CPU2.PCLKCR0 is selected and fed to the individual EPWM modules based on their respective CPUSEL bit. Reset type: SYSRSn
17	RESERVED	R-0	0h	Reserved
16	HRCAL	R/W	0h	HRCAL Clock Enable Bit: When set, this enables the clock to the HRPWM module 1: HRCALclock is enabled 0: HRCAL clock is disabled Reset type: SYSRSn
15	RESERVED	R/W	0h	Reserved

Table 3-73. PCLKCR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14	CLA1BGCRC	R/W	0h	CLA1BGCRC Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
13	CPUBGCRC	R/W	0h	CPUBGCRC Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
12-6	RESERVED	R-0	0h	Reserved
5	CPUTIMER2	R/W	1h	CPUTIMER2 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
4	CPUTIMER1	R/W	1h	CPUTIMER1 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
3	CPUTIMER0	R/W	1h	CPUTIMER0 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
2	DMA	R/W	0h	DMA Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
1	RESERVED	R/W	0h	Reserved
0	CLA1	R/W	0h	CLA1 Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.16.5.5 PCLKCR1 Register (Offset = 24h) [Reset = 0h]

PCLKCR1 is shown in Figure 3-70 and described in Table 3-74.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-70. PCLKCR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						EMIF2	EMIF1
R-0-0h						R/W-0h	R/W-0h

Table 3-74. PCLKCR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-2	RESERVED	R-0	0h	Reserved
1	EMIF2	R/W	0h	EMIF2 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Notes: [1] These bits are not used (R/W) in CPU2.PCLKCR1 register. EMIF1 & EMIF2 clock enabled are controlled only from CPU1.PCLKCR1 register. Reset type: SYSRSn
0	EMIF1	R/W	0h	EMIF1 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Notes: [1] These bits are not used (R/W) in CPU2.PCLKCR1 register. EMIF1 & EMIF2 clock enabled are controlled only from CPU1.PCLKCR1 register. Reset type: SYSRSn

3.16.5.6 PCLKCR2 Register (Offset = 26h) [Reset = 0h]

PCLKCR2 is shown in Figure 3-71 and described in Table 3-75.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-71. PCLKCR2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
EPWM16	EPWM15	EPWM14	EPWM13	EPWM12	EPWM11	EPWM10	EPWM9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
EPWM8	EPWM7	EPWM6	EPWM5	EPWM4	EPWM3	EPWM2	EPWM1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-75. PCLKCR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15	EPWM16	R/W	0h	EPWM16 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
14	EPWM15	R/W	0h	EPWM15 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
13	EPWM14	R/W	0h	EPWM14 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
12	EPWM13	R/W	0h	EPWM13 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
11	EPWM12	R/W	0h	EPWM12 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
10	EPWM11	R/W	0h	EPWM11 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
9	EPWM10	R/W	0h	EPWM10 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

Table 3-75. PCLKCR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	EPWM9	R/W	0h	EPWM9 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
7	EPWM8	R/W	0h	EPWM8 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
6	EPWM7	R/W	0h	EPWM7 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
5	EPWM6	R/W	0h	EPWM6 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
4	EPWM5	R/W	0h	EPWM5 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
3	EPWM4	R/W	0h	EPWM4 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
2	EPWM3	R/W	0h	EPWM3 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
1	EPWM2	R/W	0h	EPWM2 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	EPWM1	R/W	0h	EPWM1 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.16.5.7 PCLKCR3 Register (Offset = 28h) [Reset = 0h]

PCLKCR3 is shown in Figure 3-72 and described in Table 3-76.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-72. PCLKCR3 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED	ECAP7	ECAP6	ECAP5	ECAP4	ECAP3	ECAP2	ECAP1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-76. PCLKCR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	RESERVED	R-0	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	ECAP7	R/W	0h	ECAP7 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
5	ECAP6	R/W	0h	ECAP6 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
4	ECAP5	R/W	0h	ECAP5 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
3	ECAP4	R/W	0h	ECAP4 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
2	ECAP3	R/W	0h	ECAP3 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
1	ECAP2	R/W	0h	ECAP2 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

Table 3-76. PCLKCR3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	ECAP1	R/W	0h	ECAP1 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.16.5.8 PCLKCR4 Register (Offset = 2Ah) [Reset = 0h]

PCLKCR4 is shown in Figure 3-73 and described in Table 3-77.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-73. PCLKCR4 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	EQEP3	EQEP2	EQEP1
R-0-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-77. PCLKCR4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-4	RESERVED	R-0	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	EQEP3	R/W	0h	EQEP3 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
1	EQEP2	R/W	0h	EQEP2 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	EQEP1	R/W	0h	EQEP1 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.16.5.9 PCLKCR6 Register (Offset = 2Eh) [Reset = 0h]

PCLKCR6 is shown in Figure 3-74 and described in Table 3-78.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-74. PCLKCR6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	SD2	SD1
R-0-0h								R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-78. PCLKCR6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	RESERVED	R-0	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	SD2	R/W	0h	SD2 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	SD1	R/W	0h	SD1 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.16.5.10 PCLKCR7 Register (Offset = 30h) [Reset = 0h]

PCLKCR7 is shown in Figure 3-75 and described in Table 3-79.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-75. PCLKCR7 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				SCI_D	SCI_C	SCI_B	SCI_A
R-0-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-79. PCLKCR7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-4	RESERVED	R-0	0h	Reserved
3	SCI_D	R/W	0h	SCI_D Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
2	SCI_C	R/W	0h	SCI_C Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
1	SCI_B	R/W	0h	SCI_B Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	SCI_A	R/W	0h	SCI_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.16.5.11 PCLKCR8 Register (Offset = 32h) [Reset = 0h]

PCLKCR8 is shown in Figure 3-76 and described in Table 3-80.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-76. PCLKCR8 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED						RESERVED	RESERVED
R-0-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				SPI_D	SPI_C	SPI_B	SPI_A
R-0-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-80. PCLKCR8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R-0	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15-4	RESERVED	R-0	0h	Reserved
3	SPI_D	R/W	0h	SPI_D Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
2	SPI_C	R/W	0h	SPI_C Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
1	SPI_B	R/W	0h	SPI_B Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	SPI_A	R/W	0h	SPI_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.16.5.12 PCLKCR9 Register (Offset = 34h) [Reset = 0h]

PCLKCR9 is shown in Figure 3-77 and described in Table 3-81.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-77. PCLKCR9 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						I2C_B	I2C_A
R-0-0h						R/W-0h	R/W-0h

Table 3-81. PCLKCR9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-2	RESERVED	R-0	0h	Reserved
1	I2C_B	R/W	0h	I2C_B Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	I2C_A	R/W	0h	I2C_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.16.5.13 PCLKCR10 Register (Offset = 36h) [Reset = 0h]

PCLKCR10 is shown in Figure 3-78 and described in Table 3-82.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-78. PCLKCR10 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	MCAN_A	RESERVED	RESERVED	CAN_B	CAN_A
R-0-0h	R-0-0h	R-0-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-82. PCLKCR10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	RESERVED	R-0	0h	Reserved
7	RESERVED	R-0	0h	Reserved
6	RESERVED	R-0	0h	Reserved
5	RESERVED	R-0	0h	Reserved
4	MCAN_A	R/W	0h	MCAN_A Clock Enable bit: 0: Module clock is gated-off including module bit clock. 1: Module clock is turned-on Note: This bit is reserved on CPU2. Reset type: SYSRSn
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	CAN_B	R/W	0h	CAN_B Clock Enable bit: 0: Module clock is gated-off i including module bit clock. 1: Module clock is turned-on Reset type: SYSRSn
0	CAN_A	R/W	0h	CAN_A Clock Enable bit: 0: Module clock is gated-off including module bit clock. 1: Module clock is turned-on Reset type: SYSRSn

3.16.5.14 PCLKCR11 Register (Offset = 38h) [Reset = 0h]

PCLKCR11 is shown in Figure 3-79 and described in Table 3-83.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-79. PCLKCR11 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED						RESERVED	USB_A
R-0-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						McBSP_B	McBSP_A
R-0-0h						R/W-0h	R/W-0h

Table 3-83. PCLKCR11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R-0	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	USB_A	R/W	0h	USB_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Notes: [1] This bit is not used (R/W) in CPU2.PCLKCR11 register. USB_A clock enabled is controlled only from CPU1.PCLKCR11 register Reset type: SYSRSn
15-2	RESERVED	R-0	0h	Reserved
1	McBSP_B	R/W	0h	McBSP_B Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	McBSP_A	R/W	0h	McBSP_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.16.5.15 PCLKCR13 Register (Offset = 3Ch) [Reset = 0h]

PCLKCR13 is shown in Figure 3-80 and described in Table 3-84.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-80. PCLKCR13 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				ADC_D	ADC_C	ADC_B	ADC_A
R-0-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-84. PCLKCR13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-4	RESERVED	R-0	0h	Reserved
3	ADC_D	R/W	0h	ADC_D Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
2	ADC_C	R/W	0h	ADC_C Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
1	ADC_B	R/W	0h	ADC_B Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	ADC_A	R/W	0h	ADC_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.16.5.16 PCLKCR14 Register (Offset = 3Eh) [Reset = 0h]

PCLKCR14 is shown in Figure 3-81 and described in Table 3-85.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-81. PCLKCR14 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
CMPSS8	CMPSS7	CMPSS6	CMPSS5	CMPSS4	CMPSS3	CMPSS2	CMPSS1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-85. PCLKCR14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	RESERVED	R-0	0h	Reserved
7	CMPSS8	R/W	0h	CMPSS8 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
6	CMPSS7	R/W	0h	CMPSS7 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
5	CMPSS6	R/W	0h	CMPSS6 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
4	CMPSS5	R/W	0h	CMPSS5 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
3	CMPSS4	R/W	0h	CMPSS4 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
2	CMPSS3	R/W	0h	CMPSS3 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
1	CMPSS2	R/W	0h	CMPSS2 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

Table 3-85. PCLKCR14 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	CMPSS1	R/W	0h	CMPSS1 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.16.5.17 PCLKCR16 Register (Offset = 42h) [Reset = 0h]

PCLKCR16 is shown in Figure 3-82 and described in Table 3-86.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-82. PCLKCR16 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED				RESERVED	DAC_C	DAC_B	DAC_A
R-0-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED							
R-0-0h							

Table 3-86. PCLKCR16 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R-0	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	DAC_C	R/W	0h	Buffered_DAC12_3 Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
17	DAC_B	R/W	0h	Buffered_DAC12_2 Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
16	DAC_A	R/W	0h	Buffered_DAC12_1 Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
15-0	RESERVED	R-0	0h	Reserved

3.16.5.18 PCLKCR17 Register (Offset = 44h) [Reset = 0h]

PCLKCR17 is shown in Figure 3-83 and described in Table 3-87.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-83. PCLKCR17 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
CLB8	CLB7	CLB6	CLB5	CLB4	CLB3	CLB2	CLB1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-87. PCLKCR17 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R-0	0h	Reserved
7	CLB8	R/W	0h	CLB8 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
6	CLB7	R/W	0h	CLB7 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
5	CLB6	R/W	0h	CLB6 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
4	CLB5	R/W	0h	CLB5 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
3	CLB4	R/W	0h	CLB4 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
2	CLB3	R/W	0h	CLB3 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
1	CLB2	R/W	0h	CLB2 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

Table 3-87. PCLKCR17 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	CLB1	R/W	0h	CLB1 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.16.5.19 PCLKCR18 Register (Offset = 46h) [Reset = 0h]

PCLKCR18 is shown in Figure 3-84 and described in Table 3-88.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-84. PCLKCR18 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
FSIRX_H	FSIRX_G	FSIRX_F	FSIRX_E	FSIRX_D	FSIRX_C	FSIRX_B	FSIRX_A
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	FSITX_B	FSITX_A
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-88. PCLKCR18 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R-0	0h	Reserved
23	FSIRX_H	R/W	0h	FSIRX_H Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
22	FSIRX_G	R/W	0h	FSIRX_G Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
21	FSIRX_F	R/W	0h	FSIRX_F Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
20	FSIRX_E	R/W	0h	FSIRX_E Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
19	FSIRX_D	R/W	0h	FSIRX_D Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
18	FSIRX_C	R/W	0h	FSIRX_C Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
17	FSIRX_B	R/W	0h	FSIRX_B Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

Table 3-88. PCLKCR18 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	FSIRX_A	R/W	0h	FSIRX_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
15-8	RESERVED	R-0	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	FSITX_B	R/W	0h	FSITX_B Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	FSITX_A	R/W	0h	FSITX_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.16.5.20 PCLKCR20 Register (Offset = 4Ah) [Reset = 0h]

PCLKCR20 is shown in Figure 3-85 and described in Table 3-89.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-85. PCLKCR20 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	PMBUS_A
R-0-0h						R/W-0h	R/W-0h

Table 3-89. PCLKCR20 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R-0	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	PMBUS_A	R/W	0h	PMBUS_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.16.5.21 PCLKCR21 Register (Offset = 4Ch) [Reset = 0h]

PCLKCR21 is shown in Figure 3-86 and described in Table 3-90.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-86. PCLKCR21 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED					DCC2	DCC1	DCC0
R-0-0h					R/W-0h	R/W-0h	R/W-0h

Table 3-90. PCLKCR21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R-0	0h	Reserved
2	DCC2	R/W	0h	DCC2 Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
1	DCC1	R/W	0h	DCC1 Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	DCC0	R/W	0h	DCC0 Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.16.5.22 PCLKCR22 Register (Offset = 4Eh) [Reset = 0h]

PCLKCR22 is shown in Figure 3-87 and described in Table 3-91.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-87. PCLKCR22 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED							MPOSTCLK
R-0-0h							R/W-0h

Table 3-91. PCLKCR22 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R-0	0h	Reserved
0	MPOSTCLK	R/W	0h	MPOST Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Note: This bit is reserved on CPU2. Reset type: SYSRSn

3.16.5.23 PCLKCR23 Register (Offset = 50h) [Reset = 0h]

PCLKCR23 is shown in Figure 3-88 and described in Table 3-92.

Return to the Summary Table.

Peripheral Clock Gating Registers
Note: All the clock inputs of the specific IP may be gated using bits of PCLKCR registers.

Figure 3-88. PCLKCR23 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED							ETHERCAT
R-0-0h							R/W-0h

Table 3-92. PCLKCR23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R-0	0h	Reserved
0	ETHERCAT	R/W	0h	ETHERCAT Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Note: This bit is reserved on CPU2. Reset type: SYSRSn

3.16.5.24 SIMRESET Register (Offset = 70h) [Reset = 0h]

SIMRESET is shown in Figure 3-89 and described in Table 3-93.

Return to the Summary Table.

Simulated Reset Register
Note: This register exists only on CPU1

Figure 3-89. SIMRESET Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						XRSn	CPU1RSn
R-0h						R-0/W1S-0h	R-0/W1S-0h

Table 3-93. SIMRESET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to this register succeeds only if this field is written with a value of 0xa5a5 Note: [1] Due to this KEY, only 32-bit writes will succeed (provided the KEY matches). 16-bit writes to the upper or lower half of this register will be ignored Reset type: XRSn
15-2	RESERVED	R	0h	Reserved
1	XRSn	R-0/W1S	0h	Writing a 1 to this field generates a XRSn like reset. Writing a 0 has no effect. Note: Writing to this pin will pull the XRSn pin low for 512 INTOSC1 clock cycles. Reset type: XRSn
0	CPU1RSn	R-0/W1S	0h	Writing a 1 to this field generates a reset to to CPU1. Writing a 0 has no effect. Reset type: XRSn

3.16.5.25 LPMCR Register (Offset = 76h) [Reset = FCh]

LPMCR is shown in Figure 3-90 and described in Table 3-94.

Return to the Summary Table.

LPM Control Register

Figure 3-90. LPMCR Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED						
R/W1S-0h	R-0-0h						
23	22	21	20	19	18	17	16
RESERVED						RESERVED	
R-0-0h						R/W-0h	
15	14	13	12	11	10	9	8
WDINTE	RESERVED						
R/W-0h	R-0-0h						
7	6	5	4	3	2	1	0
QUALSTDBY						LPM	
R/W-3Fh						R/W-0h	

Table 3-94. LPMCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W1S	0h	Reserved
30-18	RESERVED	R-0	0h	Reserved
17-16	RESERVED	R/W	0h	Reserved
15	WDINTE	R/W	0h	When this bit is set to 1, it enables the watchdog interrupt signal to wake the device from STANDBY mode. Note: [1] To use this signal, the user must also enable the WDINTn signal using the WDENINT bit in the SCSR register. Reset type: SYSRSn
14-8	RESERVED	R-0	0h	Reserved
7-2	QUALSTDBY	R/W	3Fh	Select number of OSCCLK clock cycles to qualify the selected inputs when waking the from STANDBY mode: 000000 = 2 OSCCLKs 000001 = 3 OSCCLKs ..... 111111 = 65 OSCCLKs Note: The LPMCR.QUALSTDBY register should be set to a value greater than the ratio of INTOSC1/PLLSYSCLK to ensure proper wake up. Reset type: SYSRSn
1-0	LPM	R/W	0h	These bits set the low power mode for the device. Takes effect when CPU executes the IDLE instruction (when IDLE instruction is out of EXE Phase of the Pipeline) 00: IDLE Mode 01: STANDBY Mode 1x: Reserved Reset type: SYSRSn

3.16.5.26 GPIOLPMSEL0 Register (Offset = 78h) [Reset = 0h]

GPIOLPMSEL0 is shown in Figure 3-91 and described in Table 3-95.

Return to the Summary Table.

GPIO LPM Wakeup select registers
Connects the selected pin to the LPM circuit. Refer to LPM section of the TRM for the wakeup capabilities of the selected pin.

Figure 3-91. GPIOLPMSEL0 Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-95. GPIOLPMSEL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
30	GPIO30	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
29	GPIO29	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
28	GPIO28	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
27	GPIO27	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
26	GPIO26	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
25	GPIO25	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
24	GPIO24	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
23	GPIO23	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
22	GPIO22	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn

Table 3-95. GPIOLPMSEL0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	GPIO21	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
20	GPIO20	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
19	GPIO19	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
18	GPIO18	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
17	GPIO17	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
16	GPIO16	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
15	GPIO15	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
14	GPIO14	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
13	GPIO13	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
12	GPIO12	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
11	GPIO11	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
10	GPIO10	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
9	GPIO9	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
8	GPIO8	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
7	GPIO7	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
6	GPIO6	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
5	GPIO5	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
4	GPIO4	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
3	GPIO3	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn

Table 3-95. GPIOLPMSEL0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	GPIO2	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
1	GPIO1	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
0	GPIO0	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn

3.16.5.27 GPIOLPMSEL1 Register (Offset = 7Ah) [Reset = 0h]

GPIOLPMSEL1 is shown in Figure 3-92 and described in Table 3-96.

Return to the Summary Table.

GPIO LPM Wakeup select registers
Connects the selected pin to the LPM circuit. Refer to LPM section of the TRM for the wakeup capabilities of the selected pin.

Figure 3-92. GPIOLPMSEL1 Register

31	30	29	28	27	26	25	24
GPIO63	GPIO62	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	GPIO38	GPIO37	GPIO36	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-96. GPIOLPMSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO63	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
30	GPIO62	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
29	GPIO61	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
28	GPIO60	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
27	GPIO59	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
26	GPIO58	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
25	GPIO57	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
24	GPIO56	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
23	GPIO55	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
22	GPIO54	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn

Table 3-96. GPIOLPMSEL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	GPIO53	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
20	GPIO52	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
19	GPIO51	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
18	GPIO50	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
17	GPIO49	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
16	GPIO48	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
15	GPIO47	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
14	GPIO46	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
13	GPIO45	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
12	GPIO44	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
11	GPIO43	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
10	GPIO42	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
9	GPIO41	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
8	GPIO40	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
7	GPIO39	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
6	GPIO38	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
5	GPIO37	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
4	GPIO36	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
3	GPIO35	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn

Table 3-96. GPIOLPMSEL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	GPIO34	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
1	GPIO33	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
0	GPIO32	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn

3.16.5.28 TMR2CLKCTL Register (Offset = 7Ch) [Reset = 0h]

TMR2CLKCTL is shown in Figure 3-93 and described in Table 3-97.

Return to the Summary Table.

Timer2 Clock Measurement functionality control register
This memory mapped register requires a delay of 69 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 69 NOP instructions.

Figure 3-93. TMR2CLKCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED			TMR2CLKPRESCALE			TMR2CLKSRCSEL	
R-0-0h			R/W-0h			R/W-0h	

Table 3-97. TMR2CLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-6	RESERVED	R-0	0h	Reserved
5-3	TMR2CLKPRESCALE	R/W	0h	CPU Timer 2 Clock Pre-Scale Value: These bits select the pre-scale value for the selected clock source for CPU Timer 2: 000: /1 (default on reset) 001: /2 010: /4 011: /8 100: /16 101: spare (defaults to : /16) 110: spare (defaults to : /16) 111: spare (defaults to : /16) Reset type: SYSRSn
2-0	TMR2CLKSRCSEL	R/W	0h	CPU Timer 2 Clock Source Select Bit: This bit selects the source for CPU Timer 2: 000 =SYSCLK Selected (default on reset, pre-scale is bypassed) 001 = INTOSC1 010 = INTOSC2 011 = XTAL 110 = AUXPLLCLK other values are reserved Reset type: SYSRSn

3.16.5.29 RESCCLR Register (Offset = 7Eh) [Reset = 0h]

RESCCLR is shown in Figure 3-94 and described in Table 3-98.

Return to the Summary Table.

Reset Cause Clear Register

Figure 3-94. RESCCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED				SIMRESET_XRSn	SIMRESET_CPU1RSn	ECAT_RESET_OUT	SCCRESETn
R-0-0h				W1C-0h	W1C-0h	W1C-0h	W1C-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	HWBISTn	RESERVED	NMIWDRSn	WDRSn	XRSn	POR
R-0-0h	W1C-0h	W1C-0h	R-0-0h	W1C-0h	W1C-0h	W1C-0h	W1C-0h

Table 3-98. RESCCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R-0	0h	Reserved
11	SIMRESET_XRSn	W1C	0h	Clear bit for corresponding status bit in RESC. Read of RESCCLR always gives 0. Writing a 1 to this bit clears the status bit in RESC to 0 Writing 0 has no effect. Reset type: SYSRSn
10	SIMRESET_CPU1RSn	W1C	0h	Clear bit for corresponding status bit in RESC. Read of RESCCLR always gives 0. Writing a 1 to this bit clears the status bit in RESC to 0 Writing 0 has no effect. Reset type: SYSRSn
9	ECAT_RESET_OUT	W1C	0h	Clear bit for corresponding status bit in RESC. Read of RESCCLR always gives 0. Writing a 1 to this bit clears the status bit in RESC to 0 Writing 0 has no effect. Reset type: SYSRSn
8	SCCRESETn	W1C	0h	Clear bit for corresponding status bit in RESC. Read of RESCCLR always gives 0. Writing a 1 to this bit clears the status bit in RESC to 0 Writing 0 has no effect. Reset type: SYSRSn
7	RESERVED	R-0	0h	Reserved
6	RESERVED	W1C	0h	Reserved
5	HWBISTn	W1C	0h	Clear bit for corresponding status bit in RESC. Read of RESCCLR always gives 0. Writing a 1 to this bit clears the status bit in RESC to 0 Writing 0 has no effect. Reset type: SYSRSn
4	RESERVED	R-0	0h	Reserved

Table 3-98. RESCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	NMIWDRSn	W1C	0h	Clear bit for corresponding status bit in RESC. Read of RESCCLR always gives 0. Writing a 1 to this bit clears the status bit in RESC to 0 Writing 0 has no effect. Reset type: SYSRSn
2	WDRSn	W1C	0h	Clear bit for corresponding status bit in RESC. Read of RESCCLR always gives 0. Writing a 1 to this bit clears the status bit in RESC to 0 Writing 0 has no effect. Reset type: SYSRSn
1	XRSn	W1C	0h	Clear bit for corresponding status bit in RESC. Read of RESCCLR always gives 0. Writing a 1 to this bit clears the status bit in RESC to 0 Writing 0 has no effect. Reset type: SYSRSn
0	POR	W1C	0h	Clear bit for corresponding status bit in RESC. Read of RESCCLR always gives 0. Writing a 1 to this bit clears the status bit in RESC to 0 Writing 0 has no effect. Reset type: SYSRSn

3.16.5.30 RESC Register (Offset = 80h) [Reset = X]

RESC is shown in Figure 3-95 and described in Table 3-99.

Return to the Summary Table.

Reset Cause register

Figure 3-95. RESC Register

31	30	29	28	27	26	25	24
TRSTn_pin_status	XRSn_pin_status	RESERVED					
R-X	R-X	R-0-0h					
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED				SIMRESET_XRSn	SIMRESET_CPU1RSn	ECAT_RESET_OUT	SCCRESETn
R-0-0h				R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	HWBISTn	RESERVED	NMIWDRSn	WDRSn	XRSn	POR
R-0-0h	R/W1C-0h	R/W1C-0h	R-0-0h	R/W1C-0h	R/W1C-0h	R/W1C-1h	R/W1C-1h

Table 3-99. RESC Register Field Descriptions

Bit	Field	Type	Reset	Description
31	TRSTn_pin_status	R	X	Reading this bit provides the current status of TRSTn at the respective C28x CPUs TRSTn input port. Reset value is reflective of the pin status. Reset type: PORESETn
30	XRSn_pin_status	R	X	Reading this bit provides the current status of the XRSn pin. Reset value is reflective of the pin status. Reset type: PORESETn
29-12	RESERVED	R-0	0h	Reserved
11	SIMRESET_XRSn	R/W1C	0h	If this bit is set, indicates that the device was reset by SIMRESET_XRSn Writing a 1 to this bit will force the bit to 0 Writing of 0 will have no effect. Reset type: PORESETn
10	SIMRESET_CPU1RSn	R/W1C	0h	If this bit is set, indicates that the device was reset by SIMRESET_CPU1RSn Writing a 1 to this bit will force the bit to 0 Writing of 0 will have no effect. Reset type: PORESETn
9	ECAT_RESET_OUT	R/W1C	0h	If this bit is set, indicates that the device was reset by ECAT_RESET_OUT Writing a 1 to this bit will force the bit to 0 Writing of 0 will have no effect. Reset type: PORESETn
8	SCCRESETn	R/W1C	0h	If this bit is set, indicates that the device was reset by SCCRESETn Writing a 1 to this bit will force the bit to 0 Writing of 0 will have no effect. Reset type: PORESETn
7	RESERVED	R-0	0h	Reserved
6	RESERVED	R/W1C	0h	Reserved

Table 3-99. RESC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	HWBISTn	R/W1C	0h	If this bit is set, indicates that the device was reset by HWBISTn Writing a 1 to this bit will force the bit to 0 Writing of 0 will have no effect. Reset type: PORESETn
4	RESERVED	R-0	0h	Reserved
3	NMIWDRSn	R/W1C	0h	If this bit is set, indicates that the device was reset by NMIWDRSn. Writing a 1 to this bit will force the bit to 0 Writing of 0 will have no effect. To know the exact cause of NMI after the reset, software needs to read CPU1/2.NMISHDFLG registers Reset type: PORESETn
2	WDRSn	R/W1C	0h	If this bit is set, indicates that the device was reset by WDRSn Writing a 1 to this bit will force the bit to 0 Writing of 0 will have no effect. Reset type: PORESETn
1	XRSn	R/W1C	1h	If this bit is set, indicates that the device was reset by XRSn Writing a 1 to this bit will force the bit to 0 Writing of 0 will have no effect. Reset type: PORESETn
0	POR	R/W1C	1h	If this bit is set, indicates that the device was reset by POR Writing a 1 to this bit will force the bit to 0 Writing of 0 will have no effect. Reset type: PORESETn

3.16.5.31 MCANWAKESTATUS Register (Offset = 98h) [Reset = 0h]

MCANWAKESTATUS is shown in Figure 3-96 and described in Table 3-100.

Return to the Summary Table.

MCAN Wake Status Register

Figure 3-96. MCANWAKESTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							WAKE
R-0h							R-0h

Table 3-100. MCANWAKESTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	WAKE	R	0h	0 : wakeup event has not occured. 1 : wakeup event has occured. Reset type: CPU1.SYSRSn

3.16.5.32 MCANWAKESTATUSCLR Register (Offset = 9Ah) [Reset = 0h]

MCANWAKESTATUSCLR is shown in Figure 3-97 and described in Table 3-101.

Return to the Summary Table.

MCAN Wake Status Clear Register

Figure 3-97. MCANWAKESTATUSCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							WAKE
R-0h							R-0/W1S-0h

Table 3-101. MCANWAKESTATUSCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	WAKE	R-0/W1S	0h	0 : No effect. 1 : Clears WAKE bit of MCANWAKESTATUS register Reset type: CPU1.SYSRSn

3.16.6 CPU_ID_REGS Registers

Table 3-102 lists the memory-mapped registers for the CPU_ID_REGS registers. All register offset addresses not listed in Table 3-102 should be considered as reserved locations and the register contents should not be modified.

Table 3-102. CPU_ID_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CPUID	Indicates CPU1 or CPU2		Go

Complex bit access types are encoded to fit into small table cells. Table 3-103 shows the codes that are used for access types in this section.

Table 3-103. CPU_ID_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.6.1 CPUID Register (Offset = 0h) [Reset = X]

CPUID is shown in Figure 3-98 and described in Table 3-104.

Return to the Summary Table.

This register can be used to identify on which CPU the code is executing.

Figure 3-98. CPUID Register

15	14	13	12	11	10	9	8
RESERVED							
R-X							
7	6	5	4	3	2	1	0
CPUID							
R-X							

Table 3-104. CPUID Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	X	Reserved
7-0	CPUID	R	X	CPUID = 1 for CPU1, 2 for CPU2 Reset type: N/A

3.16.7 CPU1_PERIPH_AC_REGS Registers

Table 3-105 lists the memory-mapped registers for the CPU1_PERIPH_AC_REGS registers. All register offset addresses not listed in Table 3-105 should be considered as reserved locations and the register contents should not be modified.

Table 3-105. CPU1_PERIPH_AC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	ADCA_AC	ADCA Master Access Control Register	EALLOW	Go
2h	ADCB_AC	ADCB Master Access Control Register	EALLOW	Go
4h	ADCC_AC	ADCC Master Access Control Register	EALLOW	Go
6h	ADCD_AC	ADCD Master Access Control Register	EALLOW	Go
10h	CMPSS1_AC	CMPSS1 Master Access Control Register	EALLOW	Go
12h	CMPSS2_AC	CMPSS2 Master Access Control Register	EALLOW	Go
14h	CMPSS3_AC	CMPSS3 Master Access Control Register	EALLOW	Go
16h	CMPSS4_AC	CMPSS4 Master Access Control Register	EALLOW	Go
18h	CMPSS5_AC	CMPSS5 Master Access Control Register	EALLOW	Go
1Ah	CMPSS6_AC	CMPSS6 Master Access Control Register	EALLOW	Go
1Ch	CMPSS7_AC	CMPSS7 Master Access Control Register	EALLOW	Go
1Eh	CMPSS8_AC	CMPSS8 Master Access Control Register	EALLOW	Go
28h	DACA_AC	DACA Master Access Control Register	EALLOW	Go
2Ah	DACB_AC	DACB Master Access Control Register	EALLOW	Go
2Ch	DACC_AC	DACC Master Access Control Register	EALLOW	Go
48h	EPWM1_AC	EPWM1 Master Access Control Register	EALLOW	Go
4Ah	EPWM2_AC	EPWM2 Master Access Control Register	EALLOW	Go
4Ch	EPWM3_AC	EPWM3 Master Access Control Register	EALLOW	Go
4Eh	EPWM4_AC	EPWM4 Master Access Control Register	EALLOW	Go
50h	EPWM5_AC	EPWM5 Master Access Control Register	EALLOW	Go
52h	EPWM6_AC	EPWM6 Master Access Control Register	EALLOW	Go
54h	EPWM7_AC	EPWM7 Master Access Control Register	EALLOW	Go
56h	EPWM8_AC	EPWM8 Master Access Control Register	EALLOW	Go
58h	EPWM9_AC	EPWM9 Master Access Control Register	EALLOW	Go
5Ah	EPWM10_AC	EPWM10 Master Access Control Register	EALLOW	Go
5Ch	EPWM11_AC	EPWM11 Master Access Control Register	EALLOW	Go
5Eh	EPWM12_AC	EPWM12 Master Access Control Register	EALLOW	Go
60h	EPWM13_AC	EPWM13 Master Access Control Register	EALLOW	Go
62h	EPWM14_AC	EPWM14 Master Access Control Register	EALLOW	Go
64h	EPWM15_AC	EPWM15 Master Access Control Register	EALLOW	Go
66h	EPWM16_AC	EPWM16 Master Access Control Register	EALLOW	Go
70h	EQEP1_AC	EQEP1 Master Access Control Register	EALLOW	Go
72h	EQEP2_AC	EQEP2 Master Access Control Register	EALLOW	Go
74h	EQEP3_AC	EQEP3 Master Access Control Register	EALLOW	Go
80h	ECAP1_AC	ECAP1 Master Access Control Register	EALLOW	Go
82h	ECAP2_AC	ECAP2 Master Access Control Register	EALLOW	Go
84h	ECAP3_AC	ECAP3 Master Access Control Register	EALLOW	Go
86h	ECAP4_AC	ECAP4 Master Access Control Register	EALLOW	Go
88h	ECAP5_AC	ECAP5 Master Access Control Register	EALLOW	Go
8Ah	ECAP6_AC	ECAP6 Master Access Control Register	EALLOW	Go
8Ch	ECAP7_AC	ECAP7 Master Access Control Register	EALLOW	Go

Table 3-105. CPU1_PERIPH_AC_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
A8h	SDFM1_AC	SDFM1 Master Access Control Register	EALLOW	Go
AAh	SDFM2_AC	SDFM2 Master Access Control Register	EALLOW	Go
B0h	CLB1_AC	CLB1 Master Access Control Register	EALLOW	Go
B2h	CLB2_AC	CLB2 Master Access Control Register	EALLOW	Go
B4h	CLB3_AC	CLB3 Master Access Control Register	EALLOW	Go
B6h	CLB4_AC	CLB4 Master Access Control Register	EALLOW	Go
B8h	CLB5_AC	CLB5 Master Access Control Register	EALLOW	Go
BAh	CLB6_AC	CLB6 Master Access Control Register	EALLOW	Go
BCh	CLB7_AC	CLB7 Master Access Control Register	EALLOW	Go
BEh	CLB8_AC	CLB8 Master Access Control Register	EALLOW	Go
110h	SPIA_AC	SPIA Master Access Control Register	EALLOW	Go
112h	SPIB_AC	SPIB Master Access Control Register	EALLOW	Go
114h	SPIC_AC	SPIC Master Access Control Register	EALLOW	Go
116h	SPID_AC	SPID Master Access Control Register	EALLOW	Go
130h	PMBUS_A_AC	PMBUSA Master Access Control Register	EALLOW	Go
140h	CAN_A_AC	CAN_A Master Access Control Register	EALLOW	Go
142h	CAN_B_AC	CAN_B Master Access Control Register	EALLOW	Go
150h	MCBSPA_AC	MCBSPA Master Access Control Register	EALLOW	Go
152h	MCBSPB_AC	MCBSPB Master Access Control Register	EALLOW	Go
180h	USBA_AC	USBA Master Access Control Register	EALLOW	Go
1A8h	HRPWM_AC	HRPWM Master Access Control Register	EALLOW	Go
1AAh	ETHERCAT_AC	ETHERCAT Master Access Control Register	EALLOW	Go
1B0h	FSIATX_AC	FSIATX Master Access Control Register	EALLOW	Go
1B2h	FSIARX_AC	FSIARX Master Access Control Register	EALLOW	Go
1B4h	FSIBTX_AC	FSIBTX Master Access Control Register	EALLOW	Go
1B6h	FSIBRX_AC	FSIBRX Master Access Control Register	EALLOW	Go
1BAh	FSICRX_AC	FSICRX Master Access Control Register	EALLOW	Go
1BEh	FSIDRX_AC	FSIDRX Master Access Control Register	EALLOW	Go
1C2h	FSIERX_AC	FSIERX Master Access Control Register	EALLOW	Go
1C6h	FSIFRX_AC	FSIFRX Master Access Control Register	EALLOW	Go
1CAh	FSIGRX_AC	FSIGRX Master Access Control Register	EALLOW	Go
1CEh	FSIHRX_AC	FSIHRX Master Access Control Register	EALLOW	Go
1D0h	MCANA_AC	MCANA Master Access Control Register	EALLOW	Go
1FEh	PERIPH_AC_LOCK	Lock Register to stop Write access to peripheral Access register.	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-106 shows the codes that are used for access types in this section.

Table 3-106. CPU1_PERIPH_AC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		

Table 3-106. CPU1_PERIPH_AC_REGS Access Type Codes (continued)

Access Type	Code	Description
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.7.1 ADCA_AC Register (Offset = 0h) [Reset = 3Fh]

ADCA_AC is shown in Figure 3-99 and described in Table 3-107.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-99. ADCA_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-107. ADCA_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.2 ADCB_AC Register (Offset = 2h) [Reset = 3Fh]

ADCB_AC is shown in Figure 3-100 and described in Table 3-108.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-100. ADCB_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-108. ADCB_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.3 ADCC_AC Register (Offset = 4h) [Reset = 3Fh]

ADCC_AC is shown in Figure 3-101 and described in Table 3-109.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-101. ADCC_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-109. ADCC_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.4 ADCD_AC Register (Offset = 6h) [Reset = 3Fh]

ADCD_AC is shown in Figure 3-102 and described in Table 3-110.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-102. ADCD_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-110. ADCD_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.5 CMPSS1_AC Register (Offset = 10h) [Reset = 3Fh]

CMPSS1_AC is shown in Figure 3-103 and described in Table 3-111.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-103. CMPSS1_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-111. CMPSS1_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.6 CMPSS2_AC Register (Offset = 12h) [Reset = 3Fh]

CMPSS2_AC is shown in Figure 3-104 and described in Table 3-112.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-104. CMPSS2_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-112. CMPSS2_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.7 CMPSS3_AC Register (Offset = 14h) [Reset = 3Fh]

CMPSS3_AC is shown in Figure 3-105 and described in Table 3-113.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-105. CMPSS3_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-113. CMPSS3_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.8 CMPSS4_AC Register (Offset = 16h) [Reset = 3Fh]

CMPSS4_AC is shown in Figure 3-106 and described in Table 3-114.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-106. CMPSS4_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-114. CMPSS4_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.9 CMPSS5_AC Register (Offset = 18h) [Reset = 3Fh]

CMPSS5_AC is shown in Figure 3-107 and described in Table 3-115.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-107. CMPSS5_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-115. CMPSS5_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.10 CMPSS6_AC Register (Offset = 1Ah) [Reset = 3Fh]

CMPSS6_AC is shown in Figure 3-108 and described in Table 3-116.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-108. CMPSS6_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-116. CMPSS6_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.11 CMPSS7_AC Register (Offset = 1Ch) [Reset = 3Fh]

CMPSS7_AC is shown in Figure 3-109 and described in Table 3-117.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-109. CMPSS7_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-117. CMPSS7_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.12 CMPSS8_AC Register (Offset = 1Eh) [Reset = 3Fh]

CMPSS8_AC is shown in Figure 3-110 and described in Table 3-118.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-110. CMPSS8_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-118. CMPSS8_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.13 DACA_AC Register (Offset = 28h) [Reset = 3Fh]

DACA_AC is shown in Figure 3-111 and described in Table 3-119.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-111. DACA_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-119. DACA_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.14 DACB_AC Register (Offset = 2Ah) [Reset = 3Fh]

DACB_AC is shown in Figure 3-112 and described in Table 3-120.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-112. DACB_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-120. DACB_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.15 DACC_AC Register (Offset = 2Ch) [Reset = 3Fh]

DACC_AC is shown in Figure 3-113 and described in Table 3-121.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-113. DACC_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-121. DACC_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.16 EPWM1_AC Register (Offset = 48h) [Reset = 3Fh]

EPWM1_AC is shown in Figure 3-114 and described in Table 3-122.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-114. EPWM1_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-122. EPWM1_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.17 EPWM2_AC Register (Offset = 4Ah) [Reset = 3Fh]

EPWM2_AC is shown in Figure 3-115 and described in Table 3-123.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-115. EPWM2_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-123. EPWM2_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.18 EPWM3_AC Register (Offset = 4Ch) [Reset = 3Fh]

EPWM3_AC is shown in Figure 3-116 and described in Table 3-124.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-116. EPWM3_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-124. EPWM3_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.19 EPWM4_AC Register (Offset = 4Eh) [Reset = 3Fh]

EPWM4_AC is shown in Figure 3-117 and described in Table 3-125.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-117. EPWM4_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-125. EPWM4_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.20 EPWM5_AC Register (Offset = 50h) [Reset = 3Fh]

EPWM5_AC is shown in Figure 3-118 and described in Table 3-126.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-118. EPWM5_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-126. EPWM5_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.21 EPWM6_AC Register (Offset = 52h) [Reset = 3Fh]

EPWM6_AC is shown in Figure 3-119 and described in Table 3-127.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-119. EPWM6_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-127. EPWM6_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.22 EPWM7_AC Register (Offset = 54h) [Reset = 3Fh]

EPWM7_AC is shown in Figure 3-120 and described in Table 3-128.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-120. EPWM7_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-128. EPWM7_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.23 EPWM8_AC Register (Offset = 56h) [Reset = 3Fh]

EPWM8_AC is shown in Figure 3-121 and described in Table 3-129.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-121. EPWM8_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-129. EPWM8_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.24 EPWM9_AC Register (Offset = 58h) [Reset = 3Fh]

EPWM9_AC is shown in Figure 3-122 and described in Table 3-130.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-122. EPWM9_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-130. EPWM9_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.25 EPWM10_AC Register (Offset = 5Ah) [Reset = 3Fh]

EPWM10_AC is shown in Figure 3-123 and described in Table 3-131.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-123. EPWM10_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-131. EPWM10_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.26 EPWM11_AC Register (Offset = 5Ch) [Reset = 3Fh]

EPWM11_AC is shown in Figure 3-124 and described in Table 3-132.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-124. EPWM11_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-132. EPWM11_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.27 EPWM12_AC Register (Offset = 5Eh) [Reset = 3Fh]

EPWM12_AC is shown in Figure 3-125 and described in Table 3-133.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-125. EPWM12_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-133. EPWM12_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.28 EPWM13_AC Register (Offset = 60h) [Reset = 3Fh]

EPWM13_AC is shown in Figure 3-126 and described in Table 3-134.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-126. EPWM13_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-134. EPWM13_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.29 EPWM14_AC Register (Offset = 62h) [Reset = 3Fh]

EPWM14_AC is shown in Figure 3-127 and described in Table 3-135.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-127. EPWM14_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-135. EPWM14_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.30 EPWM15_AC Register (Offset = 64h) [Reset = 3Fh]

EPWM15_AC is shown in Figure 3-128 and described in Table 3-136.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-128. EPWM15_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-136. EPWM15_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.31 EPWM16_AC Register (Offset = 66h) [Reset = 3Fh]

EPWM16_AC is shown in Figure 3-129 and described in Table 3-137.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-129. EPWM16_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-137. EPWM16_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.32 EQEP1_AC Register (Offset = 70h) [Reset = 3Fh]

EQEP1_AC is shown in Figure 3-130 and described in Table 3-138.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-130. EQEP1_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-138. EQEP1_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.33 EQEP2_AC Register (Offset = 72h) [Reset = 3Fh]

EQEP2_AC is shown in Figure 3-131 and described in Table 3-139.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-131. EQEP2_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-139. EQEP2_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.34 EQEP3_AC Register (Offset = 74h) [Reset = 3Fh]

EQEP3_AC is shown in Figure 3-132 and described in Table 3-140.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-132. EQEP3_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-140. EQEP3_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.35 ECAP1_AC Register (Offset = 80h) [Reset = 3Fh]

ECAP1_AC is shown in Figure 3-133 and described in Table 3-141.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-133. ECAP1_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-141. ECAP1_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.36 ECAP2_AC Register (Offset = 82h) [Reset = 3Fh]

ECAP2_AC is shown in Figure 3-134 and described in Table 3-142.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-134. ECAP2_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-142. ECAP2_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.37 ECAP3_AC Register (Offset = 84h) [Reset = 3Fh]

ECAP3_AC is shown in Figure 3-135 and described in Table 3-143.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-135. ECAP3_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-143. ECAP3_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.38 ECAP4_AC Register (Offset = 86h) [Reset = 3Fh]

ECAP4_AC is shown in Figure 3-136 and described in Table 3-144.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-136. ECAP4_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-144. ECAP4_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.39 ECAP5_AC Register (Offset = 88h) [Reset = 3Fh]

ECAP5_AC is shown in Figure 3-137 and described in Table 3-145.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-137. ECAP5_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-145. ECAP5_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.40 ECAP6_AC Register (Offset = 8Ah) [Reset = 3Fh]

ECAP6_AC is shown in Figure 3-138 and described in Table 3-146.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-138. ECAP6_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-146. ECAP6_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.41 ECAP7_AC Register (Offset = 8Ch) [Reset = 3Fh]

ECAP7_AC is shown in Figure 3-139 and described in Table 3-147.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-139. ECAP7_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-147. ECAP7_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.42 SDFM1_AC Register (Offset = A8h) [Reset = 3Fh]

SDFM1_AC is shown in Figure 3-140 and described in Table 3-148.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-140. SDFM1_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-148. SDFM1_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.43 SDFM2_AC Register (Offset = AAh) [Reset = 3Fh]

SDFM2_AC is shown in Figure 3-141 and described in Table 3-149.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-141. SDFM2_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-149. SDFM2_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.44 CLB1_AC Register (Offset = B0h) [Reset = 3Fh]

CLB1_AC is shown in Figure 3-142 and described in Table 3-150.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-142. CLB1_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		RESERVED		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-150. CLB1_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	RESERVED	R/W	3h	Reserved
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.45 CLB2_AC Register (Offset = B2h) [Reset = 3Fh]

CLB2_AC is shown in Figure 3-143 and described in Table 3-151.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-143. CLB2_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		RESERVED		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-151. CLB2_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	RESERVED	R/W	3h	Reserved
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.46 CLB3_AC Register (Offset = B4h) [Reset = 3Fh]

CLB3_AC is shown in Figure 3-144 and described in Table 3-152.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-144. CLB3_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		RESERVED		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-152. CLB3_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	RESERVED	R/W	3h	Reserved
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.47 CLB4_AC Register (Offset = B6h) [Reset = 3Fh]

CLB4_AC is shown in Figure 3-145 and described in Table 3-153.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-145. CLB4_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		RESERVED		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-153. CLB4_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	RESERVED	R/W	3h	Reserved
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.48 CLB5_AC Register (Offset = B8h) [Reset = 3Fh]

CLB5_AC is shown in Figure 3-146 and described in Table 3-154.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-146. CLB5_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		RESERVED		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-154. CLB5_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	RESERVED	R/W	3h	Reserved
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.49 CLB6_AC Register (Offset = BAh) [Reset = 3Fh]

CLB6_AC is shown in Figure 3-147 and described in Table 3-155.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-147. CLB6_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		RESERVED		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-155. CLB6_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	RESERVED	R/W	3h	Reserved
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.50 CLB7_AC Register (Offset = BCh) [Reset = 3Fh]

CLB7_AC is shown in Figure 3-148 and described in Table 3-156.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-148. CLB7_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		RESERVED		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-156. CLB7_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	RESERVED	R/W	3h	Reserved
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.51 CLB8_AC Register (Offset = BEh) [Reset = 3Fh]

CLB8_AC is shown in Figure 3-149 and described in Table 3-157.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-149. CLB8_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		RESERVED		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-157. CLB8_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	RESERVED	R/W	3h	Reserved
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.52 SPIA_AC Register (Offset = 110h) [Reset = 3Fh]

SPIA_AC is shown in Figure 3-150 and described in Table 3-158.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-150. SPIA_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-158. SPIA_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.53 SPIB_AC Register (Offset = 112h) [Reset = 3Fh]

SPIB_AC is shown in Figure 3-151 and described in Table 3-159.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-151. SPIB_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-159. SPIB_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.54 SPIC_AC Register (Offset = 114h) [Reset = 3Fh]

SPIC_AC is shown in Figure 3-152 and described in Table 3-160.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-152. SPIC_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-160. SPIC_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.55 SPID_AC Register (Offset = 116h) [Reset = 3Fh]

SPID_AC is shown in Figure 3-153 and described in Table 3-161.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-153. SPID_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-161. SPID_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.56 PMBUS_A_AC Register (Offset = 130h) [Reset = 3Fh]

PMBUS_A_AC is shown in Figure 3-154 and described in Table 3-162.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-154. PMBUS_A_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-162. PMBUS_A_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.57 CAN_A_AC Register (Offset = 140h) [Reset = 3Fh]

CAN_A_AC is shown in Figure 3-155 and described in Table 3-163.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-155. CAN_A_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		RESERVED		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-163. CAN_A_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	RESERVED	R/W	3h	Reserved
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.58 CAN_B_AC Register (Offset = 142h) [Reset = 3Fh]

CAN_B_AC is shown in Figure 3-156 and described in Table 3-164.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-156. CAN_B_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED			DMA1_ACC		RESERVED		CPUx_ACC
R-0-0h			R/W-3h		R/W-3h		R/W-3h

Table 3-164. CAN_B_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	RESERVED	R/W	3h	Reserved
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.59 MCBSPA_AC Register (Offset = 150h) [Reset = 3Fh]

MCBSPA_AC is shown in Figure 3-157 and described in Table 3-165.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-157. MCBSPA_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-165. MCBSPA_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.60 MCBSPB_AC Register (Offset = 152h) [Reset = 3Fh]

MCBSPB_AC is shown in Figure 3-158 and described in Table 3-166.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-158. MCBSPB_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-166. MCBSPB_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.61 USBA_AC Register (Offset = 180h) [Reset = 3Fh]

USBA_AC is shown in Figure 3-159 and described in Table 3-167.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-159. USBA_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		RESERVED		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-167. USBA_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	RESERVED	R/W	3h	Reserved
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.62 HRPWM_AC Register (Offset = 1A8h) [Reset = 3Fh]

HRPWM_AC is shown in Figure 3-160 and described in Table 3-168.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.
Note: Following registers are controlled by this register:
HRPWR
HRCAL
HRPRD
HRCNT0
HRCNT1
HRMSTEP

Figure 3-160. HRPWM_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-168. HRPWM_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Note: Following registers are covered by this register HRPWR HRCAL HRPRD HRCNT0 HRCNT1 HRMSTEP Reset type: XRSn

Table 3-168. HRPWM_AC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Note: Following registers are covered by this register HRPWR HRCAL HRPRD HRCNT0 HRCNT1 HRMSTEP Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Note: Following registers are covered by this register HRPWR HRCAL HRPRD HRCNT0 HRCNT1 HRMSTEP Reset type: XRSn

3.16.7.63 ETHERCAT_AC Register (Offset = 1AAh) [Reset = 3Fh]

ETHERCAT_AC is shown in Figure 3-161 and described in Table 3-169.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-161. ETHERCAT_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED			DMA1_ACC		RESERVED		CPU1_ACC
R-0-0h			R/W-3h		R/W-3h		R/W-3h

Table 3-169. ETHERCAT_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	RESERVED	R/W	3h	Reserved
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.64 FSIATX_AC Register (Offset = 1B0h) [Reset = 3Fh]

FSIATX_AC is shown in Figure 3-162 and described in Table 3-170.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-162. FSIATX_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-170. FSIATX_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.65 FSIARX_AC Register (Offset = 1B2h) [Reset = 3Fh]

FSIARX_AC is shown in Figure 3-163 and described in Table 3-171.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-163. FSIARX_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-171. FSIARX_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.66 FSIBTX_AC Register (Offset = 1B4h) [Reset = 3Fh]

FSIBTX_AC is shown in Figure 3-164 and described in Table 3-172.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-164. FSIBTX_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-172. FSIBTX_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.67 FSIBRX_AC Register (Offset = 1B6h) [Reset = 3Fh]

FSIBRX_AC is shown in Figure 3-165 and described in Table 3-173.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-165. FSIBRX_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-173. FSIBRX_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.68 FSICRX_AC Register (Offset = 1BAh) [Reset = 3Fh]

FSICRX_AC is shown in Figure 3-166 and described in Table 3-174.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-166. FSICRX_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-174. FSICRX_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.69 FSIDRX_AC Register (Offset = 1BEh) [Reset = 3Fh]

FSIDRX_AC is shown in Figure 3-167 and described in Table 3-175.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-167. FSIDRX_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-175. FSIDRX_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.70 FSIERX_AC Register (Offset = 1C2h) [Reset = 3Fh]

FSIERX_AC is shown in Figure 3-168 and described in Table 3-176.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-168. FSIERX_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-176. FSIERX_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.71 FSIFRX_AC Register (Offset = 1C6h) [Reset = 3Fh]

FSIFRX_AC is shown in Figure 3-169 and described in Table 3-177.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-169. FSIFRX_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-177. FSIFRX_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.72 FSIGRX_AC Register (Offset = 1CAh) [Reset = 3Fh]

FSIGRX_AC is shown in Figure 3-170 and described in Table 3-178.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-170. FSIGRX_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-178. FSIGRX_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.73 FSIHRX_AC Register (Offset = 1CEh) [Reset = 3Fh]

FSIHRX_AC is shown in Figure 3-171 and described in Table 3-179.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-171. FSIHRX_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		CLA1_ACC		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-179. FSIHRX_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.74 MCANA_AC Register (Offset = 1D0h) [Reset = 3Fh]

MCANA_AC is shown in Figure 3-172 and described in Table 3-180.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to peripheral from corresponding connected master.

Figure 3-172. MCANA_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		RESERVED		CPUx_ACC	
R-0-0h		R/W-3h		R/W-3h		R/W-3h	

Table 3-180. MCANA_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	RESERVED	R/W	3h	Reserved
1-0	CPUx_ACC	R/W	3h	Defines Access control definition for the CPUx as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.16.7.75 PERIPH_AC_LOCK Register (Offset = 1FEh) [Reset = 0h]

PERIPH_AC_LOCK is shown in Figure 3-173 and described in Table 3-181.

Return to the Summary Table.

Based on status bit control the Access registers are either RD/WR or RD only.

Figure 3-173. PERIPH_AC_LOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK_AC_WR
R-0-0h							R/WSonce-0h

Table 3-181. PERIPH_AC_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R-0	0h	Reserved
0	LOCK_AC_WR	R/WSonce	0h	Defines Access control definition for the CPUx as: 1: Access Control registers are Read Only 0: Read/Write Access allowed to Access Control registers. Writing '1' sets the bit, writing '0' has no effect. Reset type: CPUx.SYSRSn

3.16.8 CPUTIMER_REGS Registers

Table 3-182 lists the memory-mapped registers for the CPUTIMER_REGS registers. All register offset addresses not listed in Table 3-182 should be considered as reserved locations and the register contents should not be modified.

Table 3-182. CPUTIMER_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	TIM	CPU-Timer, Counter Register		Go
2h	PRD	CPU-Timer, Period Register		Go
4h	TCR	CPU-Timer, Control Register		Go
6h	TPR	CPU-Timer, Prescale Register		Go
7h	TPRH	CPU-Timer, Prescale Register High		Go

Complex bit access types are encoded to fit into small table cells. Table 3-183 shows the codes that are used for access types in this section.

Table 3-183. CPUTIMER_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.8.1 TIM Register (Offset = 0h) [Reset = FFFFh]

TIM is shown in Figure 3-174 and described in Table 3-184.

Return to the Summary Table.

CPU-Timer, Counter Register

Figure 3-174. TIM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSW																LSW															
R/W-0h																R/W-FFFFh															

Table 3-184. TIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MSW	R/W	0h	CPU-Timer Counter Registers The TIMH register holds the high 16 bits of the current 32-bit count of the timer. The TIMH:TIM decrements by one every (TDDRH:TDDR+1) clock cycles, where TDDRH:TDDR is the timer prescale dividedown value. When the TIMH:TIM decrements to zero, the TIMH:TIM register is reloaded with the period value contained in the PRDH:PRD registers. The timer interrupt (TINT) signal is generated. Reset type: SYSRSn
15-0	LSW	R/W	FFFFh	CPU-Timer Counter Registers The TIM register holds the low 16 bits of the current 32-bit count of the timer. The TIMH:TIM decrements by one every (TDDRH:TDDR+1) clock cycles, where TDDRH:TDDR is the timer prescale dividedown value. When the TIMH:TIM decrements to zero, the TIMH:TIM register is reloaded with the period value contained in the PRDH:PRD registers. The timer interrupt (TINT) signal is generated. Reset type: SYSRSn

3.16.8.2 PRD Register (Offset = 2h) [Reset = FFFFh]

PRD is shown in Figure 3-175 and described in Table 3-185.

Return to the Summary Table.

CPU-Timer, Period Register

Figure 3-175. PRD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSW																LSW															
R/W-0h																R/W-FFFFh															

Table 3-185. PRD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MSW	R/W	0h	CPU-Timer Period Registers The PRDH register holds the high 16 bits of the 32-bit period. When the TIMH:TIM decrements to zero, the TIMH:TIM register is reloaded with the period value contained in the PRDH:PRD registers, at the start of the next timer input clock cycle (the output of the prescaler). The PRDH:PRD contents are also loaded into the TIMH:TIM when you set the timer reload bit (TRB) in the Timer Control Register (TCR). Reset type: SYSRSn
15-0	LSW	R/W	FFFFh	CPU-Timer Period Registers The PRD register holds the low 16 bits of the 32-bit period. When the TIMH:TIM decrements to zero, the TIMH:TIM register is reloaded with the period value contained in the PRDH:PRD registers, at the start of the next timer input clock cycle (the output of the prescaler). The PRDH:PRD contents are also loaded into the TIMH:TIM when you set the timer reload bit (TRB) in the Timer Control Register (TCR). Reset type: SYSRSn

3.16.8.3 TCR Register (Offset = 4h) [Reset = 1h]

TCR is shown in Figure 3-176 and described in Table 3-186.

Return to the Summary Table.

CPU-Timer, Control Register

Figure 3-176. TCR Register

15	14	13	12	11	10	9	8
TIF	TIE	RESERVED		FREE	SOFT	RESERVED	
R/W1C-0h	R/W-0h	R-0h		R/W-0h	R/W-0h	R-0h	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RESERVED		TRB	TSS	RESERVED			
R-0h		R/W-0h	R/W-0h	R-1h			

Table 3-186. TCR Register Field Descriptions

Bit	Field	Type	Reset	Description
15	TIF	R/W1C	0h	CPU-Timer Overflow Flag. TIF indicates whether a timer overflow has happened since TIF was last cleared. TIF is not cleared automatically and does not need to be cleared to enable the next timer interrupt. Reset type: SYSRSn 0h (R/W) = The CPU-Timer has not decremented to zero. Writes of 0 are ignored. 1h (R/W) = This flag gets set when the CPU-timer decrements to zero. Writing a 1 to this bit clears the flag.
14	TIE	R/W	0h	CPU-Timer Interrupt Enable. Reset type: SYSRSn 0h (R/W) = The CPU-Timer interrupt is disabled. 1h (R/W) = The CPU-Timer interrupt is enabled. If the timer decrements to zero, and TIE is set, the timer asserts its interrupt request.
13-12	RESERVED	R	0h	Reserved
11	FREE	R/W	0h	If the FREE bit is set to 1, then, upon a software breakpoint, the timer continues to run. If FREE is 0, then the SOFT bit controls the emulation behavior. Reset type: SYSRSn 0h (R/W) = Stop after the next decrement of the TIMH:TIM (hard stop) (SOFT bit controls the emulation behavior) 1h (R/W) = Free Run (SOFT bit is don't care, counter is free running)
10	SOFT	R/W	0h	If the FREE bit is set to 1, then, upon a software breakpoint, the timer continues to run (that is, free runs). In this case, SOFT is a don't care. But if FREE is 0, then SOFT takes effect. Reset type: SYSRSn 0h (R/W) = Stop after the next decrement of the TIMH:TIM (hard stop). (ONLY if FREE=0, if FREE=1 this bit is don't care) 1h (R/W) = Stop after the TIMH:TIM decrements to 0 (soft stop) In the SOFT STOP mode, the timer generates an interrupt before shutting down (since reaching 0 is the interrupt causing condition). (ONLY if FREE=0, if FREE=1 this bit is don't care)
9-6	RESERVED	R	0h	Reserved

Table 3-186. TCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	TRB	R/W	0h	Timer reload Reset type: SYSRSn 0h (R/W) = The TRB bit is always read as zero. Writes of 0 are ignored. 1h (R/W) = When you write a 1 to TRB, the TIMH:TIM is loaded with the value in the PRDH:PRD, and the prescaler counter (PSCH:PSC) is loaded with the value in the timer dividedown register (TDDRH:TDDR).
4	TSS	R/W	0h	CPU-Timer stop status bit. TSS is a 1-bit flag that stops or starts the CPU-timer. Reset type: SYSRSn 0h (R/W) = Reads of 0 indicate the CPU-timer is running. To start or restart the CPU-timer, set TSS to 0. At reset, TSS is cleared to 0 and the CPU-timer immediately starts. 1h (R/W) = Reads of 1 indicate that the CPU-timer is stopped. To stop the CPU-timer, set TSS to 1.
3-0	RESERVED	R	1h	Reserved

3.16.8.4 TPR Register (Offset = 6h) [Reset = 0h]

TPR is shown in Figure 3-177 and described in Table 3-187.

Return to the Summary Table.

CPU-Timer, Prescale Register

Figure 3-177. TPR Register

15	14	13	12	11	10	9	8
PSC							
R-0h							
7	6	5	4	3	2	1	0
TDDR							
R/W-0h							

Table 3-187. TPR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	PSC	R	0h	CPU-Timer Prescale Counter. These bits hold the current prescale count for the timer. For every timer clock source cycle that the PSCH:PSC value is greater than 0, the PSCH:PSC decrements by one. One timer clock (output of the timer prescaler) cycle after the PSCH:PSC reaches 0, the PSCH:PSC is loaded with the contents of the TDDRH:TDDR, and the timer counter register (TIMH:TIM) decrements by one. The PSCH:PSC is also reloaded whenever the timer reload bit (TRB) is set by software. The PSCH:PSC can be checked by reading the register, but it cannot be set directly. It must get its value from the timer divide-down register (TDDRH:TDDR). At reset, the PSCH:PSC is set to 0. Reset type: SYSRSn
7-0	TDDR	R/W	0h	CPU-Timer Divide-Down. Every (TDDRH:TDDR + 1) timer clock source cycles, the timer counter register (TIMH:TIM) decrements by one. At reset, the TDDRH:TDDR bits are cleared to 0. To increase the overall timer count by an integer factor, write this factor minus one to the TDDRH:TDDR bits. When the prescaler counter (PSCH:PSC) value is 0, one timer clock source cycle later, the contents of the TDDRH:TDDR reload the PSCH:PSC, and the TIMH:TIM decrements by one. TDDRH:TDDR also reloads the PSCH:PSC whenever the timer reload bit (TRB) is set by software. Reset type: SYSRSn

3.16.8.5 TPRH Register (Offset = 7h) [Reset = 0h]

TPRH is shown in Figure 3-178 and described in Table 3-188.

Return to the Summary Table.

CPU-Timer, Prescale Register High

Figure 3-178. TPRH Register

15	14	13	12	11	10	9	8
PSCH							
R-0h							
7	6	5	4	3	2	1	0
TDDRH							
R/W-0h							

Table 3-188. TPRH Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	PSCH	R	0h	See description of TIMERxTPR. Reset type: SYSRSn
7-0	TDDRH	R/W	0h	See description of TIMERxTPR. Reset type: SYSRSn

3.16.9 DEV_CFG_REGS Registers

Table 3-189 lists the memory-mapped registers for the DEV_CFG_REGS registers. All register offset addresses not listed in Table 3-189 should be considered as reserved locations and the register contents should not be modified.

Table 3-189. DEV_CFG_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	DEVCFGLOCK1	Lock bit for DEVCFG registers	EALLOW	Go
2h	DEVCFGLOCK2	Lock bit for DEVCFG registers	EALLOW	Go
8h	PARTIDL	Lower 32-bit of Device PART Identification Number		Go
Ah	PARTIDH	Upper 32-bit of Device PART Identification Number		Go
Ch	REVID	Device Revision Number		Go
60h	PERCNF1	Peripheral Configuration register		Go
74h	FUSEERR	e-Fuse error Status register		Go
82h	SOFTPRES0	Processing Block Software Reset register	EALLOW	Go
84h	SOFTPRES1	EMIF Software Reset register	EALLOW	Go
86h	SOFTPRES2	Peripheral Software Reset register	EALLOW	Go
88h	SOFTPRES3	Peripheral Software Reset register	EALLOW	Go
8Ah	SOFTPRES4	Peripheral Software Reset register	EALLOW	Go
8Eh	SOFTPRES6	Peripheral Software Reset register	EALLOW	Go
90h	SOFTPRES7	Peripheral Software Reset register	EALLOW	Go
92h	SOFTPRES8	Peripheral Software Reset register	EALLOW	Go
94h	SOFTPRES9	Peripheral Software Reset register	EALLOW	Go
96h	SOFTPRES10	Peripheral Software Reset register	EALLOW	Go
98h	SOFTPRES11	Peripheral Software Reset register	EALLOW	Go
9Ch	SOFTPRES13	Peripheral Software Reset register	EALLOW	Go
9Eh	SOFTPRES14	Peripheral Software Reset register	EALLOW	Go
A2h	SOFTPRES16	Peripheral Software Reset register	EALLOW	Go
A4h	SOFTPRES17	Reserved Peripheral Software Reset register	EALLOW	Go
A6h	SOFTPRES18	Reserved Peripheral Software Reset register	EALLOW	Go
AAh	SOFTPRES20	Peripheral Software Reset register	EALLOW	Go
ACh	SOFTPRES21	Peripheral Software Reset register	EALLOW	Go
B0h	SOFTPRES23	Peripheral Software Reset register	EALLOW	Go
D6h	CPUSEL0	CPU Select register for common peripherals	EALLOW	Go
D8h	CPUSEL1	CPU Select register for common peripherals	EALLOW	Go
DAh	CPUSEL2	CPU Select register for common peripherals	EALLOW	Go
DEh	CPUSEL4	CPU Select register for common peripherals	EALLOW	Go
E0h	CPUSEL5	CPU Select register for common peripherals	EALLOW	Go
E2h	CPUSEL6	CPU Select register for common peripherals	EALLOW	Go
E4h	CPUSEL7	CPU Select register for common peripherals	EALLOW	Go
E6h	CPUSEL8	CPU Select register for common peripherals	EALLOW	Go
E8h	CPUSEL9	CPU Select register for common peripherals	EALLOW	Go
ECh	CPUSEL11	CPU Select register for common peripherals	EALLOW	Go
EEh	CPUSEL12	CPU Select register for common peripherals	EALLOW	Go
F2h	CPUSEL14	CPU Select register for common peripherals	EALLOW	Go
F4h	CPUSEL15	CPU Select register for common peripherals	EALLOW	Go

Table 3-189. DEV_CFG_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
F6h	CPUSEL16	CPU Select register for common peripherals	EALLOW	Go
FAh	CPUSEL18	CPU Select register for common peripherals	EALLOW	Go
108h	CPUSEL25	CPU Select register for common peripherals	EALLOW	Go
122h	CPU2RESCTL	CPU2 Reset Control Register	EALLOW	Go
124h	RSTSTAT	Reset Status register for secondary C28x CPUs		Go
125h	LPMSTAT	LPM Status Register for secondary C28x CPUs		Go
19Ah	USBTYPE	Configures USB Type for the device	EALLOW	Go
19Bh	ECAPTYPE	Configures ECAP Type for the device	EALLOW	Go
19Ch	SDFMTYPE	Configures SDFM Type for the device	EALLOW	Go
19Eh	MEMMAPTYPE	Configures Memory Map Type for the device	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-190 shows the codes that are used for access types in this section.

Table 3-190. DEV_CFG_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
WOnce	W Once	Write Write once
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.9.1 DEVCFGLOCK1 Register (Offset = 0h) [Reset = 0h]

DEVCFGLOCK1 is shown in Figure 3-179 and described in Table 3-191.

Return to the Summary Table.

Lock bit for DEVCFG registers
The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed
Note:
Any SOnce bit in this register, once set can only be cleared through a CPU1.SYSRSn. Write of 0 to any bit of this register has no effect

Figure 3-179. DEVCFGLOCK1 Register

31	30	29	28	27	26	25	24
RESERVED						CPUSEL25	RESERVED
R-0-0h						R/WSonce-0h	R-0-0h
23	22	21	20	19	18	17	16
RESERVED					CPUSEL18	RESERVED	CPUSEL16
R-0-0h					R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
15	14	13	12	11	10	9	8
CPUSEL15	CPUSEL14	RESERVED	CPUSEL12	CPUSEL11	RESERVED	CPUSEL9	CPUSEL8
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
7	6	5	4	3	2	1	0
CPUSEL7	CPUSEL6	CPUSEL5	CPUSEL4	RESERVED	CPUSEL2	CPUSEL1	CPUSEL0
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 3-191. DEVCFGLOCK1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R-0	0h	Reserved
25	CPUSEL25	R/WSonce	0h	Lock bit for CPUSEL25 register: 0: Register is not locked 1: Register is locked. Reset type: CPU1.SYSRSn
24-19	RESERVED	R-0	0h	Reserved
18	CPUSEL18	R/WSonce	0h	Lock bit for CPUSEL18 register: 0: Register is not locked 1: Register is locked. Reset type: CPU1.SYSRSn
17	RESERVED	R/WSonce	0h	Reserved
16	CPUSEL16	R/WSonce	0h	Lock bit for CPUSEL16 register: 0: Register is not locked 1: Register is locked. Reset type: CPU1.SYSRSn
15	CPUSEL15	R/WSonce	0h	Lock bit for CPUSEL15 register: 0: Register is not locked 1: Register is locked. Reset type: CPU1.SYSRSn
14	CPUSEL14	R/WSonce	0h	Lock bit for CPUSEL14 register: 0: Register is not locked 1: Register is locked. Reset type: CPU1.SYSRSn
13	RESERVED	R/WSonce	0h	Reserved

Table 3-191. DEVCFGLOCK1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	CPUSEL12	R/WSonce	0h	Lock bit for CPUSEL12 register: 0: Register is not locked 1: Register is locked. Reset type: CPU1.SYSRSn
11	CPUSEL11	R/WSonce	0h	Lock bit for CPUSEL11 register: 0: Register is not locked 1: Register is locked. Reset type: CPU1.SYSRSn
10	RESERVED	R/WSonce	0h	Reserved
9	CPUSEL9	R/WSonce	0h	Lock bit for CPUSEL9 register: 0: Register is not locked 1: Register is locked. Reset type: CPU1.SYSRSn
8	CPUSEL8	R/WSonce	0h	Lock bit for CPUSEL8 register: 0: Register is not locked 1: Register is locked. Reset type: CPU1.SYSRSn
7	CPUSEL7	R/WSonce	0h	Lock bit for CPUSEL7 register: 0: Register is not locked 1: Register is locked. Reset type: CPU1.SYSRSn
6	CPUSEL6	R/WSonce	0h	Lock bit for CPUSEL6 register: 0: Register is not locked 1: Register is locked. Reset type: CPU1.SYSRSn
5	CPUSEL5	R/WSonce	0h	Lock bit for CPUSEL5 register: 0: Register is not locked 1: Register is locked. Reset type: CPU1.SYSRSn
4	CPUSEL4	R/WSonce	0h	Lock bit for CPUSEL4 register: 0: Register is not locked 1: Register is locked. Reset type: CPU1.SYSRSn
3	RESERVED	R/WSonce	0h	Reserved
2	CPUSEL2	R/WSonce	0h	Lock bit for CPUSEL2 register: 0: Register is not locked 1: Register is locked. Reset type: CPU1.SYSRSn
1	CPUSEL1	R/WSonce	0h	Lock bit for CPUSEL1 register: 0: Register is not locked 1: Register is locked. Reset type: CPU1.SYSRSn
0	CPUSEL0	R/WSonce	0h	Lock bit for CPUSEL0 register: 0: Register is not locked 1: Register is locked. Reset type: CPU1.SYSRSn

3.16.9.2 DEVCFGLOCK2 Register (Offset = 2h) [Reset = 0h]

DEVCFGLOCK2 is shown in Figure 3-180 and described in Table 3-192.

Return to the Summary Table.

Lock bit for DEVCFG registers
The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed
Note:
Any SOnce bit in this register, once set can only be cleared through a CPU1.SYSRSn. Write of 0 to any bit of this register has no effect

Figure 3-180. DEVCFGLOCK2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	RESERVED
R-0-0h						R/WSonce-0h	R/WSonce-0h

Table 3-192. DEVCFGLOCK2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R-0	0h	Reserved
1	RESERVED	R/WSonce	0h	Reserved
0	RESERVED	R/WSonce	0h	Reserved

3.16.9.3 PARTIDL Register (Offset = 8h) [Reset = X]

PARTIDL is shown in Figure 3-181 and described in Table 3-193.

Return to the Summary Table.

Lower 32-bit of Device PART Identification Number

Figure 3-181. PARTIDL Register

31	30	29	28	27	26	25	24
PARTID_FORMAT_REVISION				RESERVED			
R/W-2h				R-0h			
23	22	21	20	19	18	17	16
FLASH_SIZE							
R/W-7h							
15	14	13	12	11	10	9	8
RESERVED	INSTASPIN		RESERVED	RESERVED	PIN_COUNT		
R-0h	R/W-1h		R/W-0h	R/W-0h	R/W-X		
7	6	5	4	3	2	1	0
QUAL		RESERVED	RESERVED		RESERVED		
R/W-X		R-0h	R/W-0h		R/W-0h		

Table 3-193. PARTIDL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	PARTID_FORMAT_REVISION	R/W	2h	Loaded from OTP by boot ROM 0xF = invalid PART ID (assume max config in flash tools) 0x0 = first revision of part id format 0x1 = second revision of format 0x2 = third revision of part id format Reset type: PORESETn
27-24	RESERVED	R	0h	Reserved
23-16	FLASH_SIZE	R/W	7h	0x8 - 1024KB 0x7 - 512KB 0x6 - 256KB 0x5 - 128KB 0x4 - 64KB 0x3 - 32KB Reset type: PORESETn
15	RESERVED	R	0h	Reserved
14-13	INSTASPIN	R/W	1h	0 = InstaSPIN Motion (Fast+Spin) 1 = InstaSPIN-FOC 2 = NONE 3 = NONE Reset type: PORESETn
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10-8	PIN_COUNT	R/W	X	0 = 56 pin (Reserved) 1 = 64 pin (Q100) 2 = 64 pin 3 = 80 pin 4 = 48 pin 5 = 100 pin (Reserved) 6 = 176 pin 7 = 337 pin Reset type: PORESETn

Table 3-193. PARTIDL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	QUAL	R/W	X	0 = Engineering sample.(TMX) 1 = Pilot production (TMP) 2 = Fully qualified (TMS) Reset type: PORESETn
5	RESERVED	R	0h	Reserved
4-3	RESERVED	R/W	0h	Reserved
2-0	RESERVED	R/W	0h	Reserved

3.16.9.4 PARTIDH Register (Offset = Ah) [Reset = X]

PARTIDH is shown in Figure 3-182 and described in Table 3-194.

Return to the Summary Table.

Upper 32-bit of Device PART Identification Number

Figure 3-182. PARTIDH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DEVICE_CLASS_ID								PARTNO							
R-3h								R/W-X							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FAMILY								RESERVED				RESERVED			
R/W-X								R-0h				R-0h			

Table 3-194. PARTIDH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	DEVICE_CLASS_ID	R	3h	Device Class ID Reset type: PORESETn
23-16	PARTNO	R/W	X	Part Number Designator 0xFF - F28388x 0xFD - F28386x 0xFB - F28384x Reset type: PORESETn
15-8	FAMILY	R/W	X	Device Family 0x3 - DUAL CORE 0x4 - SINGLE CORE Other values Reserved Reset type: PORESETn Reset type: PORESETn
7-4	RESERVED	R	0h	Reserved
3-0	RESERVED	R	0h	Reserved

3.16.9.5 REVID Register (Offset = Ch) [Reset = 0h]

REVID is shown in Figure 3-183 and described in Table 3-195.

Return to the Summary Table.

Device Revision Number

Figure 3-183. REVID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																REVID															
R-0-0h																R-0h															

Table 3-195. REVID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-0	REVID	R	0h	These 32-bits specify the silicon revision. See your device specific datasheet for details. Reset type: N/A

3.16.9.6 PERCNF1 Register (Offset = 60h) [Reset = 0h]

PERCNF1 is shown in Figure 3-184 and described in Table 3-196.

Return to the Summary Table.

Peripheral Configuration register

Figure 3-184. PERCNF1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED						RESERVED	USB_A_PHY
R-0-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				ADC_D_MODE	ADC_C_MODE	ADC_B_MODE	ADC_A_MODE
R-0-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-196. PERCNF1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R-0	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	USB_A_PHY	R/W	0h	Internal PHY is present present or not for the USB_A module: 0: Internal USB PHY Module is not present 1: Internal USB PHY Module is present. Reset type: PORESETn
15-4	RESERVED	R-0	0h	Reserved
3	ADC_D_MODE	R/W	0h	0: 16-bit or 12-bit configurable in software 1: Only 12-bit operation available Reset type: PORESETn
2	ADC_C_MODE	R/W	0h	0: 16-bit or 12-bit configurable in software 1: Only 12-bit operation available Reset type: PORESETn
1	ADC_B_MODE	R/W	0h	0: 16-bit or 12-bit configurable in software 1: Only 12-bit operation available Reset type: PORESETn
0	ADC_A_MODE	R/W	0h	0: 16-bit or 12-bit configurable in software 1: Only 12-bit operation available Reset type: PORESETn

3.16.9.7 FUSEERR Register (Offset = 74h) [Reset = 0h]

FUSEERR is shown in Figure 3-185 and described in Table 3-197.

Return to the Summary Table.

e-Fuse error Status register

Figure 3-185. FUSEERR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED										ERR	ALERR				
R-0-0h										R-0h	R-0h				

Table 3-197. FUSEERR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-6	RESERVED	R-0	0h	Reserved
5	ERR	R	0h	Efuse Self Test Error Status set by hardware after fuse self test completes, in case of self test error 0: No error during fuse self test 1: Fuse self test error Reset type: XRSn
4-0	ALERR	R	0h	Efuse Autoload Error Status set by hardware after fuse auto load completes 00000: No error in auto load Other: Non zero value indicates error in autoload Note: [1] 10101 means a single-bit error during autoload. Since this gets corrected by the ECC mechanism, this value shouldn't be treated as an error condition. Reset type: XRSn

3.16.9.8 SOFTPRES0 Register (Offset = 82h) [Reset = 0h]

SOFTPRES0 is shown in Figure 3-186 and described in Table 3-198.

Return to the Summary Table.

Processing Block Software Reset register
When bits in this register are set, the respective module is in reset. All design data is lost and the module registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-186. SOFTPRES0 Register

31	30	29	28	27	26	25	24
RESERVED						CPU2_ERAD	CPU1_ERAD
R-0-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED					RESERVED	CPU2_CLA1BGCRC	CPU2_CPUBGCRC
R-0-0h					R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	CPU1_CLA1BGCRC	CPU1_CPUBGCRC	RESERVED				
R/W-0h	R/W-0h	R/W-0h	R-0-0h				
7	6	5	4	3	2	1	0
RESERVED				RESERVED	CPU2_CLA1	RESERVED	CPU1_CLA1
R-0-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-198. SOFTPRES0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R-0	0h	Reserved
25	CPU2_ERAD	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
24	CPU1_ERAD	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
23-19	RESERVED	R-0	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	CPU2_CLA1BGCRC	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
16	CPU2_CPUBGCRC	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
15	RESERVED	R/W	0h	Reserved
14	CPU1_CLA1BGCRC	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
13	CPU1_CPUBGCRC	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
12-4	RESERVED	R-0	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	CPU2_CLA1	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

Table 3-198. SOFTPRES0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	RESERVED	R/W	0h	Reserved
0	CPU1_CLA1	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

3.16.9.9 SOFTPRES1 Register (Offset = 84h) [Reset = 0h]

SOFTPRES1 is shown in Figure 3-187 and described in Table 3-199.

Return to the Summary Table.

Processing Block Software Reset register
When bits in this register are set, the respective module is in reset. All design data is lost and the module registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-187. SOFTPRES1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						EMIF2	EMIF1
R-0-0h						R/W-0h	R/W-0h

Table 3-199. SOFTPRES1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-2	RESERVED	R-0	0h	Reserved
1	EMIF2	R/W	0h	When this bit is set, only the control logic of the respective EMIF2 is reset. It does not reset the internal registers except the Total Access register and the Total Activate register. Refer to EMIF chapter for more details on the EMIF SOFTRESET feature. This bit must be manually cleared after being set. 1: EMIF2 is under SOFTRESET 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
0	EMIF1	R/W	0h	When this bit is set, only the control logic of the respective EMIF1 is reset. It does not reset the internal registers except the Total Access register and the Total Activate register. Refer to EMIF chapter for more details on the EMIF SOFTRESET feature. This bit must be manually cleared after being set. 1: EMIF1 is under SOFTRESET 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

3.16.9.10 SOFTPRES2 Register (Offset = 86h) [Reset = 0h]

SOFTPRES2 is shown in Figure 3-188 and described in Table 3-200.

Return to the Summary Table.

Processing Block Software Reset register
When bits in this register are set, the respective module is in reset. All design data is lost and the module registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-188. SOFTPRES2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
EPWM16	EPWM15	EPWM14	EPWM13	EPWM12	EPWM11	EPWM10	EPWM9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
EPWM8	EPWM7	EPWM6	EPWM5	EPWM4	EPWM3	EPWM2	EPWM1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-200. SOFTPRES2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15	EPWM16	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
14	EPWM15	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
13	EPWM14	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
12	EPWM13	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
11	EPWM12	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
10	EPWM11	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
9	EPWM10	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
8	EPWM9	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
7	EPWM8	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

Table 3-200. SOFTPRES2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	EPWM7	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
5	EPWM6	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
4	EPWM5	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
3	EPWM4	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
2	EPWM3	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
1	EPWM2	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
0	EPWM1	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

3.16.9.11 SOFTPRES3 Register (Offset = 88h) [Reset = 0h]

SOFTPRES3 is shown in Figure 3-189 and described in Table 3-201.

Return to the Summary Table.

Processing Block Software Reset register
When bits in this register are set, the respective module is in reset. All design data is lost and the module registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-189. SOFTPRES3 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED	ECAP7	ECAP6	ECAP5	ECAP4	ECAP3	ECAP2	ECAP1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-201. SOFTPRES3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	RESERVED	R-0	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	ECAP7	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
5	ECAP6	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
4	ECAP5	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
3	ECAP4	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
2	ECAP3	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
1	ECAP2	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
0	ECAP1	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

3.16.9.12 SOFTPRES4 Register (Offset = 8Ah) [Reset = 0h]

SOFTPRES4 is shown in Figure 3-190 and described in Table 3-202.

Return to the Summary Table.

Processing Block Software Reset register
When bits in this register are set, the respective module is in reset. All design data is lost and the module registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-190. SOFTPRES4 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	EQEP3	EQEP2	EQEP1
R-0-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-202. SOFTPRES4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-4	RESERVED	R-0	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	EQEP3	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
1	EQEP2	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
0	EQEP1	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

3.16.9.13 SOFTPRES6 Register (Offset = 8Eh) [Reset = 0h]

SOFTPRES6 is shown in Figure 3-191 and described in Table 3-203.

Return to the Summary Table.

Processing Block Software Reset register
When bits in this register are set, the respective module is in reset. All design data is lost and the module registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-191. SOFTPRES6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	SD2	SD1
R-0-0h								R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-203. SOFTPRES6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	RESERVED	R-0	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	SD2	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
0	SD1	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

3.16.9.14 SOFTPRES7 Register (Offset = 90h) [Reset = 0h]

SOFTPRES7 is shown in Figure 3-192 and described in Table 3-204.

Return to the Summary Table.

Processing Block Software Reset register
When bits in this register are set, the respective module is in reset. All design data is lost and the module registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-192. SOFTPRES7 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				SCI_D	SCI_C	SCI_B	SCI_A
R-0-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-204. SOFTPRES7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-4	RESERVED	R-0	0h	Reserved
3	SCI_D	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
2	SCI_C	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
1	SCI_B	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
0	SCI_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

3.16.9.15 SOFTPRES8 Register (Offset = 92h) [Reset = 0h]

SOFTPRES8 is shown in Figure 3-193 and described in Table 3-205.

Return to the Summary Table.

Processing Block Software Reset register
When bits in this register are set, the respective module is in reset. All design data is lost and the module registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-193. SOFTPRES8 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED						RESERVED	RESERVED
R-0-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				SPI_D	SPI_C	SPI_B	SPI_A
R-0-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-205. SOFTPRES8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R-0	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15-4	RESERVED	R-0	0h	Reserved
3	SPI_D	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
2	SPI_C	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
1	SPI_B	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
0	SPI_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

3.16.9.16 SOFTPRES9 Register (Offset = 94h) [Reset = 0h]

SOFTPRES9 is shown in Figure 3-194 and described in Table 3-206.

Return to the Summary Table.

Processing Block Software Reset register
When bits in this register are set, the respective module is in reset. All design data is lost and the module registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-194. SOFTPRES9 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						I2C_B	I2C_A
R-0-0h						R/W-0h	R/W-0h

Table 3-206. SOFTPRES9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-2	RESERVED	R-0	0h	Reserved
1	I2C_B	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
0	I2C_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

3.16.9.17 SOFTPRES10 Register (Offset = 96h) [Reset = 0h]

SOFTPRES10 is shown in Figure 3-195 and described in Table 3-207.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-195. SOFTPRES10 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED			MCAN_A	RESERVED	RESERVED	CAN_B	CAN_A
R-0-0h			R/W-0h	R/W-0h	R-0-0h	R/W-0h	R/W-0h

Table 3-207. SOFTPRES10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R-0	0h	Reserved
4	MCAN_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R-0	0h	Reserved
1	CAN_B	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
0	CAN_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

3.16.9.18 SOFTPRES11 Register (Offset = 98h) [Reset = 0h]

SOFTPRES11 is shown in Figure 3-196 and described in Table 3-208.

Return to the Summary Table.

Processing Block Software Reset register
When bits in this register are set, the respective module is in reset. All design data is lost and the module registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-196. SOFTPRES11 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED						RESERVED	USB_A
R-0-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						McBSP_B	McBSP_A
R-0-0h						R/W-0h	R/W-0h

Table 3-208. SOFTPRES11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R-0	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	USB_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
15-2	RESERVED	R-0	0h	Reserved
1	McBSP_B	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
0	McBSP_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

3.16.9.19 SOFTPRES13 Register (Offset = 9Ch) [Reset = 0h]

SOFTPRES13 is shown in Figure 3-197 and described in Table 3-209.

Return to the Summary Table.

Processing Block Software Reset register
When bits in this register are set, the respective module is in reset. All design data is lost and the module registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-197. SOFTPRES13 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				ADC_D	ADC_C	ADC_B	ADC_A
R-0-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-209. SOFTPRES13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-4	RESERVED	R-0	0h	Reserved
3	ADC_D	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
2	ADC_C	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
1	ADC_B	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
0	ADC_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

3.16.9.20 SOFTPRES14 Register (Offset = 9Eh) [Reset = 0h]

SOFTPRES14 is shown in Figure 3-198 and described in Table 3-210.

Return to the Summary Table.

Processing Block Software Reset register
When bits in this register are set, the respective module is in reset. All design data is lost and the module registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-198. SOFTPRES14 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
CMPSS8	CMPSS7	CMPSS6	CMPSS5	CMPSS4	CMPSS3	CMPSS2	CMPSS1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-210. SOFTPRES14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	RESERVED	R-0	0h	Reserved
7	CMPSS8	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
6	CMPSS7	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
5	CMPSS6	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
4	CMPSS5	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
3	CMPSS4	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
2	CMPSS3	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
1	CMPSS2	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
0	CMPSS1	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

3.16.9.21 SOFTPRES16 Register (Offset = A2h) [Reset = 0h]

SOFTPRES16 is shown in Figure 3-199 and described in Table 3-211.

Return to the Summary Table.

Processing Block Software Reset register
When bits in this register are set, the respective module is in reset. All design data is lost and the module registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-199. SOFTPRES16 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED				RESERVED	DAC_C	DAC_B	DAC_A
R-0-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED							
R-0-0h							

Table 3-211. SOFTPRES16 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R-0	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	DAC_C	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
17	DAC_B	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
16	DAC_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
15-0	RESERVED	R-0	0h	Reserved

3.16.9.22 SOFTPRES17 Register (Offset = A4h) [Reset = 0h]

SOFTPRES17 is shown in Figure 3-200 and described in Table 3-212.

Return to the Summary Table.

Processing Block Software Reset register
When bits in this register are set, the respective module is in reset. All design data is lost and the module registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-200. SOFTPRES17 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
CLB8	CLB7	CLB6	CLB5	CLB4	CLB3	CLB2	CLB1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-212. SOFTPRES17 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R-0	0h	Reserved
7	CLB8	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
6	CLB7	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
5	CLB6	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
4	CLB5	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
3	CLB4	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
2	CLB3	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
1	CLB2	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
0	CLB1	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

3.16.9.23 SOFTPRES18 Register (Offset = A6h) [Reset = 0h]

SOFTPRES18 is shown in Figure 3-201 and described in Table 3-213.

Return to the Summary Table.

Processing Block Software Reset register
When bits in this register are set, the respective module is in reset. All design data is lost and the module registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-201. SOFTPRES18 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
FSIRX_H	FSIRX_G	FSIRX_F	FSIRX_E	FSIRX_D	FSIRX_C	FSIRX_B	FSIRX_A
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	FSITX_B	FSITX_A
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-213. SOFTPRES18 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R-0	0h	Reserved
23	FSIRX_H	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
22	FSIRX_G	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
21	FSIRX_F	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
20	FSIRX_E	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
19	FSIRX_D	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
18	FSIRX_C	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
17	FSIRX_B	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
16	FSIRX_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
15-8	RESERVED	R/W	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved

Table 3-213. SOFTPRES18 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	FSITX_B	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
0	FSITX_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

3.16.9.24 SOFTPRES20 Register (Offset = AAh) [Reset = 0h]

SOFTPRES20 is shown in Figure 3-202 and described in Table 3-214.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-202. SOFTPRES20 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	PMBUS_A
R-0-0h						R/W-0h	R/W-0h

Table 3-214. SOFTPRES20 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R-0	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	PMBUS_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

3.16.9.25 SOFTPRES21 Register (Offset = ACh) [Reset = 0h]

SOFTPRES21 is shown in Figure 3-203 and described in Table 3-215.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-203. SOFTPRES21 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED					DCC2	DCC1	DCC0
R-0-0h					R/W-0h	R/W-0h	R/W-0h

Table 3-215. SOFTPRES21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R-0	0h	Reserved
2	DCC2	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
1	DCC1	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn
0	DCC0	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

3.16.9.26 SOFTPRES23 Register (Offset = B0h) [Reset = 1h]

SOFTPRES23 is shown in Figure 3-204 and described in Table 3-216.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-204. SOFTPRES23 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ETHERCAT
R-0h							R/W-1h

Table 3-216. SOFTPRES23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	ETHERCAT	R/W	1h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: CPU1.SYSRSn

3.16.9.27 CPUSEL0 Register (Offset = D6h) [Reset = 0h]

CPUSEL0 is shown in Figure 3-205 and described in Table 3-217.

Return to the Summary Table.

CPU Select register for common peripherals
This register must be configured prior to enabling the peripheral clocks.
The clock for each peripheral is derived from the selected CPU subsystem. The clock mux controlled by this register is not glitch-free, therefore the CPUSELx register must be configured before the PCLKCRx register.
The reset for each peripheral is also driven from the selected CPU.

Figure 3-205. CPUSEL0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
EPWM16	EPWM15	EPWM14	EPWM13	EPWM12	EPWM11	EPWM10	EPWM9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
EPWM8	EPWM7	EPWM6	EPWM5	EPWM4	EPWM3	EPWM2	EPWM1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-217. CPUSEL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15	EPWM16	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
14	EPWM15	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
13	EPWM14	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
12	EPWM13	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
11	EPWM12	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
10	EPWM11	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
9	EPWM10	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
8	EPWM9	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
7	EPWM8	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn

Table 3-217. CPUSEL0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	EPWM7	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
5	EPWM6	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
4	EPWM5	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
3	EPWM4	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
2	EPWM3	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
1	EPWM2	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
0	EPWM1	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn

3.16.9.28 CPUSEL1 Register (Offset = D8h) [Reset = 0h]

CPUSEL1 is shown in Figure 3-206 and described in Table 3-218.

Return to the Summary Table.

CPU Select register for common peripherals
This register must be configured prior to enabling the peripheral clocks.
The clock for each peripheral is derived from the selected CPU subsystem. The clock mux controlled by this register is not glitch-free, therefore the CPUSELx register must be configured before the PCLKCRx register.
The reset for each peripheral is also driven from the selected CPU.

Figure 3-206. CPUSEL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED	ECAP7	ECAP6	ECAP5	ECAP4	ECAP3	ECAP2	ECAP1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-218. CPUSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	RESERVED	R-0	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	ECAP7	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
5	ECAP6	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
4	ECAP5	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
3	ECAP4	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
2	ECAP3	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
1	ECAP2	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
0	ECAP1	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn

3.16.9.29 CPUSEL2 Register (Offset = DAh) [Reset = 0h]

CPUSEL2 is shown in Figure 3-207 and described in Table 3-219.

Return to the Summary Table.

CPU Select register for common peripherals
This register must be configured prior to enabling the peripheral clocks.
The clock for each peripheral is derived from the selected CPU subsystem. The clock mux controlled by this register is not glitch-free, therefore the CPUSELx register must be configured before the PCLKCRx register.
The reset for each peripheral is also driven from the selected CPU.

Figure 3-207. CPUSEL2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	EQEP3	EQEP2	EQEP1
R-0-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-219. CPUSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-4	RESERVED	R-0	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	EQEP3	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
1	EQEP2	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
0	EQEP1	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn

3.16.9.30 CPUSEL4 Register (Offset = DEh) [Reset = 0h]

CPUSEL4 is shown in Figure 3-208 and described in Table 3-220.

Return to the Summary Table.

CPU Select register for common peripherals
This register must be configured prior to enabling the peripheral clocks.
The clock for each peripheral is derived from the selected CPU subsystem. The clock mux controlled by this register is not glitch-free, therefore the CPUSELx register must be configured before the PCLKCRx register.
The reset for each peripheral is also driven from the selected CPU.

Figure 3-208. CPUSEL4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	SD2	SD1
R-0-0h								R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-220. CPUSEL4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	RESERVED	R-0	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	SD2	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
0	SD1	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn

3.16.9.31 CPUSEL5 Register (Offset = E0h) [Reset = 0h]

CPUSEL5 is shown in Figure 3-209 and described in Table 3-221.

Return to the Summary Table.

CPU Select register for common peripherals
This register must be configured prior to enabling the peripheral clocks.
The clock for each peripheral is derived from the selected CPU subsystem. The clock mux controlled by this register is not glitch-free, therefore the CPUSELx register must be configured before the PCLKCRx register.
The reset for each peripheral is also driven from the selected CPU.

Figure 3-209. CPUSEL5 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				SCI_D	SCI_C	SCI_B	SCI_A
R-0-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-221. CPUSEL5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-4	RESERVED	R-0	0h	Reserved
3	SCI_D	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
2	SCI_C	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
1	SCI_B	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
0	SCI_A	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn

3.16.9.32 CPUSEL6 Register (Offset = E2h) [Reset = 0h]

CPUSEL6 is shown in Figure 3-210 and described in Table 3-222.

Return to the Summary Table.

CPU Select register for common peripherals
This register must be configured prior to enabling the peripheral clocks.
The clock for each peripheral is derived from the selected CPU subsystem. The clock mux controlled by this register is not glitch-free, therefore the CPUSELx register must be configured before the PCLKCRx register.
The reset for each peripheral is also driven from the selected CPU.

Figure 3-210. CPUSEL6 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED						RESERVED	RESERVED
R-0-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				SPI_D	SPI_C	SPI_B	SPI_A
R-0-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-222. CPUSEL6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R-0	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15-4	RESERVED	R-0	0h	Reserved
3	SPI_D	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
2	SPI_C	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
1	SPI_B	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
0	SPI_A	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn

3.16.9.33 CPUSEL7 Register (Offset = E4h) [Reset = 0h]

CPUSEL7 is shown in Figure 3-211 and described in Table 3-223.

Return to the Summary Table.

CPU Select register for common peripherals
This register must be configured prior to enabling the peripheral clocks.
The clock for each peripheral is derived from the selected CPU subsystem. The clock mux controlled by this register is not glitch-free, therefore the CPUSELx register must be configured before the PCLKCRx register.
The reset for each peripheral is also driven from the selected CPU.

Figure 3-211. CPUSEL7 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						I2C_B	I2C_A
R-0-0h						R/W-0h	R/W-0h

Table 3-223. CPUSEL7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-2	RESERVED	R-0	0h	Reserved
1	I2C_B	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
0	I2C_A	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn

3.16.9.34 CPUSEL8 Register (Offset = E6h) [Reset = 0h]

CPUSEL8 is shown in Figure 3-212 and described in Table 3-224.

Return to the Summary Table.

CPU Select register for common peripherals
This register must be configured prior to enabling the peripheral clocks.
The clock for each peripheral is derived from the selected CPU subsystem. The clock mux controlled by this register is not glitch-free, therefore the CPUSELx register must be configured before the PCLKCRx register.
The reset for each peripheral is also driven from the selected CPU.

Figure 3-212. CPUSEL8 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED			MCAN_A	RESERVED	RESERVED	CAN_B	CAN_A
R-0-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-224. CPUSEL8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-5	RESERVED	R-0	0h	Reserved
4	MCAN_A	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	CAN_B	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
0	CAN_A	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn

3.16.9.35 CPUSEL9 Register (Offset = E8h) [Reset = 0h]

CPUSEL9 is shown in Figure 3-213 and described in Table 3-225.

Return to the Summary Table.

CPU Select register for common peripherals
This register must be configured prior to enabling the peripheral clocks.
The clock for each peripheral is derived from the selected CPU subsystem. The clock mux controlled by this register is not glitch-free, therefore the CPUSELx register must be configured before the PCLKCRx register.
The reset for each peripheral is also driven from the selected CPU.

Figure 3-213. CPUSEL9 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						McBSP_B	McBSP_A
R-0-0h						R/W-0h	R/W-0h

Table 3-225. CPUSEL9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-2	RESERVED	R-0	0h	Reserved
1	McBSP_B	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
0	McBSP_A	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn

3.16.9.36 CPUSEL11 Register (Offset = ECh) [Reset = 0h]

CPUSEL11 is shown in Figure 3-214 and described in Table 3-226.

Return to the Summary Table.

CPU Select register for common peripherals
This register must be configured prior to enabling the peripheral clocks.
The clock for each peripheral is derived from the selected CPU subsystem. The clock mux controlled by this register is not glitch-free, therefore the CPUSELx register must be configured before the PCLKCRx register.
The reset for each peripheral is also driven from the selected CPU.

Figure 3-214. CPUSEL11 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				ADC_D	ADC_C	ADC_B	ADC_A
R-0-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-226. CPUSEL11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-4	RESERVED	R-0	0h	Reserved
3	ADC_D	R/W	0h	These CPUSEL bits affect the ownership of only ADC Configuration registers by CPU1 or CPU2. ADC result registers are readable from all masters without any CPUSEL dependency. 0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
2	ADC_C	R/W	0h	These CPUSEL bits affect the ownership of only ADC Configuration registers by CPU1 or CPU2. ADC result registers are readable from all masters without any CPUSEL dependency. 0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
1	ADC_B	R/W	0h	These CPUSEL bits affect the ownership of only ADC Configuration registers by CPU1 or CPU2. ADC result registers are readable from all masters without any CPUSEL dependency. 0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
0	ADC_A	R/W	0h	These CPUSEL bits affect the ownership of only ADC Configuration registers by CPU1 or CPU2. ADC result registers are readable from all masters without any CPUSEL dependency. 0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn

3.16.9.37 CPUSEL12 Register (Offset = EEh) [Reset = 0h]

CPUSEL12 is shown in Figure 3-215 and described in Table 3-227.

Return to the Summary Table.

CPU Select register for common peripherals
This register must be configured prior to enabling the peripheral clocks.
The clock for each peripheral is derived from the selected CPU subsystem. The clock mux controlled by this register is not glitch-free, therefore the CPUSELx register must be configured before the PCLKCRx register.
The reset for each peripheral is also driven from the selected CPU.

Figure 3-215. CPUSEL12 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
CMPSS8	CMPSS7	CMPSS6	CMPSS5	CMPSS4	CMPSS3	CMPSS2	CMPSS1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-227. CPUSEL12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	RESERVED	R-0	0h	Reserved
7	CMPSS8	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
6	CMPSS7	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
5	CMPSS6	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
4	CMPSS5	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
3	CMPSS4	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
2	CMPSS3	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
1	CMPSS2	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
0	CMPSS1	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn

3.16.9.38 CPUSEL14 Register (Offset = F2h) [Reset = 0h]

CPUSEL14 is shown in Figure 3-216 and described in Table 3-228.

Return to the Summary Table.

CPU Select register for common peripherals
This register must be configured prior to enabling the peripheral clocks.
The clock for each peripheral is derived from the selected CPU subsystem. The clock mux controlled by this register is not glitch-free, therefore the CPUSELx register must be configured before the PCLKCRx register.
The reset for each peripheral is also driven from the selected CPU.

Figure 3-216. CPUSEL14 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED				RESERVED	DAC_C	DAC_B	DAC_A
R-0-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED							
R-0-0h							

Table 3-228. CPUSEL14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R-0	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	DAC_C	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
17	DAC_B	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
16	DAC_A	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
15-0	RESERVED	R-0	0h	Reserved

3.16.9.39 CPUSEL15 Register (Offset = F4h) [Reset = 0h]

CPUSEL15 is shown in Figure 3-217 and described in Table 3-229.

Return to the Summary Table.

CPU Select register for common peripherals
This register must be configured prior to enabling the peripheral clocks.
The clock for each peripheral is derived from the selected CPU subsystem. The clock mux controlled by this register is not glitch-free, therefore the CPUSELx register must be configured before the PCLKCRx register.
The reset for each peripheral is also driven from the selected CPU.

Figure 3-217. CPUSEL15 Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
CLB8	CLB7	CLB6	CLB5	CLB4	CLB3	CLB2	CLB1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-229. CPUSEL15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R/W	0h	Reserved
7	CLB8	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
6	CLB7	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
5	CLB6	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
4	CLB5	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
3	CLB4	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
2	CLB3	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
1	CLB2	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
0	CLB1	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn

3.16.9.40 CPUSEL16 Register (Offset = F6h) [Reset = 0h]

CPUSEL16 is shown in Figure 3-218 and described in Table 3-230.

Return to the Summary Table.

CPU Select register for common peripherals
This register must be configured prior to enabling the peripheral clocks.
The clock for each peripheral is derived from the selected CPU subsystem. The clock mux controlled by this register is not glitch-free, therefore the CPUSELx register must be configured before the PCLKCRx register.
The reset for each peripheral is also driven from the selected CPU.

Figure 3-218. CPUSEL16 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
FSIRX_H	FSIRX_G	FSIRX_F	FSIRX_E	FSIRX_D	FSIRX_C	FSIRX_B	FSIRX_A
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	FSITX_B	FSITX_A
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-230. CPUSEL16 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R-0	0h	Reserved
23	FSIRX_H	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
22	FSIRX_G	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
21	FSIRX_F	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
20	FSIRX_E	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
19	FSIRX_D	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
18	FSIRX_C	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
17	FSIRX_B	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
16	FSIRX_A	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
15-8	RESERVED	R-0	0h	Reserved
7	RESERVED	R/W	0h	Reserved

Table 3-230. CPUSEL16 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	FSITX_B	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn
0	FSITX_A	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn

3.16.9.41 CPUSEL18 Register (Offset = FAh) [Reset = 0h]

CPUSEL18 is shown in Figure 3-219 and described in Table 3-231.

Return to the Summary Table.

CPU Select register for common peripherals
This register must be configured prior to enabling the peripheral clocks.
The clock for each peripheral is derived from the selected CPU subsystem. The clock mux controlled by this register is not glitch-free, therefore the CPUSELx register must be configured before the PCLKCRx register.
The reset for each peripheral is also driven from the selected CPU.

Figure 3-219. CPUSEL18 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	PMBUS_A
R-0h						R/W-0h	R/W-0h

Table 3-231. CPUSEL18 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	PMBUS_A	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn

3.16.9.42 CPUSEL25 Register (Offset = 108h) [Reset = 0h]

CPUSEL25 is shown in Figure 3-220 and described in Table 3-232.

Return to the Summary Table.

CPU Select register for common peripherals
This register must be configured prior to enabling the peripheral clocks.
The clock for each peripheral is derived from the selected CPU subsystem. The clock mux controlled by this register is not glitch-free, therefore the CPUSELx register must be configured before the PCLKCRx register.
The reset for each peripheral is also driven from the selected CPU.

Figure 3-220. CPUSEL25 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							HRCAL_A
R-0h							R/W-0h

Table 3-232. CPUSEL25 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	HRCAL_A	R/W	0h	0: Connected to CPU1 1: Connected to CPU2 Reset type: CPU1.SYSRSn

3.16.9.43 CPU2RESCTL Register (Offset = 122h) [Reset = 1h]

CPU2RESCTL is shown in Figure 3-221 and described in Table 3-233.

Return to the Summary Table.

CPU2 Reset Control Register

Figure 3-221. CPU2RESCTL Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED							RESET
R-0-0h							R/W-1h

Table 3-233. CPU2RESCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to this register succeeds only if this field is written with a value of 0xa5a5 Note: [1] Due to this KEY, only 32-bit writes will succeed (provided the KEY matches). 16-bit writes to the upper or lower half of this register will be ignored Reset type: CPU1.SYSRSn
15-1	RESERVED	R-0	0h	Reserved
0	RESET	R/W	1h	This bit controls the reset input of CPU2 core. 1: CPU2 is held in reset (CPU2.RSn = 0) 0: CPU2 reset is deactivated (CPU2.RSn = 1) Note: [1] If CPU2 is not used at-all by an application, it's advisable to put CPU2 in STANDBY mode rather than in reset to save on active power component on the CPU2 subsystem. This is because, all clocks keep toggling when reset is active on the CPU2 sub-system. [2] Note: If CPU2 is in Standby mode, writing to this bit will have no effect. CPU2 may be reset by any Chip-level reset (POR, XRSn, CPU1.WDRSn, or CPU1.NMIWDRSn). Alternately CPU2 may be woken up by any configured wake-up event. Reset type: CPU1.SYSRSn

3.16.9.44 RSTSTAT Register (Offset = 124h) [Reset = 0h]

RSTSTAT is shown in Figure 3-222 and described in Table 3-234.

Return to the Summary Table.

Reset Status register for secondary C28x CPUs

Figure 3-222. RSTSTAT Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				CPU2HWBISTRST		CPU2NMIWDRST	CPU2RES
R-0-0h				R/W1S-0h		R/W1S-0h	R-0h

Table 3-234. RSTSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R-0	0h	Reserved
3-2	CPU2HWBISTRST	R/W1S	0h	CPU2HWBISTRST0 and CPU2HWBISTRST1 together tells whether a HWBIST reset was issued to CPU2 or not 00: CPU2 was not reset by the CPU2 HWBIST 11: CPU2 was reset due to CPU2 HWBIST reset This status bit is a latched flag. This flag can be cleared by the CPU1 by writing a 1 Reset type: CPU1.SYSRSn
1	CPU2NMIWDRST	R/W1S	0h	Indicates whether a CPU2.NMIWD reset was issued to CPU2 or not 0: CPU2 was not reset by the CPU2.NMIWD 1: CPU2 was reset due to CPU2.NMIWD reset Reset type: CPU1.SYSRSn
0	CPU2RES	R	0h	Reset status of CPU2 to CPU1 0: CPU2 core is in reset 1: CPU2 core is out of reset Reset type: CPU1.SYSRSn

3.16.9.45 LPMSTAT Register (Offset = 125h) [Reset = 0h]

LPMSTAT is shown in Figure 3-223 and described in Table 3-235.

Return to the Summary Table.

LPM Status Register for secondary C28x CPUs

Figure 3-223. LPMSTAT Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						CPU2LPMSTAT	
R-0-0h						R-0h	

Table 3-235. LPMSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R-0	0h	Reserved
1-0	CPU2LPMSTAT	R	0h	These bits indicate the power mode CPU2 00: CPU2 is in ACTIVE mode 01: CPU2 is in IDLE mode 10: CPU2 is in STANDBY mode 11: Reserved Reset type: CPU1.SYSRSn

3.16.9.46 USBTYPE Register (Offset = 19Ah) [Reset = 0h]

USBTYPE is shown in Figure 3-224 and described in Table 3-236.

Return to the Summary Table.

Based on the configuration enables disables features associated with the USB type.

Figure 3-224. USBTYPE Register

15	14	13	12	11	10	9	8
LOCK	RESERVED						
R/WSonce-0h	R-0-0h						
7	6	5	4	3	2	1	0
RESERVED						TYPE	
R-0-0h						R/W-0h	

Table 3-236. USBTYPE Register Field Descriptions

Bit	Field	Type	Reset	Description
15	LOCK	R/WSonce	0h	1: Write to this register is not allowed. 0: Write to this register is allowed. Reset type: CPU1.SYSRSn
14-2	RESERVED	R-0	0h	Reserved
1-0	TYPE	R/W	0h	"00,10,11" : 1. Global interrupt feature is not enabled, interrupts fired unconditionally. "01" : 1.Global interrupt feature is enabled, refer to the USB chapter for more details about global interrupt feature. Reset type: CPU1.SYSRSn

3.16.9.47 ECAPTYPE Register (Offset = 19Bh) [Reset = 0h]

ECAPTYPE is shown in Figure 3-225 and described in Table 3-237.

Return to the Summary Table.

Based on the configuration enables disables features associated with the SDFM type.

Figure 3-225. ECAPTYPE Register

15	14	13	12	11	10	9	8
LOCK	RESERVED						
R/WSonce-0h	R-0-0h						
7	6	5	4	3	2	1	0
RESERVED						TYPE	
R-0-0h						R/W-0h	

Table 3-237. ECAPTYPE Register Field Descriptions

Bit	Field	Type	Reset	Description
15	LOCK	R/WSonce	0h	1: Write to this register is not allowed. 0: Write to this register is allowed. Reset type: CPU1.SYSRSn
14-2	RESERVED	R-0	0h	Reserved
1-0	TYPE	R/W	0h	"00,10,11" : 1. No EALLOW protection to ECAP registers. "01" : 1. ECAP registers are EALLOW protected. Reset type: CPU1.SYSRSn

3.16.9.48 SDFMTYPE Register (Offset = 19Ch) [Reset = 0h]

SDFMTYPE is shown in Figure 3-226 and described in Table 3-238.

Return to the Summary Table.

Based on the configuration enables disables features associated with the SDFM type.

Figure 3-226. SDFMTYPE Register

15	14	13	12	11	10	9	8
LOCK	RESERVED						
R/WSonce-0h	R-0-0h						
7	6	5	4	3	2	1	0
RESERVED						TYPE	
R-0-0h						R/W-0h	

Table 3-238. SDFMTYPE Register Field Descriptions

Bit	Field	Type	Reset	Description
15	LOCK	R/WSonce	0h	1: Write to this register is not allowed. 0: Write to this register is allowed. Reset type: CPU1.SYSRSn
14-2	RESERVED	R-0	0h	Reserved
1-0	TYPE	R/W	0h	"00,10,11" : 1. Data Ready conditions combined with the fault conditions on the SDFM interrupt line. 2. Data ready interrupts from individual filters are not generated. "01" : 1. Data Ready conditions do not generate the SDFMINT. 2. Each filter generates a separate data ready interrupts. Reset type: CPU1.SYSRSn

3.16.9.49 MEMMAPTYPE Register (Offset = 19Eh) [Reset = 0h]

MEMMAPTYPE is shown in Figure 3-227 and described in Table 3-239.

Return to the Summary Table.

Based on the configuration enables the memory map.

Figure 3-227. MEMMAPTYPE Register

15	14	13	12	11	10	9	8
LOCK	RESERVED						
R/WSonce-0h	R-0-0h						
7	6	5	4	3	2	1	0
RESERVED						TYPE	
R-0-0h						R/W-0h	

Table 3-239. MEMMAPTYPE Register Field Descriptions

Bit	Field	Type	Reset	Description
15	LOCK	R/WSonce	0h	1: Write to this register is not allowed. 0: Write to this register is allowed. Reset type: CPU1.SYSRSn
14-2	RESERVED	R-0	0h	Reserved
1-0	TYPE	R/W	0h	"00,10,11" : 1. Disables re-mapping SDRAM in lower 128kb of address space. "01" : 1. Enables re-mapping SDRAM in lower 128kb of address space. Reset type: CPU1.SYSRSn

3.16.10 DMA_CLA_SRC_SEL_REGS Registers

Table 3-240 lists the memory-mapped registers for the DMA_CLA_SRC_SEL_REGS registers. All register offset addresses not listed in Table 3-240 should be considered as reserved locations and the register contents should not be modified.

Table 3-240. DMA_CLA_SRC_SEL_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CLA1TASKSRCSELLOCK	CLA1 Task Trigger Source Select Lock Register	EALLOW	Go
4h	DMACHSRCSELLOCK	DMA Channel Triger Source Select Lock Register	EALLOW	Go
6h	CLA1TASKSRCSEL1	CLA1 Task Trigger Source Select Register-1	EALLOW	Go
8h	CLA1TASKSRCSEL2	CLA1 Task Trigger Source Select Register-2	EALLOW	Go
16h	DMACHSRCSEL1	DMA Channel Trigger Source Select Register-1	EALLOW	Go
18h	DMACHSRCSEL2	DMA Channel Trigger Source Select Register-2	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-241 shows the codes that are used for access types in this section.

Table 3-241. DMA_CLA_SRC_SEL_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.10.1 CLA1TASKSRCSELLOCK Register (Offset = 0h) [Reset = 0h]

CLA1TASKSRCSELLOCK is shown in Figure 3-228 and described in Table 3-242.

Return to the Summary Table.

CLA1 Task Trigger Source Select Lock Register

Figure 3-228. CLA1TASKSRCSELLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						CLA1TASKSRCSEL2	CLA1TASKSRCSEL1
R-0-0h						R/WSonce-0h	R/WSonce-0h

Table 3-242. CLA1TASKSRCSELLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-2	RESERVED	R-0	0h	Reserved
1	CLA1TASKSRCSEL2	R/WSonce	0h	CLA1TASKSRCSEL2 Register Lock bit: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any SOnce bit in this register, once set can only be cleared through a SYSRSn. Write of 0 to any bit of this register has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: SYSRSn
0	CLA1TASKSRCSEL1	R/WSonce	0h	CLA1TASKSRCSEL1 Register Lock bit: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any SOnce bit in this register, once set can only be cleared through a SYSRSn. Write of 0 to any bit of this register has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: SYSRSn

3.16.10.2 DMACHSRCSELLOCK Register (Offset = 4h) [Reset = 0h]

DMACHSRCSELLOCK is shown in Figure 3-229 and described in Table 3-243.

Return to the Summary Table.

DMA Channel Triger Source Select Lock Register

Figure 3-229. DMACHSRCSELLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						DMACHSRCSEL2	DMACHSRCSEL1
R-0-0h						R/WSonce-0h	R/WSonce-0h

Table 3-243. DMACHSRCSELLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-2	RESERVED	R-0	0h	Reserved
1	DMACHSRCSEL2	R/WSonce	0h	DMACHSRCSEL2 Register Lock bit: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any SOnce bit in this register, once set can only be cleared through a SYSRSn. Write of 0 to any bit of this register has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: SYSRSn
0	DMACHSRCSEL1	R/WSonce	0h	DMACHSRCSEL1 Register Lock bit: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any SOnce bit in this register, once set can only be cleared through a SYSRSn. Write of 0 to any bit of this register has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: SYSRSn

3.16.10.3 CLA1TASKSRCSEL1 Register (Offset = 6h) [Reset = 0h]

CLA1TASKSRCSEL1 is shown in Figure 3-230 and described in Table 3-244.

Return to the Summary Table.

CLA1 Task Trigger Source Select Register-1

Figure 3-230. CLA1TASKSRCSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TASK4								TASK3								TASK2								TASK1							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 3-244. CLA1TASKSRCSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	TASK4	R/W	0h	Selects the Trigger Source for TASK4 of CLA1 Reset type: SYSRSn
23-16	TASK3	R/W	0h	Selects the Trigger Source for TASK3 of CLA1 Reset type: SYSRSn
15-8	TASK2	R/W	0h	Selects the Trigger Source for TASK2 of CLA1 Reset type: SYSRSn
7-0	TASK1	R/W	0h	Selects the Trigger Source for TASK1 of CLA1 Reset type: SYSRSn

3.16.10.4 CLA1TASKSRCSEL2 Register (Offset = 8h) [Reset = 0h]

CLA1TASKSRCSEL2 is shown in Figure 3-231 and described in Table 3-245.

Return to the Summary Table.

CLA1 Task Trigger Source Select Register-2

Figure 3-231. CLA1TASKSRCSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TASK8								TASK7								TASK6								TASK5							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 3-245. CLA1TASKSRCSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	TASK8	R/W	0h	Selects the Trigger Source for TASK8 of CLA1 Reset type: SYSRSn
23-16	TASK7	R/W	0h	Selects the Trigger Source for TASK7 of CLA1 Reset type: SYSRSn
15-8	TASK6	R/W	0h	Selects the Trigger Source for TASK6 of CLA1 Reset type: SYSRSn
7-0	TASK5	R/W	0h	Selects the Trigger Source for TASK5 of CLA1 Reset type: SYSRSn

3.16.10.5 DMACHSRCSEL1 Register (Offset = 16h) [Reset = 0h]

DMACHSRCSEL1 is shown in Figure 3-232 and described in Table 3-246.

Return to the Summary Table.

DMA Channel Trigger Source Select Register-1

Figure 3-232. DMACHSRCSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH4								CH3								CH2								CH1							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 3-246. DMACHSRCSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	CH4	R/W	0h	Selects the Trigger and Sync Source CH4 of DMA Reset type: SYSRSn
23-16	CH3	R/W	0h	Selects the Trigger and Sync Source CH3 of DMA Reset type: SYSRSn
15-8	CH2	R/W	0h	Selects the Trigger and Sync Source CH2 of DMA Reset type: SYSRSn
7-0	CH1	R/W	0h	Selects the Trigger and Sync Source CH1 of DMA Reset type: SYSRSn

3.16.10.6 DMACHSRCSEL2 Register (Offset = 18h) [Reset = 0h]

DMACHSRCSEL2 is shown in Figure 3-233 and described in Table 3-247.

Return to the Summary Table.

DMA Channel Trigger Source Select Register-2

Figure 3-233. DMACHSRCSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CH6								CH5							
R-0-0h																R/W-0h								R/W-0h							

Table 3-247. DMACHSRCSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	CH6	R/W	0h	Selects the Trigger and Sync Source CH6 of DMA Reset type: SYSRSn
7-0	CH5	R/W	0h	Selects the Trigger and Sync Source CH5 of DMA Reset type: SYSRSn

3.16.11 MEM_CFG_REGS Registers

Table 3-248 lists the memory-mapped registers for the MEM_CFG_REGS registers. All register offset addresses not listed in Table 3-248 should be considered as reserved locations and the register contents should not be modified.

Table 3-248. MEM_CFG_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	DxLOCK	Dedicated RAM Config Lock Register	EALLOW	Go
2h	DxCOMMIT	Dedicated RAM Config Lock Commit Register	EALLOW	Go
8h	DxACCPROT0	Dedicated RAM Config Register	EALLOW	Go
10h	DxTEST	Dedicated RAM TEST Register	EALLOW	Go
12h	DxINIT	Dedicated RAM Init Register	EALLOW	Go
14h	DxINITDONE	Dedicated RAM InitDone Status Register		Go
16h	DxRAMTEST_LOCK	Lock register to Dx RAM TEST registers		Go
20h	LSxLOCK	Local Shared RAM Config Lock Register	EALLOW	Go
22h	LSxCOMMIT	Local Shared RAM Config Lock Commit Register	EALLOW	Go
24h	LSxMSEL	Local Shared RAM Master Sel Register	EALLOW	Go
26h	LSxCLAPGM	Local Shared RAM Prog/Exe control Register	EALLOW	Go
28h	LSxACCPROT0	Local Shared RAM Config Register 0	EALLOW	Go
2Ah	LSxACCPROT1	Local Shared RAM Config Register 1	EALLOW	Go
30h	LSxTEST	Local Shared RAM TEST Register	EALLOW	Go
32h	LSxINIT	Local Shared RAM Init Register	EALLOW	Go
34h	LSxINITDONE	Local Shared RAM InitDone Status Register		Go
36h	LSxRAMTEST_LOCK	Lock register to LSx RAM TEST registers		Go
40h	GSxLOCK	Global Shared RAM Config Lock Register	EALLOW	Go
42h	GSxCOMMIT	Global Shared RAM Config Lock Commit Register	EALLOW	Go
44h	GSxMSEL	Global Shared RAM Master Sel Register	EALLOW	Go
48h	GSxACCPROT0	Global Shared RAM Access Protection Register 0	EALLOW	Go
4Ah	GSxACCPROT1	Global Shared RAM Access Protection Register 1	EALLOW	Go
4Ch	GSxACCPROT2	Global Shared RAM Access Protection Register 2	EALLOW	Go
4Eh	GSxACCPROT3	Global Shared RAM Access Protection Register 3	EALLOW	Go
50h	GSxTEST	Global Shared RAM TEST Register	EALLOW	Go
52h	GSxINIT	Global Shared RAM Init Register	EALLOW	Go
54h	GSxINITDONE	Global Shared RAM InitDone Status Register		Go
56h	GSxRAMTEST_LOCK	Lock register to GSx RAM TEST registers		Go
60h	MSGxLOCK	Message RAM Config Lock Register		Go
62h	MSGxCOMMIT	Message RAM Config Lock Commit Register		Go
68h	MSGxACCPROT0	Message RAM Access Protection Register 0	EALLOW	Go
6Ah	MSGxACCPROT1	Message RAM Access Protection Register 1	EALLOW	Go
6Ch	MSGxACCPROT2	Message RAM Access Protection Register 2	EALLOW	Go
70h	MSGxTEST	Message RAM TEST Register	EALLOW	Go
72h	MSGxINIT	Message RAM Init Register	EALLOW	Go
74h	MSGxINITDONE	Message RAM InitDone Status Register		Go
76h	MSGxRAMTEST_LOCK	Lock register to MSGx RAM TEST registers		Go
A0h	ROM_LOCK	ROM Config Lock Register		Go
A2h	ROM_TEST	ROM TEST Register		Go
A4h	ROM_FORCE_ERROR	ROM Force Error register		Go

Table 3-248. MEM_CFG_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
AAh	PERI_MEM_TEST_LOCK	Peripheral Memory Test Lock Register		Go
ACh	PERI_MEM_TEST_CONTROL	Peripheral Memory Test control Register		Go

Complex bit access types are encoded to fit into small table cells. Table 3-249 shows the codes that are used for access types in this section.

Table 3-249. MEM_CFG_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.11.1 DxLOCK Register (Offset = 0h) [Reset = 0h]

DxLOCK is shown in Figure 3-234 and described in Table 3-250.

Return to the Summary Table.

Dedicated RAM Config Lock Register

Figure 3-234. DxLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				LOCK_D1	LOCK_D0	LOCK_M1	LOCK_M0
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-250. DxLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	LOCK_D1	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for D1 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
2	LOCK_D0	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for D0 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
1	LOCK_M1	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for M1 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
0	LOCK_M0	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for M0 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn

3.16.11.2 DxCOMMIT Register (Offset = 2h) [Reset = 0h]

DxCOMMIT is shown in Figure 3-235 and described in Table 3-251.

Return to the Summary Table.

Dedicated RAM Config Lock Commit Register

Figure 3-235. DxCOMMIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				COMMIT_D1	COMMIT_D0	COMMIT_M1	COMMIT_M0
R-0h				R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 3-251. DxCOMMIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	COMMIT_D1	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for D1 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in DxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn
2	COMMIT_D0	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for D0 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in DxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn
1	COMMIT_M1	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for M1 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in DxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn
0	COMMIT_M0	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for M0 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in DxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn

3.16.11.3 DxACCPROT0 Register (Offset = 8h) [Reset = 0h]

DxACCPROT0 is shown in Figure 3-236 and described in Table 3-252.

Return to the Summary Table.

Dedicated RAM Config Register

Figure 3-236. DxACCPROT0 Register

31	30	29	28	27	26	25	24
RESERVED						CPUWRPROT_D1	FETCHPROT_D1
R-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED						CPUWRPROT_D0	FETCHPROT_D0
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED						CPUWRPROT_M1	FETCHPROT_M1
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED						CPUWRPROT_M0	FETCHPROT_M0
R-0h						R/W-0h	R/W-0h

Table 3-252. DxACCPROT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved
25	CPUWRPROT_D1	R/W	0h	CPU Write Protection For D1 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
24	FETCHPROT_D1	R/W	0h	Fetch Protection For D1 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
23-18	RESERVED	R	0h	Reserved
17	CPUWRPROT_D0	R/W	0h	CPU Write Protection For D0 RAM: 0: CPU Writes are allowed. 1: CPU Writes are block. Reset type: SYSRSn
16	FETCHPROT_D0	R/W	0h	Fetch Protection For D0 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
15-10	RESERVED	R	0h	Reserved
9	CPUWRPROT_M1	R/W	0h	CPU WR Protection For M1 RAM: 0: CPU Writes are allowed. 1: CPU Writes are block. Reset type: SYSRSn
8	FETCHPROT_M1	R/W	0h	Fetch Protection For M1 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
7-2	RESERVED	R	0h	Reserved

Table 3-252. DxACCPROT0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	CPUWRPROT_M0	R/W	0h	CPU WR Protection For M0 RAM: 0: CPU Writes are allowed. 1: CPU Writes are block. Reset type: SYSRSn
0	FETCHPROT_M0	R/W	0h	Fetch Protection For M0 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn

3.16.11.4 DxTEST Register (Offset = 10h) [Reset = 0h]

DxTEST is shown in Figure 3-237 and described in Table 3-253.

Return to the Summary Table.

Dedicated RAM TEST Register

Figure 3-237. DxTEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
TEST_D1		TEST_D0		TEST_M1		TEST_M0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 3-253. DxTEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7-6	TEST_D1	R/W	0h	Selects the defferent modes for D1 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to ECC bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Reset type: SYSRSn
5-4	TEST_D0	R/W	0h	Selects the defferent modes for D0 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to ECC bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Reset type: SYSRSn
3-2	TEST_M1	R/W	0h	Selects the defferent modes for M1 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to ECC bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Reset type: SYSRSn
1-0	TEST_M0	R/W	0h	Selects the defferent modes for M0 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to ECC bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Reset type: SYSRSn

3.16.11.5 DxINIT Register (Offset = 12h) [Reset = 0h]

DxINIT is shown in Figure 3-238 and described in Table 3-254.

Return to the Summary Table.

Dedicated RAM Init Register

Figure 3-238. DxINIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				INIT_D1	INIT_D0	INIT_M1	INIT_M0
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-254. DxINIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	INIT_D1	R-0/W1S	0h	RAM Initialization control for D1 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
2	INIT_D0	R-0/W1S	0h	RAM Initialization control for D0 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
1	INIT_M1	R-0/W1S	0h	RAM Initialization control for M1 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
0	INIT_M0	R-0/W1S	0h	RAM Initialization control for M0 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn

3.16.11.6 DxINITDONE Register (Offset = 14h) [Reset = 0h]

DxINITDONE is shown in Figure 3-239 and described in Table 3-255.

Return to the Summary Table.

Dedicated RAM InitDone Status Register

Figure 3-239. DxINITDONE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				INITDONE_D1	INITDONE_D0	INITDONE_M1	INITDONE_M0
R-0h				R-0h	R-0h	R-0h	R-0h

Table 3-255. DxINITDONE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	INITDONE_D1	R	0h	RAM Initialization status for D1 RAM: 0: RAM Initialization has completed. 1: RAM Initialization has completed. Reset type: SYSRSn
2	INITDONE_D0	R	0h	RAM Initialization status for D0 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
1	INITDONE_M1	R	0h	RAM Initialization status for M1 RAM: 0: RAM Initialization is not done. 1: RAM Initialization has completed. Reset type: SYSRSn
0	INITDONE_M0	R	0h	RAM Initialization status for M0 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn

3.16.11.7 DxRAMTEST_LOCK Register (Offset = 16h) [Reset = 0h]

DxRAMTEST_LOCK is shown in Figure 3-240 and described in Table 3-256.

Return to the Summary Table.

Lock register to Dx RAM TEST registers

Figure 3-240. DxRAMTEST_LOCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
R-0/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												D1	D0	M1	M0
R-0h												R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-256. DxRAMTEST_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	A value of 0xa5a5 to this field is simultaneously required for the writes to the rest of the fields of this register to succeed, Reset type: SYSRSn
15-4	RESERVED	R	0h	Reserved
3	D1	R/W	0h	0: Allows writes to DxTEST.TEST_D1 field. 1: Blocks writes to DxTEST.TEST_D1 field Reset type: SYSRSn
2	D0	R/W	0h	0: Allows writes to DxTEST.TEST_D0 field. 1: Blocks writes to DxTEST.TEST_D0 field Reset type: SYSRSn
1	M1	R/W	0h	0: Allows writes to DxTEST.TEST_M1 field. 1: Blocks writes to DxTEST.TEST_M1 field Reset type: SYSRSn
0	M0	R/W	0h	0: Allows writes to DxTEST.TEST_M0 field. 1: Blocks writes to DxTEST.TEST_M0 field Reset type: SYSRSn

3.16.11.8 LSxLOCK Register (Offset = 20h) [Reset = 0h]

LSxLOCK is shown in Figure 3-241 and described in Table 3-257.

Return to the Summary Table.

Local Shared RAM Config Lock Register

Figure 3-241. LSxLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
LOCK_LS7	LOCK_LS6	LOCK_LS5	LOCK_LS4	LOCK_LS3	LOCK_LS2	LOCK_LS1	LOCK_LS0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-257. LSxLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	LOCK_LS7	R/W	0h	Locks the write to access protection, master select, program or data memory select, initialization control and register fields test for LS7 RAM: 0: Write to ACCPROT, INIT, CLAPGM and Mselect fields are allowed. 1: Write to ACCPROT, INIT, CLAPGM and Mselect fields are blocked. Reset type: SYSRSn
6	LOCK_LS6	R/W	0h	Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS6 RAM: 0: Write to ACCPROT, INIT, CLAPGM and Mselect fields are allowed. 1: Write to ACCPROT, INIT, CLAPGM and Mselect fields are blocked. Reset type: SYSRSn
5	LOCK_LS5	R/W	0h	Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS5 RAM: 0: Write to ACCPROT, INIT, CLAPGM and Mselect fields are allowed. 1: Write to ACCPROT, INIT, CLAPGM and Mselect fields are blocked. Reset type: SYSRSn
4	LOCK_LS4	R/W	0h	Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS4 RAM: 0: Write to ACCPROT, INIT, CLAPGM and Mselect fields are allowed. 1: Write to ACCPROT, INIT, CLAPGM and Mselect fields are blocked. Reset type: SYSRSn

Table 3-257. LSxLOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	LOCK_LS3	R/W	0h	Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS3 RAM: 0: Write to ACCPROT, INIT, CLAPGM and Mselect fields are allowed. 1: Write to ACCPROT, INIT, CLAPGM and Mselect fields are blocked. Reset type: SYSRSn
2	LOCK_LS2	R/W	0h	Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS2 RAM: 0: Write to ACCPROT, INIT, CLAPGM and Mselect fields are allowed. 1: Write to ACCPROT, INIT, CLAPGM and Mselect fields are blocked. Reset type: SYSRSn
1	LOCK_LS1	R/W	0h	Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS1 RAM: 0: Write to ACCPROT, INIT, CLAPGM and Mselect fields are allowed. 1: Write to ACCPROT, INIT, CLAPGM and Mselect fields are blocked. Reset type: SYSRSn
0	LOCK_LS0	R/W	0h	Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS0 RAM: 0: Write to ACCPROT, INIT, CLAPGM and Mselect fields are allowed. 1: Write to ACCPROT, INIT, CLAPGM and Mselect fields are blocked. Reset type: SYSRSn

3.16.11.9 LSxCOMMIT Register (Offset = 22h) [Reset = 0h]

LSxCOMMIT is shown in Figure 3-242 and described in Table 3-258.

Return to the Summary Table.

Local Shared RAM Config Lock Commit Register

Figure 3-242. LSxCOMMIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
COMMIT_LS7	COMMIT_LS6	COMMIT_LS5	COMMIT_LS4	COMMIT_LS3	COMMIT_LS2	COMMIT_LS1	COMMIT_LS0
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 3-258. LSxCOMMIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	COMMIT_LS7	R/WSonce	0h	Permanently Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS7 RAM: 0: Write to ACCPROT, INIT, CLAPGM and Mselect fields are allowed based on value of lock field in LSxLOCK register. 1: Write to ACCPROT, INIT, CLAPGM and Mselect fields are permanently blocked. Reset type: SYSRSn
6	COMMIT_LS6	R/WSonce	0h	Permanently Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS6 RAM: 0: Write to ACCPROT, INIT, CLAPGM and Mselect fields are allowed based on value of lock field in LSxLOCK register. 1: Write to ACCPROT, INIT, CLAPGM and Mselect fields are permanently blocked. Reset type: SYSRSn
5	COMMIT_LS5	R/WSonce	0h	Permanently Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS5 RAM: 0: Write to ACCPROT, INIT, CLAPGM and Mselect fields are allowed based on value of lock field in LSxLOCK register. 1: Write to ACCPROT, INIT, CLAPGM and Mselect fields are permanently blocked. Reset type: SYSRSn
4	COMMIT_LS4	R/WSonce	0h	Permanently Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS4 RAM: 0: Write to ACCPROT, INIT, CLAPGM and Mselect fields are allowed based on value of lock field in LSxLOCK register. 1: Write to ACCPROT, INIT, CLAPGM and Mselect fields are permanently blocked. Reset type: SYSRSn

Table 3-258. LSxCOMMIT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	COMMIT_LS3	R/WSonce	0h	Permanently Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS3 RAM: 0: Write to ACCPROT, INIT, CLAPGM and Mselect fields are allowed based on value of lock field in LSxLOCK register. 1: Write to ACCPROT, INIT, CLAPGM and Mselect fields are permanently blocked. Reset type: SYSRSn
2	COMMIT_LS2	R/WSonce	0h	Permanently Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS2 RAM: 0: Write to ACCPROT, INIT, CLAPGM and Mselect fields are allowed based on value of lock field in LSxLOCK register. 1: Write to ACCPROT, INIT, CLAPGM and Mselect fields are permanently blocked. Reset type: SYSRSn
1	COMMIT_LS1	R/WSonce	0h	Permanently Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS1 RAM: 0: Write to ACCPROT, INIT, CLAPGM and Mselect fields are allowed based on value of lock field in LSxLOCK register. 1: Write to ACCPROT, INIT, CLAPGM and Mselect fields are permanently blocked. Reset type: SYSRSn
0	COMMIT_LS0	R/WSonce	0h	Permanently Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS0 RAM: 0: Write to ACCPROT, INIT, CLAPGM and Mselect fields are allowed based on value of lock field in LSxLOCK register. 1: Write to ACCPROT, INIT, CLAPGM and Mselect fields are permanently blocked. Reset type: SYSRSn

3.16.11.10 LSxMSEL Register (Offset = 24h) [Reset = 0h]

LSxMSEL is shown in Figure 3-243 and described in Table 3-259.

Return to the Summary Table.

Local Shared RAM Master Sel Register

Figure 3-243. LSxMSEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
MSEL_LS7		MSEL_LS6		MSEL_LS5		MSEL_LS4	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
MSEL_LS3		MSEL_LS2		MSEL_LS1		MSEL_LS0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 3-259. LSxMSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-14	MSEL_LS7	R/W	0h	Master Select for LS7 RAM: 00: Memory is dedicated to CPU. 01: Memory is shared between CPU and CLA1 if CLAPGM_LSx bit in LSxCLAPGM register is programmed as '0'. 10: Reserved. 11: Reserved. Reset type: SYSRSn
13-12	MSEL_LS6	R/W	0h	Master Select for LS6 RAM: 00: Memory is dedicated to CPU. 01: Memory is shared between CPU and CLA1 if CLAPGM_LSx bit in LSxCLAPGM register is programmed as '0'. 10: Reserved. 11: Reserved. Reset type: SYSRSn
11-10	MSEL_LS5	R/W	0h	Master Select for LS5 RAM: 00: Memory is dedicated to CPU. 01: Memory is shared between CPU and CLA1 if CLAPGM_LSx bit in LSxCLAPGM register is programmed as '0'. 10: Reserved. 11: Reserved. Reset type: SYSRSn
9-8	MSEL_LS4	R/W	0h	Master Select for LS4 RAM: 00: Memory is dedicated to CPU. 01: Memory is shared between CPU and CLA1 if CLAPGM_LSx bit in LSxCLAPGM register is programmed as '0'. 10: Reserved. 11: Reserved. Reset type: SYSRSn

Table 3-259. LSxMSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MSEL_LS3	R/W	0h	Master Select for LS3 RAM: 00: Memory is dedicated to CPU. 01: Memory is shared between CPU and CLA1 if CLAPGM_LSx bit in LSxCLAPGM register is programmed as '0'. 10: Reserved. 11: Reserved. Reset type: SYSRSn
5-4	MSEL_LS2	R/W	0h	Master Select for LS2 RAM: 00: Memory is dedicated to CPU. 01: Memory is shared between CPU and CLA1 if CLAPGM_LSx bit in LSxCLAPGM register is programmed as '0'. 10: Reserved. 11: Reserved. Reset type: SYSRSn
3-2	MSEL_LS1	R/W	0h	Master Select for LS1 RAM: 00: Memory is dedicated to CPU. 01: Memory is shared between CPU and CLA1 if CLAPGM_LSx bit in LSxCLAPGM register is programmed as '0'. 10: Reserved. 11: Reserved. Reset type: SYSRSn
1-0	MSEL_LS0	R/W	0h	Master Select for LS0 RAM: 00: Memory is dedicated to CPU. 01: Memory is shared between CPU and CLA1 if CLAPGM_LSx bit in LSxCLAPGM register is programmed as '0'. 10: Reserved. 11: Reserved. Reset type: SYSRSn

3.16.11.11 LSxCLAPGM Register (Offset = 26h) [Reset = 0h]

LSxCLAPGM is shown in Figure 3-244 and described in Table 3-260.

Return to the Summary Table.

Local Shared RAM Prog/Exe control Register

Figure 3-244. LSxCLAPGM Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
CLAPGM_LS7	CLAPGM_LS6	CLAPGM_LS5	CLAPGM_LS4	CLAPGM_LS3	CLAPGM_LS2	CLAPGM_LS1	CLAPGM_LS0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-260. LSxCLAPGM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	CLAPGM_LS7	R/W	0h	Selects LS7 RAM as program vs data memory for CLA: 0: CLA Data memory. 1: CLA Program memory. Reset type: SYSRSn
6	CLAPGM_LS6	R/W	0h	Selects LS6 RAM as program vs data memory for CLA: 0: CLA Data memory. 1: CLA Program memory. Reset type: SYSRSn
5	CLAPGM_LS5	R/W	0h	Selects LS5 RAM as program vs data memory for CLA: 0: CLA Data memory. 1: CLA Program memory. Reset type: SYSRSn
4	CLAPGM_LS4	R/W	0h	Selects LS4 RAM as program vs data memory for CLA: 0: CLA Data memory. 1: CLA Program memory. Reset type: SYSRSn
3	CLAPGM_LS3	R/W	0h	Selects LS3 RAM as program vs data memory for CLA: 0: CLA Data memory. 1: CLA Program memory. Reset type: SYSRSn
2	CLAPGM_LS2	R/W	0h	Selects LS2 RAM as program vs data memory for CLA: 0: CLA Data memory. 1: CLA Program memory. Reset type: SYSRSn
1	CLAPGM_LS1	R/W	0h	Selects LS1 RAM as program vs data memory for CLA: 0: CLA Data memory. 1: CLA Program memory. Reset type: SYSRSn

Table 3-260. LSxCLAPGM Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	CLAPGM_LS0	R/W	0h	Selects LS0 RAM as program vs data memory for CLA: 0: CLA Data memory. 1: CLA Program memory. Reset type: SYSRSn

3.16.11.12 LSxACCPROT0 Register (Offset = 28h) [Reset = 0h]

LSxACCPROT0 is shown in Figure 3-245 and described in Table 3-261.

Return to the Summary Table.

Local Shared RAM Config Register 0

Figure 3-245. LSxACCPROT0 Register

31	30	29	28	27	26	25	24
RESERVED						CPUWRPROT_LS3	FETCHPROT_LS3
R-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED						CPUWRPROT_LS2	FETCHPROT_LS2
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED						CPUWRPROT_LS1	FETCHPROT_LS1
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED						CPUWRPROT_LS0	FETCHPROT_LS0
R-0h						R/W-0h	R/W-0h

Table 3-261. LSxACCPROT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved
25	CPUWRPROT_LS3	R/W	0h	CPU WR Protection For LS3 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
24	FETCHPROT_LS3	R/W	0h	Fetch Protection For LS3 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
23-18	RESERVED	R	0h	Reserved
17	CPUWRPROT_LS2	R/W	0h	CPU WR Protection For LS2 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
16	FETCHPROT_LS2	R/W	0h	Fetch Protection For LS2 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
15-10	RESERVED	R	0h	Reserved
9	CPUWRPROT_LS1	R/W	0h	CPU WR Protection For LS1 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
8	FETCHPROT_LS1	R/W	0h	Fetch Protection For LS1 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
7-2	RESERVED	R	0h	Reserved

Table 3-261. LSxACCPROT0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	CPUWRPROT_LS0	R/W	0h	CPU WR Protection For LS0 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
0	FETCHPROT_LS0	R/W	0h	Fetch Protection For LS0 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn

3.16.11.13 LSxACCPROT1 Register (Offset = 2Ah) [Reset = 0h]

LSxACCPROT1 is shown in Figure 3-246 and described in Table 3-262.

Return to the Summary Table.

Local Shared RAM Config Register 1

Figure 3-246. LSxACCPROT1 Register

31	30	29	28	27	26	25	24
RESERVED						CPUWRPROT_LS7	FETCHPROT_LS7
R-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED						CPUWRPROT_LS6	FETCHPROT_LS6
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED						CPUWRPROT_LS5	FETCHPROT_LS5
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED						CPUWRPROT_LS4	FETCHPROT_LS4
R-0h						R/W-0h	R/W-0h

Table 3-262. LSxACCPROT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved
25	CPUWRPROT_LS7	R/W	0h	CPU WR Protection For LS7 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
24	FETCHPROT_LS7	R/W	0h	Fetch Protection For LS7 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
23-18	RESERVED	R	0h	Reserved
17	CPUWRPROT_LS6	R/W	0h	CPU WR Protection For LS6 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
16	FETCHPROT_LS6	R/W	0h	Fetch Protection For LS6 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
15-10	RESERVED	R	0h	Reserved
9	CPUWRPROT_LS5	R/W	0h	CPU WR Protection For LS5 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
8	FETCHPROT_LS5	R/W	0h	Fetch Protection For LS5 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
7-2	RESERVED	R	0h	Reserved

Table 3-262. LSxACCPROT1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	CPUWRPROT_LS4	R/W	0h	CPU WR Protection For LS4 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
0	FETCHPROT_LS4	R/W	0h	Fetch Protection For LS4 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn

3.16.11.14 LSxTEST Register (Offset = 30h) [Reset = 0h]

LSxTEST is shown in Figure 3-247 and described in Table 3-263.

Return to the Summary Table.

Local Shared RAM TEST Register

Figure 3-247. LSxTEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
TEST_LS7		TEST_LS6		TEST_LS5		TEST_LS4	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
TEST_LS3		TEST_LS2		TEST_LS1		TEST_LS0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 3-263. LSxTEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-14	TEST_LS7	R/W	0h	Selects the defferent modes for LS7 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ecc bits. 10: Writes are allowed to ecc bits only. No write to data bits. 11: Functional Mode. Reset type: SYSRSn
13-12	TEST_LS6	R/W	0h	Selects the defferent modes for LS6 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ecc bits. 10: Writes are allowed to ecc bits only. No write to data bits. 11: Functional Mode. Reset type: SYSRSn
11-10	TEST_LS5	R/W	0h	Selects the defferent modes for LS5 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ecc bits. 10: Writes are allowed to ecc bits only. No write to data bits. 11: Functional Mode. Reset type: SYSRSn
9-8	TEST_LS4	R/W	0h	Selects the defferent modes for LS4 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ecc bits. 10: Writes are allowed to ecc bits only. No write to data bits. 11: Functional Mode. Reset type: SYSRSn
7-6	TEST_LS3	R/W	0h	Selects the defferent modes for LS3 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ecc bits. 10: Writes are allowed to ecc bits only. No write to data bits. 11: Functional Mode. Reset type: SYSRSn

Table 3-263. LSxTEST Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-4	TEST_LS2	R/W	0h	Selects the defferent modes for LS2 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ecc bits. 10: Writes are allowed to ecc bits only. No write to data bits. 11: Functional Mode. Reset type: SYSRSn
3-2	TEST_LS1	R/W	0h	Selects the defferent modes for LS1 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ecc bits. 10: Writes are allowed to ecc bits only. No write to data bits. 11: Functional Mode. Reset type: SYSRSn
1-0	TEST_LS0	R/W	0h	Selects the defferent modes for LS0 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ecc bits. 10: Writes are allowed to ecc bits only. No write to data bits. 11: Functional Mode. Reset type: SYSRSn

3.16.11.15 LSxINIT Register (Offset = 32h) [Reset = 0h]

LSxINIT is shown in Figure 3-248 and described in Table 3-264.

Return to the Summary Table.

Local Shared RAM Init Register

Figure 3-248. LSxINIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INIT_LS7	INIT_LS6	INIT_LS5	INIT_LS4	INIT_LS3	INIT_LS2	INIT_LS1	INIT_LS0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-264. LSxINIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	INIT_LS7	R-0/W1S	0h	RAM Initialization control for LS7 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
6	INIT_LS6	R-0/W1S	0h	RAM Initialization control for LS6 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
5	INIT_LS5	R-0/W1S	0h	RAM Initialization control for LS5 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
4	INIT_LS4	R-0/W1S	0h	RAM Initialization control for LS4 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
3	INIT_LS3	R-0/W1S	0h	RAM Initialization control for LS3 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
2	INIT_LS2	R-0/W1S	0h	RAM Initialization control for LS2 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
1	INIT_LS1	R-0/W1S	0h	RAM Initialization control for LS1 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn

Table 3-264. LSxINIT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INIT_LS0	R-0/W1S	0h	RAM Initialization control for LS0 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn

3.16.11.16 LSxINITDONE Register (Offset = 34h) [Reset = 0h]

LSxINITDONE is shown in Figure 3-249 and described in Table 3-265.

Return to the Summary Table.

Local Shared RAM InitDone Status Register

Figure 3-249. LSxINITDONE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INITDONE_LS7	INITDONE_LS6	INITDONE_LS5	INITDONE_LS4	INITDONE_LS3	INITDONE_LS2	INITDONE_LS1	INITDONE_LS0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 3-265. LSxINITDONE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	INITDONE_LS7	R	0h	RAM Initialization status for LS7 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
6	INITDONE_LS6	R	0h	RAM Initialization status for LS6 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
5	INITDONE_LS5	R	0h	RAM Initialization status for LS5 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
4	INITDONE_LS4	R	0h	RAM Initialization status for LS4 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
3	INITDONE_LS3	R	0h	RAM Initialization status for LS3 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
2	INITDONE_LS2	R	0h	RAM Initialization status for LS2 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
1	INITDONE_LS1	R	0h	RAM Initialization status for LS1 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn

Table 3-265. LSxINITDONE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INITDONE_LS0	R	0h	RAM Initialization status for LS0 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn

3.16.11.17 LSxRAMTEST_LOCK Register (Offset = 36h) [Reset = 0h]

LSxRAMTEST_LOCK is shown in Figure 3-250 and described in Table 3-266.

Return to the Summary Table.

Lock register to LSx RAM TEST registers

Figure 3-250. LSxRAMTEST_LOCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
R-0/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								LS7	LS6	LS5	LS4	LS3	LS2	LS1	LS0
R-0h								R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-266. LSxRAMTEST_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	A value of 0xa5a5 to this field is simultaneously required for the writes to the rest of the fields of this register to succeed, Reset type: SYSRSn
15-8	RESERVED	R	0h	Reserved
7	LS7	R/W	0h	0: Allows writes to LSxTEST.TEST_LS7 field. 1: Blocks writes to LSxTEST.TEST_LS7 field. Reset type: SYSRSn
6	LS6	R/W	0h	0: Allows writes to LSxTEST.TEST_LS6 field. 1: Blocks writes to LSxTEST.TEST_LS6 field. Reset type: SYSRSn
5	LS5	R/W	0h	0: Allows writes to LSxTEST.TEST_LS5 field. 1: Blocks writes to LSxTEST.TEST_LS5 field. Reset type: SYSRSn
4	LS4	R/W	0h	0: Allows writes to LSxTEST.TEST_LS4 field. 1: Blocks writes to LSxTEST.TEST_LS4 field. Reset type: SYSRSn
3	LS3	R/W	0h	0: Allows writes to LSxTEST.TEST_LS3 field. 1: Blocks writes to LSxTEST.TEST_LS3 field. Reset type: SYSRSn
2	LS2	R/W	0h	0: Allows writes to LSxTEST.TEST_LS2 field. 1: Blocks writes to LSxTEST.TEST_LS2 field. Reset type: SYSRSn
1	LS1	R/W	0h	0: Allows writes to LSxTEST.TEST_LS1 field. 1: Blocks writes to LSxTEST.TEST_LS1 field. Reset type: SYSRSn
0	LS0	R/W	0h	0: Allows writes to LSxTEST.TEST_LS0 field. 1: Blocks writes to LSxTEST.TEST_LS0 field. Reset type: SYSRSn

3.16.11.18 GSxLOCK Register (Offset = 40h) [Reset = 0h]

GSxLOCK is shown in Figure 3-251 and described in Table 3-267.

Return to the Summary Table.

Global Shared RAM Config Lock Register

Figure 3-251. GSxLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
LOCK_GS15	LOCK_GS14	LOCK_GS13	LOCK_GS12	LOCK_GS11	LOCK_GS10	LOCK_GS9	LOCK_GS8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
LOCK_GS7	LOCK_GS6	LOCK_GS5	LOCK_GS4	LOCK_GS3	LOCK_GS2	LOCK_GS1	LOCK_GS0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-267. GSxLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	LOCK_GS15	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS15 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
14	LOCK_GS14	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS14 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
13	LOCK_GS13	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS13 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
12	LOCK_GS12	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS12 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
11	LOCK_GS11	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS11 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
10	LOCK_GS10	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS10 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn

Table 3-267. GSxLOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	LOCK_GS9	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS9 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
8	LOCK_GS8	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS8 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
7	LOCK_GS7	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS7 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
6	LOCK_GS6	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS6 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
5	LOCK_GS5	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS5 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
4	LOCK_GS4	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS4 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
3	LOCK_GS3	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS3 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
2	LOCK_GS2	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS2 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
1	LOCK_GS1	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS1 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
0	LOCK_GS0	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS0 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn

3.16.11.19 GSxCOMMIT Register (Offset = 42h) [Reset = 0h]

GSxCOMMIT is shown in Figure 3-252 and described in Table 3-268.

Return to the Summary Table.

Global Shared RAM Config Lock Commit Register

Figure 3-252. GSxCOMMIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
COMMIT_GS15	COMMIT_GS14	COMMIT_GS13	COMMIT_GS12	COMMIT_GS11	COMMIT_GS10	COMMIT_GS9	COMMIT_GS8
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
7	6	5	4	3	2	1	0
COMMIT_GS7	COMMIT_GS6	COMMIT_GS5	COMMIT_GS4	COMMIT_GS3	COMMIT_GS2	COMMIT_GS1	COMMIT_GS0
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 3-268. GSxCOMMIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	COMMIT_GS15	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS15 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn
14	COMMIT_GS14	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS14 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn
13	COMMIT_GS13	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS13 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn
12	COMMIT_GS12	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS12 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn

Table 3-268. GSxCOMMIT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	COMMIT_GS11	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS11 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn
10	COMMIT_GS10	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS10 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn
9	COMMIT_GS9	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS9 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn
8	COMMIT_GS8	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS8 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn
7	COMMIT_GS7	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS7 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn
6	COMMIT_GS6	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS6 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn
5	COMMIT_GS5	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS5 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn
4	COMMIT_GS4	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS4 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn

Table 3-268. GSxCOMMIT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	COMMIT_GS3	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS3 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn
2	COMMIT_GS2	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS2 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn
1	COMMIT_GS1	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS1 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn
0	COMMIT_GS0	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS0 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, INIT and Mselect fields are permanently blocked. Reset type: SYSRSn

3.16.11.20 GSxMSEL Register (Offset = 44h) [Reset = 0h]

GSxMSEL is shown in Figure 3-253 and described in Table 3-269.

Return to the Summary Table.

Global Shared RAM Master Sel Register

Figure 3-253. GSxMSEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
MSEL_GS15	MSEL_GS14	MSEL_GS13	MSEL_GS12	MSEL_GS11	MSEL_GS10	MSEL_GS9	MSEL_GS8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MSEL_GS7	MSEL_GS6	MSEL_GS5	MSEL_GS4	MSEL_GS3	MSEL_GS2	MSEL_GS1	MSEL_GS0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-269. GSxMSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	MSEL_GS15	R/W	0h	Master Select for GS15 RAM: 0: CPU1 is master for this memory. 1: CPU2 is master for this memory. Reset type: CPU1.SYSRSn
14	MSEL_GS14	R/W	0h	Master Select for GS14 RAM: 0: CPU1 is master for this memory. 1: CPU2 is master for this memory. Reset type: CPU1.SYSRSn
13	MSEL_GS13	R/W	0h	Master Select for GS13 RAM: 0: CPU1 is master for this memory. 1: CPU2 is master for this memory. Reset type: CPU1.SYSRSn
12	MSEL_GS12	R/W	0h	Master Select for GS12 RAM: 0: CPU1 is master for this memory. 1: CPU2 is master for this memory. Reset type: CPU1.SYSRSn
11	MSEL_GS11	R/W	0h	Master Select for GS11 RAM: 0: CPU1 is master for this memory. 1: CPU2 is master for this memory. Reset type: CPU1.SYSRSn
10	MSEL_GS10	R/W	0h	Master Select for GS10 RAM: 0: CPU1 is master for this memory. 1: CPU2 is master for this memory. Reset type: CPU1.SYSRSn
9	MSEL_GS9	R/W	0h	Master Select for GS9 RAM: 0: CPU1 is master for this memory. 1: CPU2 is master for this memory. Reset type: CPU1.SYSRSn
8	MSEL_GS8	R/W	0h	Master Select for GS8 RAM: 0: CPU1 is master for this memory. 1: CPU2 is master for this memory. Reset type: CPU1.SYSRSn

Table 3-269. GSxMSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	MSEL_GS7	R/W	0h	Master Select for GS7 RAM: 0: CPU1 is master for this memory. 1: CPU2 is master for this memory. Reset type: CPU1.SYSRSn
6	MSEL_GS6	R/W	0h	Master Select for GS6 RAM: 0: CPU1 is master for this memory. 1: CPU2 is master for this memory. Reset type: CPU1.SYSRSn
5	MSEL_GS5	R/W	0h	Master Select for GS5 RAM: 0: CPU1 is master for this memory. 1: CPU2 is master for this memory. Reset type: CPU1.SYSRSn
4	MSEL_GS4	R/W	0h	Master Select for GS4 RAM: 0: CPU1 is master for this memory. 1: CPU2 is master for this memory. Reset type: CPU1.SYSRSn
3	MSEL_GS3	R/W	0h	Master Select for GS3 RAM: 0: CPU1 is master for this memory. 1: CPU2 is master for this memory. Reset type: CPU1.SYSRSn
2	MSEL_GS2	R/W	0h	Master Select for GS2 RAM: 0: CPU1 is master for this memory. 1: CPU2 is master for this memory. Reset type: CPU1.SYSRSn
1	MSEL_GS1	R/W	0h	Master Select for GS1 RAM: 0: CPU1 is master for this memory. 1: CPU2 is master for this memory. Reset type: CPU1.SYSRSn
0	MSEL_GS0	R/W	0h	Master Select for GS0 RAM: 0: CPU1 is master for this memory. 1: CPU2 is master for this memory. Reset type: CPU1.SYSRSn

3.16.11.21 GSxACCPROT0 Register (Offset = 48h) [Reset = 0h]

GSxACCPROT0 is shown in Figure 3-254 and described in Table 3-270.

Return to the Summary Table.

Global Shared RAM Access Protection Register 0

Figure 3-254. GSxACCPROT0 Register

31	30	29	28	27	26	25	24
RESERVED					DMAWRPROT_GS3	CPUWRPROT_GS3	FETCHPROT_GS3
R-0h					R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED					DMAWRPROT_GS2	CPUWRPROT_GS2	FETCHPROT_GS2
R-0h					R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED					DMAWRPROT_GS1	CPUWRPROT_GS1	FETCHPROT_GS1
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED					DMAWRPROT_GS0	CPUWRPROT_GS0	FETCHPROT_GS0
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 3-270. GSxACCPROT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R	0h	Reserved
26	DMAWRPROT_GS3	R/W	0h	DMA WR Protection For GS3 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
25	CPUWRPROT_GS3	R/W	0h	CPU WR Protection For GS3 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
24	FETCHPROT_GS3	R/W	0h	Fetch Protection For GS3 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
23-19	RESERVED	R	0h	Reserved
18	DMAWRPROT_GS2	R/W	0h	DMA WR Protection For GS2 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
17	CPUWRPROT_GS2	R/W	0h	CPU WR Protection For GS2 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
16	FETCHPROT_GS2	R/W	0h	Fetch Protection For GS2 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
15-11	RESERVED	R	0h	Reserved

Table 3-270. GSxACCPROT0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	DMAWRPROT_GS1	R/W	0h	DMA WR Protection For GS1 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
9	CPUWRPROT_GS1	R/W	0h	CPU WR Protection For GS1 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
8	FETCHPROT_GS1	R/W	0h	Fetch Protection For GS1 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
7-3	RESERVED	R	0h	Reserved
2	DMAWRPROT_GS0	R/W	0h	DMA WR Protection For GS0 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
1	CPUWRPROT_GS0	R/W	0h	CPU WR Protection For GS0 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
0	FETCHPROT_GS0	R/W	0h	Fetch Protection For GS0 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn

3.16.11.22 GSxACCPROT1 Register (Offset = 4Ah) [Reset = 0h]

GSxACCPROT1 is shown in Figure 3-255 and described in Table 3-271.

Return to the Summary Table.

Global Shared RAM Access Protection Register 1

Figure 3-255. GSxACCPROT1 Register

31	30	29	28	27	26	25	24
RESERVED					DMAWRPROT_GS7	CPUWRPROT_GS7	FETCHPROT_GS7
R-0h					R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED					DMAWRPROT_GS6	CPUWRPROT_GS6	FETCHPROT_GS6
R-0h					R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED					DMAWRPROT_GS5	CPUWRPROT_GS5	FETCHPROT_GS5
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED					DMAWRPROT_GS4	CPUWRPROT_GS4	FETCHPROT_GS4
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 3-271. GSxACCPROT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R	0h	Reserved
26	DMAWRPROT_GS7	R/W	0h	DMA WR Protection For GS7 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
25	CPUWRPROT_GS7	R/W	0h	CPU WR Protection For GS7 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
24	FETCHPROT_GS7	R/W	0h	Fetch Protection For GS7 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
23-19	RESERVED	R	0h	Reserved
18	DMAWRPROT_GS6	R/W	0h	DMA WR Protection For GS6 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
17	CPUWRPROT_GS6	R/W	0h	CPU WR Protection For GS6 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
16	FETCHPROT_GS6	R/W	0h	Fetch Protection For GS6 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
15-11	RESERVED	R	0h	Reserved

Table 3-271. GSxACCPROT1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	DMAWRPROT_GS5	R/W	0h	DMA WR Protection For GS5 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
9	CPUWRPROT_GS5	R/W	0h	CPU WR Protection For GS5 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
8	FETCHPROT_GS5	R/W	0h	Fetch Protection For GS5 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
7-3	RESERVED	R	0h	Reserved
2	DMAWRPROT_GS4	R/W	0h	DMA WR Protection For GS4 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
1	CPUWRPROT_GS4	R/W	0h	CPU WR Protection For GS4 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
0	FETCHPROT_GS4	R/W	0h	Fetch Protection For GS4 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn

3.16.11.23 GSxACCPROT2 Register (Offset = 4Ch) [Reset = 0h]

GSxACCPROT2 is shown in Figure 3-256 and described in Table 3-272.

Return to the Summary Table.

Global Shared RAM Access Protection Register 2

Figure 3-256. GSxACCPROT2 Register

31	30	29	28	27	26	25	24
RESERVED					DMAWRPROT_GS11	CPUWRPROT_GS11	FETCHPROT_GS11
R-0h					R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED					DMAWRPROT_GS10	CPUWRPROT_GS10	FETCHPROT_GS10
R-0h					R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED					DMAWRPROT_GS9	CPUWRPROT_GS9	FETCHPROT_GS9
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED					DMAWRPROT_GS8	CPUWRPROT_GS8	FETCHPROT_GS8
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 3-272. GSxACCPROT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R	0h	Reserved
26	DMAWRPROT_GS11	R/W	0h	DMA WR Protection For GS11 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
25	CPUWRPROT_GS11	R/W	0h	CPU WR Protection For GS11 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
24	FETCHPROT_GS11	R/W	0h	Fetch Protection For GS11 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
23-19	RESERVED	R	0h	Reserved
18	DMAWRPROT_GS10	R/W	0h	DMA WR Protection For GS10 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
17	CPUWRPROT_GS10	R/W	0h	CPU WR Protection For GS10 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
16	FETCHPROT_GS10	R/W	0h	Fetch Protection For GS10 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
15-11	RESERVED	R	0h	Reserved

Table 3-272. GSxACCPROT2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	DMAWRPROT_GS9	R/W	0h	DMA WR Protection For GS9 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
9	CPUWRPROT_GS9	R/W	0h	CPU WR Protection For GS9 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
8	FETCHPROT_GS9	R/W	0h	Fetch Protection For GS9 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
7-3	RESERVED	R	0h	Reserved
2	DMAWRPROT_GS8	R/W	0h	DMA WR Protection For GS8 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
1	CPUWRPROT_GS8	R/W	0h	CPU WR Protection For GS8 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
0	FETCHPROT_GS8	R/W	0h	Fetch Protection For GS8 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn

3.16.11.24 GSxACCPROT3 Register (Offset = 4Eh) [Reset = 0h]

GSxACCPROT3 is shown in Figure 3-257 and described in Table 3-273.

Return to the Summary Table.

Global Shared RAM Access Protection Register 3

Figure 3-257. GSxACCPROT3 Register

31	30	29	28	27	26	25	24
RESERVED					DMAWRPROT_GS15	CPUWRPROT_GS15	FETCHPROT_GS15
R-0h					R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED					DMAWRPROT_GS14	CPUWRPROT_GS14	FETCHPROT_GS14
R-0h					R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED					DMAWRPROT_GS13	CPUWRPROT_GS13	FETCHPROT_GS13
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED					DMAWRPROT_GS12	CPUWRPROT_GS12	FETCHPROT_GS12
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 3-273. GSxACCPROT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R	0h	Reserved
26	DMAWRPROT_GS15	R/W	0h	DMA WR Protection For GS15 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
25	CPUWRPROT_GS15	R/W	0h	CPU WR Protection For GS15 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
24	FETCHPROT_GS15	R/W	0h	Fetch Protection For GS15 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
23-19	RESERVED	R	0h	Reserved
18	DMAWRPROT_GS14	R/W	0h	DMA WR Protection For GS14 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
17	CPUWRPROT_GS14	R/W	0h	CPU WR Protection For GS14 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
16	FETCHPROT_GS14	R/W	0h	Fetch Protection For GS14 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
15-11	RESERVED	R	0h	Reserved

Table 3-273. GSxACCPROT3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	DMAWRPROT_GS13	R/W	0h	DMA WR Protection For GS13 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
9	CPUWRPROT_GS13	R/W	0h	CPU WR Protection For GS13 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
8	FETCHPROT_GS13	R/W	0h	Fetch Protection For GS13 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
7-3	RESERVED	R	0h	Reserved
2	DMAWRPROT_GS12	R/W	0h	DMA WR Protection For GS12 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
1	CPUWRPROT_GS12	R/W	0h	CPU WR Protection For GS12 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
0	FETCHPROT_GS12	R/W	0h	Fetch Protection For GS12 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn

3.16.11.25 GSxTEST Register (Offset = 50h) [Reset = 0h]

GSxTEST is shown in Figure 3-258 and described in Table 3-274.

Return to the Summary Table.

Global Shared RAM TEST Register

Figure 3-258. GSxTEST Register

31	30	29	28	27	26	25	24
TEST_GS15		TEST_GS14		TEST_GS13		TEST_GS12	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
TEST_GS11		TEST_GS10		TEST_GS9		TEST_GS8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
TEST_GS7		TEST_GS6		TEST_GS5		TEST_GS4	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
TEST_GS3		TEST_GS2		TEST_GS1		TEST_GS0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 3-274. GSxTEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	TEST_GS15	R/W	0h	Selects the defferent modes for GS15 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
29-28	TEST_GS14	R/W	0h	Selects the defferent modes for GS14 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
27-26	TEST_GS13	R/W	0h	Selects the defferent modes for GS13 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn

Table 3-274. GSxTEST Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-24	TEST_GS12	R/W	0h	Selects the defferent modes for GS12 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
23-22	TEST_GS11	R/W	0h	Selects the defferent modes for GS11 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
21-20	TEST_GS10	R/W	0h	Selects the defferent modes for GS10 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
19-18	TEST_GS9	R/W	0h	Selects the defferent modes for GS9 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
17-16	TEST_GS8	R/W	0h	Selects the defferent modes for GS8 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
15-14	TEST_GS7	R/W	0h	Selects the defferent modes for GS7 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn

Table 3-274. GSxTEST Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13-12	TEST_GS6	R/W	0h	Selects the defferent modes for GS6 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
11-10	TEST_GS5	R/W	0h	Selects the defferent modes for GS5 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
9-8	TEST_GS4	R/W	0h	Selects the defferent modes for GS4 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
7-6	TEST_GS3	R/W	0h	Selects the defferent modes for GS3 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
5-4	TEST_GS2	R/W	0h	Selects the defferent modes for GS2 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
3-2	TEST_GS1	R/W	0h	Selects the defferent modes for GS1 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn

Table 3-274. GSxTEST Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	TEST_GS0	R/W	0h	Selects the defferent modes for GS0 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn

3.16.11.26 GSxINIT Register (Offset = 52h) [Reset = 0h]

GSxINIT is shown in Figure 3-259 and described in Table 3-275.

Return to the Summary Table.

Global Shared RAM Init Register

Figure 3-259. GSxINIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
INIT_GS15	INIT_GS14	INIT_GS13	INIT_GS12	INIT_GS11	INIT_GS10	INIT_GS9	INIT_GS8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
INIT_GS7	INIT_GS6	INIT_GS5	INIT_GS4	INIT_GS3	INIT_GS2	INIT_GS1	INIT_GS0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-275. GSxINIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	INIT_GS15	R-0/W1S	0h	RAM Initialization control for GS15 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
14	INIT_GS14	R-0/W1S	0h	RAM Initialization control for GS14 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
13	INIT_GS13	R-0/W1S	0h	RAM Initialization control for GS13 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
12	INIT_GS12	R-0/W1S	0h	RAM Initialization control for GS12 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
11	INIT_GS11	R-0/W1S	0h	RAM Initialization control for GS11 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
10	INIT_GS10	R-0/W1S	0h	RAM Initialization control for GS10 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
9	INIT_GS9	R-0/W1S	0h	RAM Initialization control for GS9 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
8	INIT_GS8	R-0/W1S	0h	RAM Initialization control for GS8 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn

Table 3-275. GSxINIT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	INIT_GS7	R-0/W1S	0h	RAM Initialization control for GS7 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
6	INIT_GS6	R-0/W1S	0h	RAM Initialization control for GS6 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
5	INIT_GS5	R-0/W1S	0h	RAM Initialization control for GS5 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
4	INIT_GS4	R-0/W1S	0h	RAM Initialization control for GS4 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
3	INIT_GS3	R-0/W1S	0h	RAM Initialization control for GS3 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
2	INIT_GS2	R-0/W1S	0h	RAM Initialization control for GS2 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
1	INIT_GS1	R-0/W1S	0h	RAM Initialization control for GS1 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
0	INIT_GS0	R-0/W1S	0h	RAM Initialization control for GS0 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn

3.16.11.27 GSxINITDONE Register (Offset = 54h) [Reset = 0h]

GSxINITDONE is shown in Figure 3-260 and described in Table 3-276.

Return to the Summary Table.

Global Shared RAM InitDone Status Register

Figure 3-260. GSxINITDONE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
INITDONE_GS15	INITDONE_GS14	INITDONE_GS13	INITDONE_GS12	INITDONE_GS11	INITDONE_GS10	INITDONE_GS9	INITDONE_GS8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
INITDONE_GS7	INITDONE_GS6	INITDONE_GS5	INITDONE_GS4	INITDONE_GS3	INITDONE_GS2	INITDONE_GS1	INITDONE_GS0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 3-276. GSxINITDONE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	INITDONE_GS15	R	0h	RAM Initialization status for GS15 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
14	INITDONE_GS14	R	0h	RAM Initialization status for GS14 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
13	INITDONE_GS13	R	0h	RAM Initialization status for GS13 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
12	INITDONE_GS12	R	0h	RAM Initialization status for GS12 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
11	INITDONE_GS11	R	0h	RAM Initialization status for GS11 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
10	INITDONE_GS10	R	0h	RAM Initialization status for GS10 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
9	INITDONE_GS9	R	0h	RAM Initialization status for GS9 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn

Table 3-276. GSxINITDONE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	INITDONE_GS8	R	0h	RAM Initialization status for GS8 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
7	INITDONE_GS7	R	0h	RAM Initialization status for GS7 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
6	INITDONE_GS6	R	0h	RAM Initialization status for GS6 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
5	INITDONE_GS5	R	0h	RAM Initialization status for GS5 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
4	INITDONE_GS4	R	0h	RAM Initialization status for GS4 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
3	INITDONE_GS3	R	0h	RAM Initialization status for GS3 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
2	INITDONE_GS2	R	0h	RAM Initialization status for GS2 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
1	INITDONE_GS1	R	0h	RAM Initialization status for GS1 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
0	INITDONE_GS0	R	0h	RAM Initialization status for GS0 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn

3.16.11.28 GSxRAMTEST_LOCK Register (Offset = 56h) [Reset = 0h]

GSxRAMTEST_LOCK is shown in Figure 3-261 and described in Table 3-277.

Return to the Summary Table.

Lock register to GSx RAM TEST registers

Figure 3-261. GSxRAMTEST_LOCK Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
GS15	GS14	GS13	GS12	GS11	GS10	GS9	GS8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GS7	GS6	GS5	GS4	GS3	GS2	GS1	GS0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-277. GSxRAMTEST_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	A value of 0xa5a5 to this field is simultaneously required for the writes to the rest of the fields of this register to succeed, Reset type: SYSRSn
15	GS15	R/W	0h	0: Allows writes to GSxTEST.TEST_GS15 field. 1: Blocks writes to GSxTEST.TEST_GS15 field. Reset type: SYSRSn
14	GS14	R/W	0h	0: Allows writes to GSxTEST.TEST_GS14 field. 1: Blocks writes to GSxTEST.TEST_GS14 field. Reset type: SYSRSn
13	GS13	R/W	0h	0: Allows writes to GSxTEST.TEST_GS13 field. 1: Blocks writes to GSxTEST.TEST_GS13 field. Reset type: SYSRSn
12	GS12	R/W	0h	0: Allows writes to GSxTEST.TEST_GS12 field. 1: Blocks writes to GSxTEST.TEST_GS12 field. Reset type: SYSRSn
11	GS11	R/W	0h	0: Allows writes to GSxTEST.TEST_GS11 field. 1: Blocks writes to GSxTEST.TEST_GS11 field. Reset type: SYSRSn
10	GS10	R/W	0h	0: Allows writes to GSxTEST.TEST_GS10 field. 1: Blocks writes to GSxTEST.TEST_GS10 field. Reset type: SYSRSn
9	GS9	R/W	0h	0: Allows writes to GSxTEST.TEST_GS9 field. 1: Blocks writes to GSxTEST.TEST_GS9 field. Reset type: SYSRSn
8	GS8	R/W	0h	0: Allows writes to GSxTEST.TEST_GS8 field. 1: Blocks writes to GSxTEST.TEST_GS8 field. Reset type: SYSRSn
7	GS7	R/W	0h	0: Allows writes to GSxTEST.TEST_GS7 field. 1: Blocks writes to GSxTEST.TEST_GS7 field. Reset type: SYSRSn

Table 3-277. GSxRAMTEST_LOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	GS6	R/W	0h	0: Allows writes to GSxTEST.TEST_GS6 field. 1: Blocks writes to GSxTEST.TEST_GS6 field. Reset type: SYSRSn
5	GS5	R/W	0h	0: Allows writes to GSxTEST.TEST_GS5 field. 1: Blocks writes to GSxTEST.TEST_GS5 field. Reset type: SYSRSn
4	GS4	R/W	0h	0: Allows writes to GSxTEST.TEST_GS4 field. 1: Blocks writes to GSxTEST.TEST_GS4 field. Reset type: SYSRSn
3	GS3	R/W	0h	0: Allows writes to GSxTEST.TEST_GS3 field. 1: Blocks writes to GSxTEST.TEST_GS3 field. Reset type: SYSRSn
2	GS2	R/W	0h	0: Allows writes to GSxTEST.TEST_GS2 field. 1: Blocks writes to GSxTEST.TEST_GS2 field. Reset type: SYSRSn
1	GS1	R/W	0h	0: Allows writes to GSxTEST.TEST_GS1 field. 1: Blocks writes to GSxTEST.TEST_GS1 field. Reset type: SYSRSn
0	GS0	R/W	0h	0: Allows writes to GSxTEST.TEST_GS0 field. 1: Blocks writes to GSxTEST.TEST_GS0 field. Reset type: SYSRSn

3.16.11.29 MSGxLOCK Register (Offset = 60h) [Reset = 0h]

MSGxLOCK is shown in Figure 3-262 and described in Table 3-278.

Return to the Summary Table.

Message RAM Config Lock Register

Figure 3-262. MSGxLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				LOCK_DMATOCLA2	LOCK_CLA2TODMA	LOCK_CPUTOCM_MSGRAM1	LOCK_CPUTOCM_MSGRAM0
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
LOCK_CPUTOCPU_MSGRAM1	LOCK_DMATOCLA1	LOCK_CLA1TODMA	RESERVED	RESERVED	LOCK_CLA1TOCPU	LOCK_CPUTOCLA1	LOCK_CPUTOCPU_MSGRAM0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-278. MSGxLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11	LOCK_DMATOCLA2	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for DMA2CLA MSG RAM: 0: Write to INIT fields are allowed. 1: Write to INIT fields are blocked. Reset type: SYSRSn
10	LOCK_CLA2TODMA	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for CLA2DMA MSG RAM: 0: Write to INIT fields are allowed. 1: Write to INIT fields are blocked. Reset type: SYSRSn
9	LOCK_CPUTOCM_MSGRAM1	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for CPU2CM MSG RAM1: 0: Write to INIT fields are allowed. 1: Write to INIT fields are blocked. Reset type: SYSRSn
8	LOCK_CPUTOCM_MSGRAM0	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for CPU2CM MSG RAM0: 0: Write to INIT fields are allowed. 1: Write to INIT fields are blocked. Reset type: SYSRSn
7	LOCK_CPUTOCPU_MSGRAM1	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for CPU2CPU MSG RAM1: 0: Write to INIT fields are allowed. 1: Write to INIT fields are blocked. Reset type: SYSRSn
6	LOCK_DMATOCLA1	R/W	0h	Locks the write to access protection, master select, initialization control and test for DMATOCLA1 RAM: 0: Write to INIT fields are allowed. 1: Write to INIT fields are blocked. Reset type: SYSRSn

Table 3-278. MSGxLOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	LOCK_CLA1TODMA	R/W	0h	Locks the write to access protection, master select, initialization control and test for CLA1TODMA RAM: 0: Write to INIT fields are allowed. 1: Write to INIT fields are blocked. Reset type: SYSRSn
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	LOCK_CLA1TOCPU	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for CLA1TOCPU RAM: 0: Write to INIT fields are allowed. 1: Write to INIT fields are blocked. Reset type: SYSRSn
1	LOCK_CPUTOCLA1	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for CPUTOCLA1 RAM: 0: Write to INIT fields are allowed. 1: Write to INIT fields are blocked. Reset type: SYSRSn
0	LOCK_CPUTOCPU_MSGRAM0	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for CPU2CPU MSG RAM0: 0: Write to ACCPROT, INIT fields are allowed. 1: Write to ACCPROT, INIT fields are blocked. Reset type: SYSRSn

3.16.11.30 MSGxCOMMIT Register (Offset = 62h) [Reset = 0h]

MSGxCOMMIT is shown in Figure 3-263 and described in Table 3-279.

Return to the Summary Table.

Message RAM Config Lock Commit Register

Figure 3-263. MSGxCOMMIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				COMMIT_DMATOCLA_MSGRAM1	COMMIT_CLATODMA_MSGRAM0	COMMIT_CPUTOCM_MSGRAM1	COMMIT_CPUTOCM_MSGRAM0
R-0h				R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
7	6	5	4	3	2	1	0
COMMIT_CPUTOCPU_MSGRAM1	COMMIT_DMATOCLA1	COMMIT_CLA1TODMA	RESERVED	RESERVED	COMMIT_CLA1TOCPU	COMMIT_CPUTOCLA1	COMMIT_CPUTOCPU_MSGRAM0
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 3-279. MSGxCOMMIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11	COMMIT_DMATOCLA_MSGRAM1	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for DMA2CLA MSG RAM: 0: Write to INIT fields are allowed. 1: Write to INIT fields are blocked. Reset type: SYSRSn
10	COMMIT_CLATODMA_MSGRAM0	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for CLA2DMA MSG RAM: 0: Write to INIT fields are allowed. 1: Write to INIT fields are blocked. Reset type: SYSRSn
9	COMMIT_CPUTOCM_MSGRAM1	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for CPU2CM MSG RAM1: 0: Write to INIT fields are allowed. 1: Write to INIT fields are blocked. Reset type: SYSRSn
8	COMMIT_CPUTOCM_MSGRAM0	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for CPU2CM MSG RAM0: 0: Write to INIT fields are allowed. 1: Write to INIT fields are blocked. Reset type: SYSRSn
7	COMMIT_CPUTOCPU_MSGRAM1	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for CPU2CPU MSG RAM1: 0: Write to INIT fields are allowed. 1: Write to INIT fields are blocked. Reset type: SYSRSn
6	COMMIT_DMATOCLA1	R/WSonce	0h	0: Write to, INIT fields are allowed. 1: Write to, INIT fields are blocked. Reset type: SYSRSn

Table 3-279. MSGxCOMMIT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	COMMIT_CLA1TODMA	R/WSonce	0h	0: Write to, INIT fields are allowed. 1: Write to, INIT fields are blocked. Reset type: SYSRSn
4	RESERVED	R/WSonce	0h	Reserved
3	RESERVED	R/WSonce	0h	Reserved
2	COMMIT_CLA1TOCPU	R/WSonce	0h	Locks the write to access protection, master select, initialization control and test register fields for CLA1TOCPU RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
1	COMMIT_CPUTOCLA1	R/WSonce	0h	Locks the write to access protection, master select, initialization control and test register fields for CPUTOCLA1 RAM: 0: Write to ACCPROT, INIT and Mselect fields are allowed. 1: Write to ACCPROT, INIT and Mselect fields are blocked. Reset type: SYSRSn
0	COMMIT_CPUTOCPU_MSGRAM0	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for D0 RAM: 0: Write to ACCPROT, INIT fields are allowed based on value of lock field in MSGxLOCK register. 1: Write to ACCPROT, INIT are permanently blocked. Reset type: SYSRSn

3.16.11.31 MSGxACCPROT0 Register (Offset = 68h) [Reset = 0h]

MSGxACCPROT0 is shown in Figure 3-264 and described in Table 3-280.

Return to the Summary Table.

Message RAM Access Protection Register 0

Figure 3-264. MSGxACCPROT0 Register

31	30	29	28	27	26	25	24
RESERVED					RESERVED	RESERVED	RESERVED
R-0h					R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED					RESERVED	RESERVED	RESERVED
R-0h					R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED					RESERVED	RESERVED	RESERVED
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED					DMAWRPROT_CPUTOCPU_MSGRAM0	CPUWRPROT_CPUTOCPU_MSGRAM0	RESERVED
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 3-280. MSGxACCPROT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R	0h	Reserved
26	RESERVED	R/W	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23-19	RESERVED	R	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15-11	RESERVED	R	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7-3	RESERVED	R	0h	Reserved
2	DMAWRPROT_CPUTOCPU_MSGRAM0	R/W	0h	DMA WR Protection For CPUTOCPU_MSGRAM0 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
1	CPUWRPROT_CPUTOCPU_MSGRAM0	R/W	0h	CPU WR Protection For CPUTOCPU_MSGRAM0 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
0	RESERVED	R/W	0h	Reserved

3.16.11.32 MSGxACCPROT1 Register (Offset = 6Ah) [Reset = 0h]

MSGxACCPROT1 is shown in Figure 3-265 and described in Table 3-281.

Return to the Summary Table.

Message RAM Access Protection Register 1

Figure 3-265. MSGxACCPROT1 Register

31	30	29	28	27	26	25	24
RESERVED					DMAWRPROT_CPUTOCPU_MSGRAM1	CPUWRPROT_CPUTOCPU_MSGRAM1	RESERVED
R-0h					R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED					RESERVED	RESERVED	RESERVED
R-0h					R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED					RESERVED	RESERVED	RESERVED
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED					RESERVED	RESERVED	RESERVED
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 3-281. MSGxACCPROT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R	0h	Reserved
26	DMAWRPROT_CPUTOCPU_MSGRAM1	R/W	0h	DMA WR Protection For CPUTOCPU_MSGRAM1 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
25	CPUWRPROT_CPUTOCPU_MSGRAM1	R/W	0h	CPU WR Protection For CPUTOCPU_MSGRAM1 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
24	RESERVED	R/W	0h	Reserved
23-19	RESERVED	R	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15-11	RESERVED	R	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7-3	RESERVED	R	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	RESERVED	R/W	0h	Reserved

3.16.11.33 MSGxACCPROT2 Register (Offset = 6Ch) [Reset = 0h]

MSGxACCPROT2 is shown in Figure 3-266 and described in Table 3-282.

Return to the Summary Table.

Message RAM Access Protection Register 2

Figure 3-266. MSGxACCPROT2 Register

31	30	29	28	27	26	25	24
RESERVED					RESERVED	RESERVED	RESERVED
R-0h					R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED					RESERVED	RESERVED	RESERVED
R-0h					R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED					DMAWRPROT_CPUTOCM_MSGRAM1	CPUWRPROT_CPUTOCM_MSGRAM1	RESERVED
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED					DMAWRPROT_CPUTOCM_MSGRAM0	CPUWRPROT_CPUTOCM_MSGRAM0	RESERVED
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 3-282. MSGxACCPROT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R	0h	Reserved
26	RESERVED	R/W	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23-19	RESERVED	R	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15-11	RESERVED	R	0h	Reserved
10	DMAWRPROT_CPUTOCM_MSGRAM1	R/W	0h	DMA WR Protection For CPUTOCM_MSGRAM1 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
9	CPUWRPROT_CPUTOCM_MSGRAM1	R/W	0h	CPU WR Protection For CPUTOCM_MSGRAM1 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
8	RESERVED	R/W	0h	Reserved
7-3	RESERVED	R	0h	Reserved
2	DMAWRPROT_CPUTOCM_MSGRAM0	R/W	0h	DMA WR Protection For CPUTOCM_MSGRAM0 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn

Table 3-282. MSGxACCPROT2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	CPUWRPROT_CPUTOCM_MSGRAM0	R/W	0h	CPU WR Protection For CPUTOCM_MSGRAM0 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
0	RESERVED	R/W	0h	Reserved

3.16.11.34 MSGxTEST Register (Offset = 70h) [Reset = 0h]

MSGxTEST is shown in Figure 3-267 and described in Table 3-283.

Return to the Summary Table.

Message RAM TEST Register

Figure 3-267. MSGxTEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED		RESERVED		TEST_CPUTOCM_MSGRAM1		TEST_CPUTOCM_MSGRAM0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
TEST_CPUTOCPU_MSGRAM1		TEST_DMATOCLA1		TEST_CLA1TODMA		RESERVED	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		TEST_CLA1TOCPU		TEST_CPUTOCLA1		TEST_CPUTOCPU_MSGRAM0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 3-283. MSGxTEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-22	RESERVED	R/W	0h	Reserved
21-20	RESERVED	R/W	0h	Reserved
19-18	TEST_CPUTOCM_MSGRAM1	R/W	0h	Selects the defferent modes for CPUTOCM MSG RAM1: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
17-16	TEST_CPUTOCM_MSGRAM0	R/W	0h	Selects the defferent modes for CPUTOCM MSG RAM0: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
15-14	TEST_CPUTOCPU_MSGRAM1	R/W	0h	Selects the defferent modes for CPUTOCPU MSG RAM0: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn

Table 3-283. MSGxTEST Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13-12	TEST_DMATOCLA1	R/W	0h	Selects the defferent modes for DMATOCLA1 MSG RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
11-10	TEST_CLA1TODMA	R/W	0h	Selects the defferent modes for CLA1TODMA MSG RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
9-8	RESERVED	R/W	0h	Reserved
7-6	RESERVED	R/W	0h	Reserved
5-4	TEST_CLA1TOCPU	R/W	0h	Selects the defferent modes for CLA1TOCPU MSG RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
3-2	TEST_CPUTOCLA1	R/W	0h	Selects the defferent modes for CPUTOCLA1 MSG RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
1-0	TEST_CPUTOCPU_MSGRAM0	R/W	0h	Selects the defferent modes for CPUTOCPU MSG RAM0: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to parity bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/nmi is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn

3.16.11.35 MSGxINIT Register (Offset = 72h) [Reset = 0h]

MSGxINIT is shown in Figure 3-268 and described in Table 3-284.

Return to the Summary Table.

Message RAM Init Register

Figure 3-268. MSGxINIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				RESERVED	RESERVED	INIT_CPUTOCM_MSGRAM1	INIT_CPUTOCM_MSGRAM0
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
INIT_CPUTOCPU_MSGRAM1	INIT_DMATOCLA1	INIT_CLA1TODMA	RESERVED	RESERVED	INIT_CLA1TOCPU	INIT_CPUTOCLA1	INIT_CPUTOCPU_MSGRAM0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-284. MSGxINIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11	RESERVED	R-0/W1S	0h	Reserved
10	RESERVED	R-0/W1S	0h	Reserved
9	INIT_CPUTOCM_MSGRAM1	R-0/W1S	0h	RAM Initialization control for CPUTOCM MSG RAM1: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
8	INIT_CPUTOCM_MSGRAM0	R-0/W1S	0h	RAM Initialization control for CPUTOCM MSG RAM0: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
7	INIT_CPUTOCPU_MSGRAM1	R-0/W1S	0h	RAM Initialization control for CPUTOCPU MSG RAM1: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
6	INIT_DMATOCLA1	R-0/W1S	0h	RAM Initialization control for DMATOCLA1 MSG RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
5	INIT_CLA1TODMA	R-0/W1S	0h	RAM Initialization control for CLA1TODMA MSG RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
4	RESERVED	R-0/W1S	0h	Reserved
3	RESERVED	R-0/W1S	0h	Reserved
2	INIT_CLA1TOCPU	R-0/W1S	0h	RAM Initialization control for CLA1TOCPU MSG RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn

Table 3-284. MSGxINIT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	INIT_CPUTOCLA1	R-0/W1S	0h	RAM Initialization control for CPUTOCLA1 MSG RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
0	INIT_CPUTOCPU_MSGRAM0	R-0/W1S	0h	RAM Initialization control for CPUTOCPU MSG RAM0: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn

3.16.11.36 MSGxINITDONE Register (Offset = 74h) [Reset = 0h]

MSGxINITDONE is shown in Figure 3-269 and described in Table 3-285.

Return to the Summary Table.

Message RAM InitDone Status Register

Figure 3-269. MSGxINITDONE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				RESERVED	RESERVED	INITDONE_CPUTOCM_MSGRAM1	INITDONE_CPUTOCM_MSGRAM0
R-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
INITDONE_CPUTOCPU_MSGRAM1	INITDONE_DMATOCLA1	INITDONE_CLA1TODMA	RESERVED	RESERVED	INITDONE_CLA1TOCPU	INITDONE_CPUTOCLA1	INITDONE_CPUTOCPU_MSGRAM0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 3-285. MSGxINITDONE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9	INITDONE_CPUTOCM_MSGRAM1	R	0h	RAM Initialization status for CPUTOCM MSG RAM1: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
8	INITDONE_CPUTOCM_MSGRAM0	R	0h	RAM Initialization status for CPUTOCM MSG RAM0: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
7	INITDONE_CPUTOCPU_MSGRAM1	R	0h	RAM Initialization status for CPUTOCPU MSG RAM1: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
6	INITDONE_DMATOCLA1	R	0h	RAM Initialization status for DMATOCLA1 MSG RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
5	INITDONE_CLA1TODMA	R	0h	RAM Initialization status for CLA1TODMA MSG RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved

Table 3-285. MSGxINITDONE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	INITDONE_CLA1TOCPU	R	0h	RAM Initialization status for CLA1TOCPU MSG RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
1	INITDONE_CPUTOCLA1	R	0h	RAM Initialization status for CPUTOCLA1 MSG RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
0	INITDONE_CPUTOCPU_MSGRAM0	R	0h	RAM Initialization status for CPUTOCPU MSG RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn

3.16.11.37 MSGxRAMTEST_LOCK Register (Offset = 76h) [Reset = 0h]

MSGxRAMTEST_LOCK is shown in Figure 3-270 and described in Table 3-286.

Return to the Summary Table.

Lock register to MSGx RAM TEST registers

Figure 3-270. MSGxRAMTEST_LOCK Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED				DMATOCLA2	CLA2TODMA	CPUTOCM_MSGRAM1	CPUTOCM_MSGRAM0
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CPUTOCPU_MSGRAM1	DMATOCLA1	CLA1TODMA	CLA2TOCPU	CPUTOCLA2	CLA1TOCPU	CPUTOCLA1	CPUTOCPU_MSGRAM0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-286. MSGxRAMTEST_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	A value of 0xa5a5 to this field is simultaneously required for the writes to the rest of the fields of this register to succeed, Reset type: SYSRSn
15-12	RESERVED	R	0h	Reserved
11	DMATOCLA2	R/W	0h	0: Allows writes to MSGxTEST.TEST_DMATOCLA2 field. 1: Blocks writes to MSGxTEST.TEST_DMATOCLA2 field. Reset type: SYSRSn
10	CLA2TODMA	R/W	0h	0: Allows writes to MSGxTEST.TEST_CLA2TODMA field. 1: Blocks writes to MSGxTEST.TEST_CLA2TODMA field. Reset type: SYSRSn
9	CPUTOCM_MSGRAM1	R/W	0h	0: Allows writes to MSGxTEST.TEST_CPUTOCM_MSGRAM1 field. 1: Blocks writes to MSGxTEST.TEST_CPUTOCM_MSGRAM1 field. Reset type: SYSRSn
8	CPUTOCM_MSGRAM0	R/W	0h	0: Allows writes to MSGxTEST.TEST_CPUTOCM_MSGRAM0 field. 1: Blocks writes to MSGxTEST.TEST_CPUTOCM_MSGRAM0 field. Reset type: SYSRSn
7	CPUTOCPU_MSGRAM1	R/W	0h	0: Allows writes to MSGxTEST.TEST_CPUTOCPU_MSGRAM1 field. 1: Blocks writes to MSGxTEST.TEST_CPUTOCPU_MSGRAM1 field. Reset type: SYSRSn
6	DMATOCLA1	R/W	0h	0: Allows writes to MSGxTEST.TEST_DMATOCLA1 field. 1: Blocks writes to MSGxTEST.TEST_DMATOCLA1 field. Reset type: SYSRSn
5	CLA1TODMA	R/W	0h	0: Allows writes to MSGxTEST.TEST_CLA1TODMA field. 1: Blocks writes to MSGxTEST.TEST_CLA1TODMA field. Reset type: SYSRSn
4	CLA2TOCPU	R/W	0h	0: Allows writes to MSGxTEST.TEST_CLA2TOCPU field. 1: Blocks writes to MSGxTEST.TEST_CLA2TOCPU field. Reset type: SYSRSn

Table 3-286. MSGxRAMTEST_LOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	CPUTOCLA2	R/W	0h	0: Allows writes to MSGxTEST.TEST_CPUTOCLA2 field. 1: Blocks writes to MSGxTEST.TEST_CPUTOCLA2 field. Reset type: SYSRSn
2	CLA1TOCPU	R/W	0h	0: Allows writes to MSGxTEST.TEST_CLA1TOCPU field. 1: Blocks writes to MSGxTEST.TEST_CLA1TOCPU field. Reset type: SYSRSn
1	CPUTOCLA1	R/W	0h	0: Allows writes to MSGxTEST.TEST_CPUTOCLA1 field. 1: Blocks writes to MSGxTEST.TEST_CPUTOCLA1 field. Reset type: SYSRSn
0	CPUTOCPU_MSGRAM0	R/W	0h	0: Allows writes to MSGxTEST.TEST_CPUTOCPU_MSGRAM0 field. 1: Blocks writes to MSGxTEST.TEST_CPUTOCPU_MSGRAM0 field. Reset type: SYSRSn

3.16.11.38 ROM_LOCK Register (Offset = A0h) [Reset = 0h]

ROM_LOCK is shown in Figure 3-271 and described in Table 3-287.

Return to the Summary Table.

ROM Config Lock Register

Figure 3-271. ROM_LOCK Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					LOCK_CLADATAROM	LOCK_SECUREROM	LOCK_BOOTROM
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 3-287. ROM_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	A value of 0xa5a5 to this field is simultaneously required for the writes to the rest of the fields of this register to succeed, Reset type: SYSRSn
15-3	RESERVED	R	0h	Reserved
2	LOCK_CLADATAROM	R/W	0h	Locks write access to test control fields (TEST and FORCE_ERROR) of CLADATAROM 0: Write access allowed 1: Write access blocked Reset type: SYSRSn
1	LOCK_SECUREROM	R/W	0h	Locks write access to test control fields (TEST and FORCE_ERROR) of SECUREROM 0: Write access allowed 1: Write access blocked Reset type: SYSRSn
0	LOCK_BOOTROM	R/W	0h	Locks write access to test control fields (TEST and FORCE_ERROR) of BOOTROM 0: Write access allowed 1: Write access blocked Reset type: SYSRSn

3.16.11.39 ROM_TEST Register (Offset = A2h) [Reset = 0h]

ROM_TEST is shown in Figure 3-272 and described in Table 3-288.

Return to the Summary Table.

ROM TEST Register

Figure 3-272. ROM_TEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		TEST_CLADATAROM		TEST_SECUREROM		TEST_BOOTROM	
R-0h		R/W-0h		R/W-0h		R/W-0h	

Table 3-288. ROM_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5-4	TEST_CLADATAROM	R/W	0h	Selects the different modes for CLADATAROM: 00: Functional Mode. 01: same as "00" but Parity check on data read is disabled (for debug) 10: Parity Bits are visible on memory map (for debug) 11: Same as "00" but NMI is not generated on errors, used for diagnostics. (for diagnostics) Reset type: SYSRSn
3-2	TEST_SECUREROM	R/W	0h	Selects the different modes for SECUREROM: 00: Functional Mode. 01: same as "00" but Parity check on data read is disabled (for debug) 10: Parity Bits are visible on memory map (for debug) 11: Same as "00" but NMI is not generated on errors, used for diagnostics. (for diagnostics) Reset type: SYSRSn
1-0	TEST_BOOTROM	R/W	0h	Selects the different modes for BOOTROM: 00: Functional Mode. 01: same as "00" but Parity check on data read is disabled (for debug) 10: Parity Bits are visible on memory map (for debug) 11: Same as "00" but NMI is not generated on errors, used for diagnostics. (for diagnostics) Reset type: SYSRSn

3.16.11.40 ROM_FORCE_ERROR Register (Offset = A4h) [Reset = 0h]

ROM_FORCE_ERROR is shown in Figure 3-273 and described in Table 3-289.

Return to the Summary Table.

ROM Force Error register

Figure 3-273. ROM_FORCE_ERROR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					FORCE_CLADATAROM_ERROR	FORCE_SECUREROM_ERROR	FORCE_BOOTROM_ERROR
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 3-289. ROM_FORCE_ERROR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-3	RESERVED	R	0h	Reserved
2	FORCE_CLADATAROM_ERROR	R/W	0h	Force parity error by feeding inverted Parity bit to Parity checking logic. Reset type: SYSRSn
1	FORCE_SECUREROM_ERROR	R/W	0h	Force parity error by feeding inverted Parity bit to Parity checking logic. Reset type: SYSRSn
0	FORCE_BOOTROM_ERROR	R/W	0h	Force parity error by feeding inverted Parity bit to Parity checking logic. Reset type: SYSRSn

3.16.11.41 PERI_MEM_TEST_LOCK Register (Offset = AAh) [Reset = 0h]

PERI_MEM_TEST_LOCK is shown in Figure 3-274 and described in Table 3-290.

Return to the Summary Table.

Peripheral Memory Test Lock Register

Figure 3-274. PERI_MEM_TEST_LOCK Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK_PERI_MEM_TEST_CONTROL
R-0h							R/W-0h

Table 3-290. PERI_MEM_TEST_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	A value of 0xa5a5 to this field is simultaneously required for the writes to the rest of the fields of this register to succeed, Reset type: SYSRSn
15-1	RESERVED	R	0h	Reserved
0	LOCK_PERI_MEM_TEST_CONTROL	R/W	0h	Locks write access to register PERI_MEM_TEST_CONTROL 0: Write access allowed 1: Write access blocked Reset type: SYSRSn

3.16.11.42 PERI_MEM_TEST_CONTROL Register (Offset = ACh) [Reset = 0h]

PERI_MEM_TEST_CONTROL is shown in Figure 3-275 and described in Table 3-291.

Return to the Summary Table.

Peripheral Memory Test control Register

Figure 3-275. PERI_MEM_TEST_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		EtherCAT_MEM_FORCE_ERROR	EtherCAT_TEST_ENABLE	RESERVED	RESERVED	RESERVED	RESERVED
R-0h		R/W-0h	R/W-0h	R-0h	R-0h	R-0h	R-0h

Table 3-291. PERI_MEM_TEST_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	EtherCAT_MEM_FORCE_ERROR	R/W	0h	Force error bit 0 : No effect 1 : Parity bit going to Parity checker module of EtherCAT is inverted to introduce parity Error Reset type: SYSRSn
4	EtherCAT_TEST_ENABLE	R/W	0h	Selects EtherCAT test mode 0 : EtherCAT test mode disabled, Error on EtherCAT memory read access will generate NMI 1 : EtherCAT test mode enabled, Error on EtherCAT memory read access will NOT generate NMI, used for diagnostics Reset type: SYSRSn
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

3.16.12 MEMORY_ERROR_REGS Registers

Table 3-292 lists the memory-mapped registers for the MEMORY_ERROR_REGS registers. All register offset addresses not listed in Table 3-292 should be considered as reserved locations and the register contents should not be modified.

Table 3-292. MEMORY_ERROR_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	UCERRFLG	Uncorrectable Error Flag Register		Go
2h	UCERRSET	Uncorrectable Error Flag Set Register	EALLOW	Go
4h	UCERRCLR	Uncorrectable Error Flag Clear Register	EALLOW	Go
6h	UCCPUREADDR	Uncorrectable CPU Read Error Address		Go
8h	UCDMAREADDR	Uncorrectable DMA Read Error Address		Go
Ah	UCCLA1READDR	Uncorrectable CLA1 Read Error Address		Go
Eh	UCECATRAMADDR	Uncorrectable etherCAT RAM Read Error Address		Go
20h	CERRFLG	Correctable Error Flag Register		Go
22h	CERRSET	Correctable Error Flag Set Register	EALLOW	Go
24h	CERRCLR	Correctable Error Flag Clear Register	EALLOW	Go
26h	CCPUREADDR	Correctable CPU Read Error Address		Go
2Ah	CCLA1READDR	Correctable CLA1 Read Error Address		Go
2Eh	CERRCNT	Correctable Error Count Register		Go
30h	CERRTHRES	Correctable Error Threshold Value Register	EALLOW	Go
32h	CEINTFLG	Correctable Error Interrupt Flag Status Register		Go
34h	CEINTCLR	Correctable Error Interrupt Flag Clear Register	EALLOW	Go
36h	CEINTSET	Correctable Error Interrupt Flag Set Register	EALLOW	Go
38h	CEINTEN	Correctable Error Interrupt Enable Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-293 shows the codes that are used for access types in this section.

Table 3-293. MEMORY_ERROR_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 3-293. MEMORY_ERROR_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.12.1 UCERRFLG Register (Offset = 0h) [Reset = 0h]

UCERRFLG is shown in Figure 3-276 and described in Table 3-294.

Return to the Summary Table.

Uncorrectable Error Flag Register

Figure 3-276. UCERRFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			ECATRAMRDERR	RESERVED	CLA1RDERR	DMARDERR	CPURDERR
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h

Table 3-294. UCERRFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-5	RESERVED	R	0h	Reserved
4	ECATRAMRDERR	R	0h	ECAT RAM Read Error Flag 0: No Error. 1: Uncorrectable error occurred on etherCAT RAM. Reset type: SYSRSn
3	RESERVED	R	0h	Reserved
2	CLA1RDERR	R	0h	CLA1 Uncorrectable Read Error Flag 0: No Error. 1: Uncorrectable error occurred during CLA1 read. Reset type: SYSRSn
1	DMARDERR	R	0h	DMA Uncorrectable Read Error Flag 0: No Error. 1: Uncorrectable error occurred during DMA read. Reset type: SYSRSn
0	CPURDERR	R	0h	CPU Uncorrectable Read Error Flag 0: No Error. 1: Uncorrectable error occurred during CPU read. Reset type: SYSRSn

3.16.12.2 UCERRSET Register (Offset = 2h) [Reset = 0h]

UCERRSET is shown in Figure 3-277 and described in Table 3-295.

Return to the Summary Table.

Uncorrectable Error Flag Set Register

Figure 3-277. UCERRSET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			ECATRAMRDERR	RESERVED	CLA1RDERR	DMARDERR	CPURDERR
R-0h			R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-295. UCERRSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-5	RESERVED	R	0h	Reserved
4	ECATRAMRDERR	R-0/W1S	0h	ECAT RAM Read Error Flag 0: No Action. 1: ECATRAMRDERR Flag in UCERRFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
3	RESERVED	R-0/W1S	0h	Reserved
2	CLA1RDERR	R-0/W1S	0h	0: No action. 1: CLA1 Read Error Flag in UCERRFLG register will be set and interrupt will be generated if enabled.. Reset type: SYSRSn
1	DMARDERR	R-0/W1S	0h	0: No action. 1: DMA Read Error Flag in UCERRFLG register will be set and interrupt will be generated if enabled.. Reset type: SYSRSn
0	CPURDERR	R-0/W1S	0h	0: No action. 1: CPU Read Error Flag in UCERRFLG register will be set and interrupt will be generated if enabled.. Reset type: SYSRSn

3.16.12.3 UCERRCLR Register (Offset = 4h) [Reset = 0h]

UCERRCLR is shown in Figure 3-278 and described in Table 3-296.

Return to the Summary Table.

Uncorrectable Error Flag Clear Register

Figure 3-278. UCERRCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			ECATRAMRDERR	RESERVED	CLA1RDERR	DMARDERR	CPURDERR
R-0h			R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-296. UCERRCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-5	RESERVED	R	0h	Reserved
4	ECATRAMRDERR	R-0/W1S	0h	ECAT RAM Read Error Flag 0: No action. 1: ECATRAMRDERR Flag in UCERRFLG register will be cleared. Reset type: SYSRSn
3	RESERVED	R-0/W1S	0h	Reserved
2	CLA1RDERR	R-0/W1S	0h	0: No action. 1: CLA1 Read Error Flag in UCERRFLG register will be cleared. Reset type: SYSRSn
1	DMARDERR	R-0/W1S	0h	0: No action. 1: DMA Read Error Flag in UCERRFLG register will be cleared . Reset type: SYSRSn
0	CPURDERR	R-0/W1S	0h	0: No action. 1: CPU Read Error Flag in UCERRFLG register will be cleared. Reset type: SYSRSn

3.16.12.4 UCCPUREADDR Register (Offset = 6h) [Reset = 0h]

UCCPUREADDR is shown in Figure 3-279 and described in Table 3-297.

Return to the Summary Table.

Uncorrectable CPU Read Error Address

Figure 3-279. UCCPUREADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCCPUREADDR																															
R-0h																															

Table 3-297. UCCPUREADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UCCPUREADDR	R	0h	This register captures the address location for which CPU read/fetch access resulted in uncorrectable ECC/Parity error. Reset type: SYSRSn

3.16.12.5 UCDMAREADDR Register (Offset = 8h) [Reset = 0h]

UCDMAREADDR is shown in Figure 3-280 and described in Table 3-298.

Return to the Summary Table.

Uncorrectable DMA Read Error Address

Figure 3-280. UCDMAREADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCDMAREADDR																															
R-0h																															

Table 3-298. UCDMAREADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UCDMAREADDR	R	0h	This register captures the address location for which DMA read access resulted in uncorrectable Parity error. Reset type: SYSRSn

3.16.12.6 UCCLA1READDR Register (Offset = Ah) [Reset = 0h]

UCCLA1READDR is shown in Figure 3-281 and described in Table 3-299.

Return to the Summary Table.

Uncorrectable CLA1 Read Error Address

Figure 3-281. UCCLA1READDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCCLA1READDR																															
R-0h																															

Table 3-299. UCCLA1READDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UCCLA1READDR	R	0h	This register captures the address location for which CLA1 read/ fetch access resulted in uncorrectable Parity error. Reset type: SYSRSn

3.16.12.7 UCECATRAMADDR Register (Offset = Eh) [Reset = 0h]

UCECATRAMADDR is shown in Figure 3-282 and described in Table 3-300.

Return to the Summary Table.

Uncorrectable etherCAT RAM Read Error Address

Figure 3-282. UCECATRAMADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCECATRAMADDR																															
R-0h																															

Table 3-300. UCECATRAMADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UCECATRAMADDR	R	0h	This register captures the address offset of the etherCAT RAM location for which read access (access could be from etherCAT master or from CPU/DMA) resulted in uncorrectable Parity error. Reset type: SYSRSn

3.16.12.8 CERRFLG Register (Offset = 20h) [Reset = 0h]

CERRFLG is shown in Figure 3-283 and described in Table 3-301.

Return to the Summary Table.

Correctable Error Flag Register

Figure 3-283. CERRFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	CLA1RDERR	DMARDERR	CPURDERR
R-0h				R-0h	R-0h	R-0h	R-0h

Table 3-301. CERRFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	CLA1RDERR	R	0h	CLA1 Correctable Read Error Flag 0: No Error. 1: Correctable error occurred during CLA1 read. Reset type: SYSRSn
1	DMARDERR	R	0h	DMA Correctable Read Error Flag 0: No Error. 1: Correctable error occurred during DMA read. Reset type: SYSRSn
0	CPURDERR	R	0h	CPU Correctable Read Error Flag 0: No Error. 1: Correctable error occurred during CPU read. Reset type: SYSRSn

3.16.12.9 CERRSET Register (Offset = 22h) [Reset = 0h]

CERRSET is shown in Figure 3-284 and described in Table 3-302.

Return to the Summary Table.

Correctable Error Flag Set Register

Figure 3-284. CERRSET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	CLA1RDERR	DMARDERR	CPURDERR
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-302. CERRSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	RESERVED	R-0/W1S	0h	Reserved
2	CLA1RDERR	R-0/W1S	0h	0: No action. 1: CLA1 Read Error Flag in CERRFLG register will be set and interrupt will be generated if enabled.. Reset type: SYSRSn
1	DMARDERR	R-0/W1S	0h	0: No action. 1: DMA Read Error Flag in CERRFLG register will be set and interrupt will be generated if enabled.. Reset type: SYSRSn
0	CPURDERR	R-0/W1S	0h	0: No action. 1: CPU Read Error Flag in CERRFLG register will be set and interrupt will be generated if enabled.. Reset type: SYSRSn

3.16.12.10 CERRCLR Register (Offset = 24h) [Reset = 0h]

CERRCLR is shown in Figure 3-285 and described in Table 3-303.

Return to the Summary Table.

Correctable Error Flag Clear Register

Figure 3-285. CERRCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	CLA1RDERR	DMARDERR	CPURDERR
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-303. CERRCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	RESERVED	R-0/W1S	0h	Reserved
2	CLA1RDERR	R-0/W1S	0h	0: No action. 1: CLA1 Read Error Flag in CERRFLG register will be cleared. Reset type: SYSRSn
1	DMARDERR	R-0/W1S	0h	0: No action. 1: DMA Read Error Flag in CERRFLG register will be cleared . Reset type: SYSRSn
0	CPURDERR	R-0/W1S	0h	0: No action. 1: CPU Read Error Flag in CERRFLG register will be cleared. Reset type: SYSRSn

3.16.12.11 CCPUREADDR Register (Offset = 26h) [Reset = 0h]

CCPUREADDR is shown in Figure 3-286 and described in Table 3-304.

Return to the Summary Table.

Correctable CPU Read Error Address

Figure 3-286. CCPUREADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCPUREADDR																															
R-0h																															

Table 3-304. CCPUREADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CCPUREADDR	R	0h	This register captures the address location for which CPU read/fetch access resulted in correctable ECC error. Reset type: SYSRSn

3.16.12.12 CCLA1READDR Register (Offset = 2Ah) [Reset = 0h]

CCLA1READDR is shown in Figure 3-287 and described in Table 3-305.

Return to the Summary Table.

Correctable CLA1 Read Error Address

Figure 3-287. CCLA1READDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCLA1READDR																															
R-0h																															

Table 3-305. CCLA1READDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CCLA1READDR	R	0h	This register captures the address location for which CLA1 read/ fetch access resulted in correctable ECC error. Reset type: SYSRSn

3.16.12.13 CERRCNT Register (Offset = 2Eh) [Reset = 0h]

CERRCNT is shown in Figure 3-288 and described in Table 3-306.

Return to the Summary Table.

Correctable Error Count Register

Figure 3-288. CERRCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CERRCNT															
R-0h																R/W-0h															

Table 3-306. CERRCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	CERRCNT	R/W	0h	This register holds the count of how many times correctable error occurred. Reset type: SYSRSn

3.16.12.14 CERRTHRES Register (Offset = 30h) [Reset = 0h]

CERRTHRES is shown in Figure 3-289 and described in Table 3-307.

Return to the Summary Table.

Correctable Error Threshold Value Register

Figure 3-289. CERRTHRES Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CERRTHRES															
R-0h																R/W-0h															

Table 3-307. CERRTHRES Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	CERRTHRES	R/W	0h	When value in CERRCNT register is greater than value configured in this register, corretable interrupt gets generated, if enabled. Reset type: SYSRSn

3.16.12.15 CEINTFLG Register (Offset = 32h) [Reset = 0h]

CEINTFLG is shown in Figure 3-290 and described in Table 3-308.

Return to the Summary Table.

Correctable Error Interrupt Flag Status Register

Figure 3-290. CEINTFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CEINTFLAG
R-0h							R-0h

Table 3-308. CEINTFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	CEINTFLAG	R	0h	Total corrected error count exceeded threshold Flag 0: Total correctable errors < Threshold value configured in CERRTHRES register. 1: Total correctable errors >= Threshold value configured in CERRTHRES register. Reset type: SYSRSn

3.16.12.16 CEINTCLR Register (Offset = 34h) [Reset = 0h]

CEINTCLR is shown in Figure 3-291 and described in Table 3-309.

Return to the Summary Table.

Correctable Error Interrupt Flag Clear Register

Figure 3-291. CEINTCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CEINTCLR
R-0h							R-0/W1S-0h

Table 3-309. CEINTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	CEINTCLR	R-0/W1S	0h	0: No action. 1: Total corrected error count exceeded flag in CEINTFLG register will be cleared. Reset type: SYSRSn

3.16.12.17 CEINTSET Register (Offset = 36h) [Reset = 0h]

CEINTSET is shown in Figure 3-292 and described in Table 3-310.

Return to the Summary Table.

Correctable Error Interrupt Flag Set Register

Figure 3-292. CEINTSET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CEINTSET
R-0h							R-0/W1S-0h

Table 3-310. CEINTSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	CEINTSET	R-0/W1S	0h	0: No action. 1: Total corrected error count exceeded flag in CEINTFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn

3.16.12.18 CEINTEN Register (Offset = 38h) [Reset = 0h]

CEINTEN is shown in Figure 3-293 and described in Table 3-311.

Return to the Summary Table.

Correctable Error Interrupt Enable Register

Figure 3-293. CEINTEN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CEINTEN
R-0h							R/W-0h

Table 3-311. CEINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	CEINTEN	R/W	0h	0: Correctable Error Interrupt is disabled. 1: Correctable Error Interrupt is enabled. Reset type: SYSRSn

3.16.13 NMI_INTRUPT_REGS Registers

Table 3-312 lists the memory-mapped registers for the NMI_INTRUPT_REGS registers. All register offset addresses not listed in Table 3-312 should be considered as reserved locations and the register contents should not be modified.

Table 3-312. NMI_INTRUPT_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	NMICFG	NMI Configuration Register	EALLOW	Go
1h	NMIFLG	NMI Flag Register (SYSRsn Clear)		Go
2h	NMIFLGCLR	NMI Flag Clear Register	EALLOW	Go
3h	NMIFLGFRC	NMI Flag Force Register	EALLOW	Go
4h	NMIWDCNT	NMI Watchdog Counter Register		Go
5h	NMIWDPRD	NMI Watchdog Period Register	EALLOW	Go
6h	NMISHDFLG	NMI Shadow Flag Register		Go
7h	ERRORSTS	Error pin status		Go
8h	ERRORSTSCLR	ERRORSTS clear register	EALLOW	Go
9h	ERRORSTSFRC	ERRORSTS force register	EALLOW	Go
Ah	ERRORCTL	Error pin control register	EALLOW	Go
Bh	ERRORLOCK	Lock register to Error pin registers.	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-313 shows the codes that are used for access types in this section.

Table 3-313. NMI_INTRUPT_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.13.1 NMICFG Register (Offset = 0h) [Reset = 0h]

NMICFG is shown in Figure 3-294 and described in Table 3-314.

Return to the Summary Table.

NMI Configuration Register

Figure 3-294. NMICFG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED							NMIE
R-0-0h							R/W1S-0h

Table 3-314. NMICFG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R-0	0h	Reserved
0	NMIE	R/W1S	0h	When set to 1 any condition will generate an NMI interrupt to the C28 CPU and kick off the NMI watchdog counter. As part of boot sequence this bit should be set after the device security related initialization is complete. 0 NMI disabled 1 NMI enabled Reset type: SYSRSn

3.16.13.2 NMIFLG Register (Offset = 1h) [Reset = 0h]

NMIFLG is shown in Figure 3-295 and described in Table 3-315.

Return to the Summary Table.

NMI Flag Register (SYSRsn Clear)

Figure 3-295. NMIFLG Register

15	14	13	12	11	10	9	8
MCAN_ERR	CRC_FAIL	ECATNMIn	CMNMIWDRSn	RESERVED	CPU2NMIWDRSn	CPU2WDRSn	CLBNMI
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
ERADNMI	PIEVECTERR	CPU2HWBISTERR	CPU1HWBISTERR	FLUNCERR	RAMUNCERR	CLOCKFAIL	NMIINT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 3-315. NMIFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15	MCAN_ERR	R	0h	0 MCAN module has not generated an ECC error. 1 MCAN module has generated an ECC error. [1] This bit is reserved for CPU2.NMIFLG register Reset type: SYSRSn
14	CRC_FAIL	R	0h	0 CPUCRC and CLACRC check has not failed. 1 CPUCRC or CLACRC check has failed. Reset type: SYSRSn
13	ECATNMIn	R	0h	0 No reset request from EtherCAT IP. 1 NMI generated from EtherCAT IP. No further NMI pulses are generated until this flag is cleared by the user. [1] This bit is reserved for CPU2.NMIFLG register Reset type: SYSRSn
12	CMNMIWDRSn	R	0h	CM NMIWDRSn Reset Indication Flag: This bit indicates if CM's NMIWDRSn was fired or not. 0 No CM.NMIWDRsn was fired 1 CM.NMIWDRSn was fired to CM [1] This bit is reserved for CPU2.NMIFLG register Reset type: SYSRSn
11	RESERVED	R	0h	Reserved
10	CPU2NMIWDRSn	R	0h	CPU2 NMIWDRSn Reset Indication Flag: This bit indicates if CPU2's NMIWDRSn was fired or not. 0 No CPU2.NMIWDRsn was fired 1 CPU2.NMIWDRSn was fired to CPU2 Note: [1] This bit is reserved for CPU2.NMIFLG register Reset type: SYSRSn
9	CPU2WDRSn	R	0h	CPU2 WDRSn Reset Indication Flag: This bit indicates if CPU2's WDRSn was fired or not. 0 No CPU2.WDRsn was fired 1 CPU2.WDRSn was fired to CPU2 Note: [1] This bit is reserved for CPU2.NMIFLG register Reset type: SYSRSn
8	CLBNMI	R	0h	Configurable Logic Block NMI Flag: This bit indicates if an NMI was generated by the Configurable Logic Block. This bit can only be cleared by the user writing to the corresponding clear bit in the NMIFLGCLR register or by an SYSRSn reset: 0,No Configurable Logic Block NMI pending 1,Configurable Logic Block NMI generated Reset type: SYSRSn

Table 3-315. NMIFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	ERADNMI	R	0h	ERAD Module NMI Flag: This bit indicates if an NMI was generated by the ERAD Module. This bit can only be cleared by the user writing to the corresponding clear bit in the NMIFLGCLR register or by an SYSRSn reset: 0,No ERAD NMI pending 1,ERAD NMI generated Reset type: SYSRSn
6	PIEVECTERR	R	0h	PIE Vector Fetch Error Flag: This bit indicates if an error occurred on an Vector Fecth by the CPU in the device. In Dual core system CPU1.NMIWD gets an NMI on an Vector fetch Error on CPU2. This bit can only be cleared by the user writing to the corresponding clear bit in the NMIFLGCLR register or by an SYSRSn reset: 0,No Vector Fetch Error condition (on the other CPU) pending 1,Vector Fetch error condition (on the other CPU) generated Reset type: SYSRSn
5	CPU2HWBISTERR	R	0h	HW BIST Error NMI Flag: This bit indicates if the time out error or a signature mismatch error condition during hardware BIST of C28 CPU2 occurred. This bit can only be cleared by the user writing to the corresponding clear bit in the NMIFLGCLR register or by an SYSRSn reset: 0,No C28 HWBIST error condition pending 1,C28 BIST error condition generated Reset type: SYSRSn
4	CPU1HWBISTERR	R	0h	HW BIST Error NMI Flag: This bit indicates if the time out error or a signature mismatch error condition during hardware BIST of C28 CPU1 occurred. This bit can only be cleared by the user writing to the corresponding clear bit in the NMIFLGCLR register or by an SYSRSn reset: 0,No C28 HWBIST error condition pending 1,C28 BIST error condition generated Reset type: SYSRSn
3	FLUNCERR	R	0h	Flash Uncorrectable Error NMI Flag: This bit indicates if an uncorrectable error occurred on a C28 Flash access and that condition is latched. This bit can only be cleared by the user writing to the corresponding clear bit in the NMIFLGCLR register or by an SYSRSn reset: 0,No C28 Flash uncorrectable error condition pending 1,C28 Flash uncorrectable error condition generated Reset type: SYSRSn
2	RAMUNCERR	R	0h	RAM Uncorrectable Error NMI Flag: This bit indicates if an uncorrectable error occurred on a RAM access (by any master) and that condition is latched. This bit can only be cleared by the user writing to the corresponding clear bit in the NMIFLGCLR register or by an SYSRSn reset: 0,No RAM uncorrectable error condition pending 1,RAM uncorrectable error condition generated Note: This nmi is a combination of uncorrectable error in RAMs and ROMs. ROM parity error would also set this flag. Reset type: SYSRSn
1	CLOCKFAIL	R	0h	Clock Fail Interrupt Flag: These bits indicates if the CLOCKFAIL condition is latched. These bits can only be cleared by the user writing to the respective bit in the NMIFLGCLR register or by an SYSRSn reset: 0,No CLOCKFAIL Condition Pending 1,CLOCKFAIL Condition Generated Reset type: SYSRSn

Table 3-315. NMIFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	NMIINT	R	0h	NMI Interrupt Flag: This bit indicates if an NMI interrupt was generated. This bit can only be cleared by the user writing to the respective bit in the NMIFLGCLR register or by an SYSRSn reset: 0 No NMI Interrupt Generated 1 NMI Interrupt Generated No further NMI interrupts pulses are generated until this flag is cleared by the user. Reset type: SYSRSn

3.16.13.3 NMIFLGCLR Register (Offset = 2h) [Reset = 0h]

NMIFLGCLR is shown in Figure 3-296 and described in Table 3-316.

Return to the Summary Table.

NMI Flag Clear Register

Figure 3-296. NMIFLGCLR Register

15	14	13	12	11	10	9	8
MCAN_ERR	CRC_FAIL	ECATNMIn	CMNMIWDRSn	RESERVED	CPU2NMIWDRSn	CPU2WDRSn	CLBNMI
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
ERADNMI	PIEVECTERR	CPU2HWBISTERR	CPU1HWBISTERR	FLUNCERR	RAMUNCERR	CLOCKFAIL	NMIINT
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-316. NMIFLGCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15	MCAN_ERR	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn
14	CRC_FAIL	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn
13	ECATNMIn	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. [3] This bit is reserved for CPU2.NMIFLG register Reset type: SYSRSn
12	CMNMIWDRSn	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. [3] This bit is reserved for CPU2.NMIFLG register Reset type: SYSRSn
11	RESERVED	R-0/W1S	0h	Reserved

Table 3-316. NMIFLGCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	CPU2NMIWDRSn	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn
9	CPU2WDRSn	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. [3] CPU2WDRSn and CPU2NMIWDRSn bits are reserved for CPU2.NMIFLGCLR registers Reset type: SYSRSn
8	CLBNMI	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn
7	ERADNMI	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn
6	PIEVECTERR	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn
5	CPU2HWBISTERR	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn

Table 3-316. NMIFLGCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	CPU1HWBISTERR	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn
3	FLUNCERR	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn
2	RAMUNCERR	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn
1	CLOCKFAIL	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn
0	NMIINT	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn

3.16.13.4 NMIFLGFRC Register (Offset = 3h) [Reset = 0h]

NMIFLGFRC is shown in Figure 3-297 and described in Table 3-317.

Return to the Summary Table.

NMI Flag Force Register

Figure 3-297. NMIFLGFRC Register

15	14	13	12	11	10	9	8
MCAN_ERR	CRC_FAIL	ECATNMIn	CMNMIWDRSn	RESERVED	CPU2NMIWDRSn	CPU2WDRSn	CLBNMI
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
ERADNMI	PIEVECTERR	CPU2HWBISTERR	CPU1HWBISTERR	FLUNCERR	RAMUNCERR	CLOCKFAIL	RESERVED
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0-0h

Table 3-317. NMIFLGFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15	MCAN_ERR	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. [1] This bit is reserved for CPU2.NMIFLG register Reset type: SYSRSn
14	CRC_FAIL	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: SYSRSn
13	ECATNMIn	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. [1] This bit is reserved for CPU2.NMIFLG register Reset type: SYSRSn
12	CMNMIWDRSn	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. [1] This bit is reserved for CPU2.NMIFLG register Reset type: SYSRSn
11	RESERVED	R-0/W1S	0h	Reserved
10	CPU2NMIWDRSn	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Note: [1] CPU2WDRSn and CPU2NMIWDRSn bits are reserved for CPU2.NMIFLGCLR registers Reset type: SYSRSn
9	CPU2WDRSn	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Note: [1] CPU2WDRSn and CPU2NMIWDRSn bits are reserved for CPU2.NMIFLGCLR registers Reset type: SYSRSn

Table 3-317. NMIFLGFRC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	CLBNMI	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: SYSRSn
7	ERADNMI	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: SYSRSn
6	PIEVECTERR	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: SYSRSn
5	CPU2HWBISTERR	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: SYSRSn
4	CPU1HWBISTERR	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: SYSRSn
3	FLUNCERR	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: SYSRSn
2	RAMUNCERR	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: SYSRSn
1	CLOCKFAIL	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: SYSRSn
0	RESERVED	R-0	0h	Reserved

3.16.13.5 NMIWDCNT Register (Offset = 4h) [Reset = 0h]

NMIWDCNT is shown in Figure 3-298 and described in Table 3-318.

Return to the Summary Table.

NMI Watchdog Counter Register

Figure 3-298. NMIWDCNT Register

15	14	13	12	11	10	9	8
NMIWDCNT							
R-0h							
7	6	5	4	3	2	1	0
NMIWDCNT							
R-0h							

Table 3-318. NMIWDCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	NMIWDCNT	R	0h	NMI Watchdog Counter: This 16-bit incremental counter will start incrementing whenever any one of the enabled FAIL flags are set. If the counter reaches the period value, an NMIRSn signal is fired which will then resets the system. The counter will reset to zero when it reaches the period value and will then restart counting if any of the enabled FAIL flags are set. If no enabled FAIL flag is set, then the counter will reset to zero and remain at zero until an enabled FAIL flag is set. Normally, the software would respond to the NMI interrupt generated and clear the offending FLAG(s) before the NMI watchdog triggers a reset. In some situations, the software may decide to allow the watchdog to reset the device anyway. The counter is clocked at the SYSCLKOUT rate. Reset type: SYSRSn

3.16.13.6 NMIWDPRD Register (Offset = 5h) [Reset = FFFFh]

NMIWDPRD is shown in Figure 3-299 and described in Table 3-319.

Return to the Summary Table.

NMI Watchdog Period Register

Figure 3-299. NMIWDPRD Register

15	14	13	12	11	10	9	8
NMIWDPRD							
R/W-FFFFh							
7	6	5	4	3	2	1	0
NMIWDPRD							
R/W-FFFFh							

Table 3-319. NMIWDPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	NMIWDPRD	R/W	FFFFh	NMI Watchdog Period: This 16-bit value contains the period value at which a reset is generated when the watchdog counter matches. At reset this value is set at the maximum. The software can decrease the period value at initialization time. Reset type: SYSRSn

3.16.13.7 NMISHDFLG Register (Offset = 6h) [Reset = 0h]

NMISHDFLG is shown in Figure 3-300 and described in Table 3-320.

Return to the Summary Table.

NMI Shadow Flag Register

Figure 3-300. NMISHDFLG Register

15	14	13	12	11	10	9	8
MCAN_ERR	CRC_FAIL	ECATNMIn	CMNMIWDRSn	RESERVED	CPU2NMIWDRSn	CPU2WDRSn	CLBNMI
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
ERADNMI	PIEVECTERR	CPU2HWBISTERR	CPU1HWBISTERR	FLUNCERR	RAMUNCERR	CLOCKFAIL	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0-0h

Table 3-320. NMISHDFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15	MCAN_ERR	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is reset only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. [2] This bit is reserved for CPU2.NMIFLG register Reset type: SYSRSn
14	CRC_FAIL	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is reset only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: SYSRSn
13	ECATNMIn	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is reset only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. [2] This bit is reserved for CPU2.NMIFLG register Reset type: SYSRSn
12	CMNMIWDRSn	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is reset only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. [2] This bit is reserved for CPU2.NMIFLG register Reset type: PORESETn
11	RESERVED	R	0h	Reserved

Table 3-320. NMISHDFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	CPU2NMIWDRSn	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is reset only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
9	CPU2WDRSn	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is reset only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. [2] CPU2WDRSn and CPU2NMIWDRSn bits are reserved for CPU2.NMIFLGCLR registers Reset type: PORESETn
8	CLBNMI	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is reset only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
7	ERADNMI	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is reset only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
6	PIEVECTERR	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is reset only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
5	CPU2HWBISTERR	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is reset only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn

Table 3-320. NMISHDFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	CPU1HWBISTERR	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is reset only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
3	FLUNCERR	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is reset only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
2	RAMUNCERR	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is reset only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
1	CLOCKFAIL	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is reset only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
0	RESERVED	R-0	0h	Reserved

3.16.13.8 ERRORSTS Register (Offset = 7h) [Reset = 2h]

ERRORSTS is shown in Figure 3-301 and described in Table 3-321.

Return to the Summary Table.

Error pin status

Figure 3-301. ERRORSTS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						PINSTS	ERROR
R-0-0h						R-1h	R-0h

Table 3-321. ERRORSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R-0	0h	Reserved
1	PINSTS	R	1h	0, Error Pin is 0 1, Error Pin is 1 Note: ERRORSTS register can be read by CPU2 but cannot be cleared by CPU2. Reset type: PORESETn
0	ERROR	R	0h	0,None of the error sources were triggered. 1, One or more of the error sources triggered, or ERRORSTS.ERROR was set by a write of 1 to ERRORSTSFRC.ERROR bit. Once set, the ERROR flag can be cleared by writing 1 to ERRORSTSCLR.ERROR bit. Following are the events/triggers which can set this bit: 1. If any of flags in NMISHDFLG register is set on CPU1/CPU2 2. Watchdog reset 3. Error on a Pie vector fetch 4. Efuse error 5. If any of flags in NMISHDFLG register is set on CM On a read of this bit, the pin Error pin state will be returned. Note: ERRORSTS register can be read by CPU2 but cannot be cleared by CPU2. Reset type: PORESETn

3.16.13.9 ERRORSTSCLR Register (Offset = 8h) [Reset = 0h]

ERRORSTSCLR is shown in Figure 3-302 and described in Table 3-322.

Return to the Summary Table.

ERRORSTS clear register

Figure 3-302. ERRORSTSCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED							ERROR
R-0-0h							R-0/W1S-0h

Table 3-322. ERRORSTSCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R-0	0h	Reserved
0	ERROR	R-0/W1S	0h	0,No effect 1, ERRORSTS.ERROR is cleared to 0 Note:This register is available only on CPU1 Reset type: PORESETn

3.16.13.10 ERRORSTSFRC Register (Offset = 9h) [Reset = 0h]

ERRORSTSFRC is shown in Figure 3-303 and described in Table 3-323.

Return to the Summary Table.

ERRORSTS force register

Figure 3-303. ERRORSTSFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED							ERROR
R-0-0h							R-0/W1S-0h

Table 3-323. ERRORSTSFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R-0	0h	Reserved
0	ERROR	R-0/W1S	0h	0,No effect 1, ERRORSTS.ERROR is set to 1 Note:This register is available only on CPU1 Reset type: PORESETn

3.16.13.11 ERRORCTL Register (Offset = Ah) [Reset = 0h]

ERRORCTL is shown in Figure 3-304 and described in Table 3-324.

Return to the Summary Table.

Error pin control register

Figure 3-304. ERRORCTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED							ERRORPOLSEL
R-0-0h							R/W-0h

Table 3-324. ERRORCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R-0	0h	Reserved
0	ERRORPOLSEL	R/W	0h	0, If ERRORSTS.ERROR is 1, Error pin will be driven with a value of 0, else 1. 1, If ERRORSTS.ERROR is 1, Error pin will be driven with a value of 1, else 0. Note:This register is available only on CPU1 Reset type: PORESETn

3.16.13.12 ERRORLOCK Register (Offset = Bh) [Reset = 0h]

ERRORLOCK is shown in Figure 3-305 and described in Table 3-325.

Return to the Summary Table.

Lock register to Error pin registers.

Figure 3-305. ERRORLOCK Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED							ERRORCTL
R-0-0h							R/WSonce-0h

Table 3-325. ERRORLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R-0	0h	Reserved
0	ERRORCTL	R/WSonce	0h	0, Writes to ERRORCTL register allowed. 1, Writes to ERRORCTL register is blocked. Writes of 0 to this bit has no effect. Write of 1 will set this bit, cleared only on a SYSRSn. Note:This register is available only on CPU1 Reset type: SYSRSn

3.16.14 PIE_CTRL_REGS Registers

Table 3-326 lists the memory-mapped registers for the PIE_CTRL_REGS registers. All register offset addresses not listed in Table 3-326 should be considered as reserved locations and the register contents should not be modified.

Table 3-326. PIE_CTRL_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	PIECTRL	ePIE Control Register		Go
1h	PIEACK	Interrupt Acknowledge Register		Go
2h	PIEIER1	Interrupt Group 1 Enable Register		Go
3h	PIEIFR1	Interrupt Group 1 Flag Register		Go
4h	PIEIER2	Interrupt Group 2 Enable Register		Go
5h	PIEIFR2	Interrupt Group 2 Flag Register		Go
6h	PIEIER3	Interrupt Group 3 Enable Register		Go
7h	PIEIFR3	Interrupt Group 3 Flag Register		Go
8h	PIEIER4	Interrupt Group 4 Enable Register		Go
9h	PIEIFR4	Interrupt Group 4 Flag Register		Go
Ah	PIEIER5	Interrupt Group 5 Enable Register		Go
Bh	PIEIFR5	Interrupt Group 5 Flag Register		Go
Ch	PIEIER6	Interrupt Group 6 Enable Register		Go
Dh	PIEIFR6	Interrupt Group 6 Flag Register		Go
Eh	PIEIER7	Interrupt Group 7 Enable Register		Go
Fh	PIEIFR7	Interrupt Group 7 Flag Register		Go
10h	PIEIER8	Interrupt Group 8 Enable Register		Go
11h	PIEIFR8	Interrupt Group 8 Flag Register		Go
12h	PIEIER9	Interrupt Group 9 Enable Register		Go
13h	PIEIFR9	Interrupt Group 9 Flag Register		Go
14h	PIEIER10	Interrupt Group 10 Enable Register		Go
15h	PIEIFR10	Interrupt Group 10 Flag Register		Go
16h	PIEIER11	Interrupt Group 11 Enable Register		Go
17h	PIEIFR11	Interrupt Group 11 Flag Register		Go
18h	PIEIER12	Interrupt Group 12 Enable Register		Go
19h	PIEIFR12	Interrupt Group 12 Flag Register		Go

Complex bit access types are encoded to fit into small table cells. Table 3-327 shows the codes that are used for access types in this section.

Table 3-327. PIE_CTRL_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		

Table 3-327. PIE_CTRL_REGS Access Type Codes (continued)

Access Type	Code	Description
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.14.1 PIECTRL Register (Offset = 0h) [Reset = 0h]

PIECTRL is shown in Figure 3-306 and described in Table 3-328.

Return to the Summary Table.

ePIE Control Register

Figure 3-306. PIECTRL Register

15	14	13	12	11	10	9	8
PIEVECT							
R-0h							
7	6	5	4	3	2	1	0
PIEVECT							ENPIE
R-0h							R/W-0h

Table 3-328. PIECTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	PIEVECT	R	0h	These bits indicate the vector address of the vector fetched from the ePIE vector table. The least significant bit of the address is ignored and only bits 1 to 15 of the address are shown. The vector value can be read by the user to determine which interrupt generated the vector fetch. Note: When a NMI is serviced, the PIEVECT bit-field does not reflect the vector as it does for other interrupts. Reset type: SYSRSn
0	ENPIE	R/W	0h	Enable vector fetching from ePIE block. This bit must be set to 1 for peripheral interrupts to work. All ePIE registers (PIEACK, PIEIFR, PIEIER) can be accessed even when the ePIE block is disabled. Reset type: SYSRSn

3.16.14.2 PIEACK Register (Offset = 1h) [Reset = 0h]

PIEACK is shown in Figure 3-307 and described in Table 3-329.

Return to the Summary Table.

Acknowledge Register
When an interrupt propagates from the ePIE to a CPU interrupt line, the interrupt group's PIEACK bit is set. This prevents other interrupts in that group from propagating to the CPU while the first interrupt is handled. Writing a 1 to a PIEACK bit clears it and allows another interrupt from the corresponding group to propagate. ISRs for PIE interrupts should clear the group's PIEACK bit before returning from the interrupt.
Writes of 0 are ignored.

Figure 3-307. PIEACK Register

15	14	13	12	11	10	9	8
RESERVED				ACK12	ACK11	ACK10	ACK9
R-0-0h				R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
ACK8	ACK7	ACK6	ACK5	ACK4	ACK3	ACK2	ACK1
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

Table 3-329. PIEACK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R-0	0h	Reserved
11	ACK12	R/W1S	0h	Acknowledge PIE Interrupt Group 12 Reset type: SYSRSn
10	ACK11	R/W1S	0h	Acknowledge PIE Interrupt Group 11 Reset type: SYSRSn
9	ACK10	R/W1S	0h	Acknowledge PIE Interrupt Group 10 Reset type: SYSRSn
8	ACK9	R/W1S	0h	Acknowledge PIE Interrupt Group 9 Reset type: SYSRSn
7	ACK8	R/W1S	0h	Acknowledge PIE Interrupt Group 8 Reset type: SYSRSn
6	ACK7	R/W1S	0h	Acknowledge PIE Interrupt Group 7 Reset type: SYSRSn
5	ACK6	R/W1S	0h	Acknowledge PIE Interrupt Group 6 Reset type: SYSRSn
4	ACK5	R/W1S	0h	Acknowledge PIE Interrupt Group 5 Reset type: SYSRSn
3	ACK4	R/W1S	0h	Acknowledge PIE Interrupt Group 4 Reset type: SYSRSn
2	ACK3	R/W1S	0h	Acknowledge PIE Interrupt Group 3 Reset type: SYSRSn
1	ACK2	R/W1S	0h	Acknowledge PIE Interrupt Group 2 Reset type: SYSRSn
0	ACK1	R/W1S	0h	Acknowledge PIE Interrupt Group 1 Reset type: SYSRSn

3.16.14.3 PIEIER1 Register (Offset = 2h) [Reset = 0h]

PIEIER1 is shown in Figure 3-308 and described in Table 3-330.

Return to the Summary Table.

Interrupt Group 1 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-308. PIEIER1 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-330. PIEIER1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 1.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 1.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 1.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 1.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 1.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 1.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 1.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 1.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 1.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 1.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 1.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 1.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 1.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 1.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 1.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 1.1 Reset type: SYSRSn

3.16.14.4 PIEIFR1 Register (Offset = 3h) [Reset = 0h]

PIEIFR1 is shown in Figure 3-309 and described in Table 3-331.

Return to the Summary Table.

Interrupt Group 1 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-309. PIEIFR1 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-331. PIEIFR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 1.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 1.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 1.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 1.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 1.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 1.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 1.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 1.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 1.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 1.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 1.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 1.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 1.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 1.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 1.2 Reset type: SYSRSn

Table 3-331. PIEIFR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 1.1 Reset type: SYSRSn

3.16.14.5 PIEIER2 Register (Offset = 4h) [Reset = 0h]

PIEIER2 is shown in Figure 3-310 and described in Table 3-332.

Return to the Summary Table.

Interrupt Group 2 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-310. PIEIER2 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-332. PIEIER2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 2.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 2.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 2.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 2.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 2.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 2.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 2.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 2.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 2.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 2.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 2.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 2.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 2.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 2.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 2.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 2.1 Reset type: SYSRSn

3.16.14.6 PIEIFR2 Register (Offset = 5h) [Reset = 0h]

PIEIFR2 is shown in Figure 3-311 and described in Table 3-333.

Return to the Summary Table.

Interrupt Group 2 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-311. PIEIFR2 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-333. PIEIFR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 2.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 2.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 2.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 2.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 2.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 2.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 2.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 2.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 2.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 2.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 2.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 2.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 2.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 2.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 2.2 Reset type: SYSRSn

Table 3-333. PIEIFR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 2.1 Reset type: SYSRSn

3.16.14.7 PIEIER3 Register (Offset = 6h) [Reset = 0h]

PIEIER3 is shown in Figure 3-312 and described in Table 3-334.

Return to the Summary Table.

Interrupt Group 3 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-312. PIEIER3 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-334. PIEIER3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 3.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 3.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 3.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 3.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 3.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 3.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 3.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 3.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 3.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 3.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 3.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 3.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 3.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 3.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 3.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 3.1 Reset type: SYSRSn

3.16.14.8 PIEIFR3 Register (Offset = 7h) [Reset = 0h]

PIEIFR3 is shown in Figure 3-313 and described in Table 3-335.

Return to the Summary Table.

Interrupt Group 3 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-313. PIEIFR3 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-335. PIEIFR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 3.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 3.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 3.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 3.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 3.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 3.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 3.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 3.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 3.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 3.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 3.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 3.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 3.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 3.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 3.2 Reset type: SYSRSn

Table 3-335. PIEIFR3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 3.1 Reset type: SYSRSn

3.16.14.9 PIEIER4 Register (Offset = 8h) [Reset = 0h]

PIEIER4 is shown in Figure 3-314 and described in Table 3-336.

Return to the Summary Table.

Interrupt Group 4 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-314. PIEIER4 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-336. PIEIER4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 4.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 4.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 4.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 4.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 4.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 4.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 4.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 4.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 4.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 4.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 4.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 4.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 4.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 4.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 4.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 4.1 Reset type: SYSRSn

3.16.14.10 PIEIFR4 Register (Offset = 9h) [Reset = 0h]

PIEIFR4 is shown in Figure 3-315 and described in Table 3-337.

Return to the Summary Table.

Interrupt Group 4 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-315. PIEIFR4 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-337. PIEIFR4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 4.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 4.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 4.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 4.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 4.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 4.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 4.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 4.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 4.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 4.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 4.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 4.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 4.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 4.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 4.2 Reset type: SYSRSn

Table 3-337. PIEIFR4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 4.1 Reset type: SYSRSn

3.16.14.11 PIEIER5 Register (Offset = Ah) [Reset = 0h]

PIEIER5 is shown in Figure 3-316 and described in Table 3-338.

Return to the Summary Table.

Interrupt Group 5 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-316. PIEIER5 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-338. PIEIER5 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 5.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 5.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 5.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 5.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 5.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 5.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 5.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 5.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 5.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 5.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 5.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 5.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 5.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 5.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 5.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 5.1 Reset type: SYSRSn

3.16.14.12 PIEIFR5 Register (Offset = Bh) [Reset = 0h]

PIEIFR5 is shown in Figure 3-317 and described in Table 3-339.

Return to the Summary Table.

Interrupt Group 5 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-317. PIEIFR5 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-339. PIEIFR5 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 5.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 5.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 5.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 5.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 5.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 5.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 5.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 5.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 5.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 5.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 5.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 5.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 5.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 5.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 5.2 Reset type: SYSRSn

Table 3-339. PIEIFR5 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 5.1 Reset type: SYSRSn

3.16.14.13 PIEIER6 Register (Offset = Ch) [Reset = 0h]

PIEIER6 is shown in Figure 3-318 and described in Table 3-340.

Return to the Summary Table.

Interrupt Group 6 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-318. PIEIER6 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-340. PIEIER6 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 6.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 6.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 6.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 6.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 6.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 6.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 6.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 6.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 6.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 6.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 6.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 6.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 6.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 6.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 6.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 6.1 Reset type: SYSRSn

3.16.14.14 PIEIFR6 Register (Offset = Dh) [Reset = 0h]

PIEIFR6 is shown in Figure 3-319 and described in Table 3-341.

Return to the Summary Table.

Interrupt Group 6 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-319. PIEIFR6 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-341. PIEIFR6 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 6.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 6.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 6.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 6.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 6.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 6.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 6.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 6.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 6.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 6.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 6.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 6.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 6.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 6.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 6.2 Reset type: SYSRSn

Table 3-341. PIEIFR6 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 6.1 Reset type: SYSRSn

3.16.14.15 PIEIER7 Register (Offset = Eh) [Reset = 0h]

PIEIER7 is shown in Figure 3-320 and described in Table 3-342.

Return to the Summary Table.

Interrupt Group 7 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-320. PIEIER7 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-342. PIEIER7 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 7.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 7.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 7.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 7.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 7.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 7.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 7.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 7.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 7.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 7.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 7.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 7.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 7.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 7.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 7.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 7.1 Reset type: SYSRSn

3.16.14.16 PIEIFR7 Register (Offset = Fh) [Reset = 0h]

PIEIFR7 is shown in Figure 3-321 and described in Table 3-343.

Return to the Summary Table.

Interrupt Group 7 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-321. PIEIFR7 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-343. PIEIFR7 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 7.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 7.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 7.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 7.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 7.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 7.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 7.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 7.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 7.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 7.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 7.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 7.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 7.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 7.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 7.2 Reset type: SYSRSn

Table 3-343. PIEIFR7 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 7.1 Reset type: SYSRSn

3.16.14.17 PIEIER8 Register (Offset = 10h) [Reset = 0h]

PIEIER8 is shown in Figure 3-322 and described in Table 3-344.

Return to the Summary Table.

Interrupt Group 8 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-322. PIEIER8 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-344. PIEIER8 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 8.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 8.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 8.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 8.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 8.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 8.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 8.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 8.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 8.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 8.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 8.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 8.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 8.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 8.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 8.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 8.1 Reset type: SYSRSn

3.16.14.18 PIEIFR8 Register (Offset = 11h) [Reset = 0h]

PIEIFR8 is shown in Figure 3-323 and described in Table 3-345.

Return to the Summary Table.

Interrupt Group 8 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-323. PIEIFR8 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-345. PIEIFR8 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 8.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 8.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 8.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 8.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 8.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 8.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 8.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 8.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 8.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 8.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 8.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 8.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 8.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 8.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 8.2 Reset type: SYSRSn

Table 3-345. PIEIFR8 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 8.1 Reset type: SYSRSn

3.16.14.19 PIEIER9 Register (Offset = 12h) [Reset = 0h]

PIEIER9 is shown in Figure 3-324 and described in Table 3-346.

Return to the Summary Table.

Interrupt Group 9 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-324. PIEIER9 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-346. PIEIER9 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 9.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 9.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 9.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 9.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 9.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 9.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 9.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 9.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 9.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 9.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 9.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 9.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 9.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 9.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 9.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 9.1 Reset type: SYSRSn

3.16.14.20 PIEIFR9 Register (Offset = 13h) [Reset = 0h]

PIEIFR9 is shown in Figure 3-325 and described in Table 3-347.

Return to the Summary Table.

Interrupt Group 9 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-325. PIEIFR9 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-347. PIEIFR9 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 9.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 9.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 9.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 9.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 9.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 9.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 9.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 9.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 9.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 9.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 9.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 9.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 9.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 9.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 9.2 Reset type: SYSRSn

Table 3-347. PIEIFR9 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 9.1 Reset type: SYSRSn

3.16.14.21 PIEIER10 Register (Offset = 14h) [Reset = 0h]

PIEIER10 is shown in Figure 3-326 and described in Table 3-348.

Return to the Summary Table.

Interrupt Group 10 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-326. PIEIER10 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-348. PIEIER10 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 10.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 10.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 10.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 10.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 10.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 10.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 10.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 10.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 10.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 10.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 10.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 10.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 10.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 10.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 10.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 10.1 Reset type: SYSRSn

3.16.14.22 PIEIFR10 Register (Offset = 15h) [Reset = 0h]

PIEIFR10 is shown in Figure 3-327 and described in Table 3-349.

Return to the Summary Table.

Interrupt Group 10 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-327. PIEIFR10 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-349. PIEIFR10 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 10.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 10.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 10.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 10.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 10.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 10.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 10.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 10.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 10.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 10.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 10.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 10.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 10.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 10.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 10.2 Reset type: SYSRSn

Table 3-349. PIEIFR10 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 10.1 Reset type: SYSRSn

3.16.14.23 PIEIER11 Register (Offset = 16h) [Reset = 0h]

PIEIER11 is shown in Figure 3-328 and described in Table 3-350.

Return to the Summary Table.

Interrupt Group 11 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-328. PIEIER11 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-350. PIEIER11 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 11.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 11.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 11.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 11.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 11.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 11.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 11.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 11.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 11.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 11.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 11.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 11.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 11.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 11.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 11.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 11.1 Reset type: SYSRSn

3.16.14.24 PIEIFR11 Register (Offset = 17h) [Reset = 0h]

PIEIFR11 is shown in Figure 3-329 and described in Table 3-351.

Return to the Summary Table.

Interrupt Group 11 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-329. PIEIFR11 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-351. PIEIFR11 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 11.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 11.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 11.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 11.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 11.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 11.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 11.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 11.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 11.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 11.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 11.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 11.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 11.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 11.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 11.2 Reset type: SYSRSn

Table 3-351. PIEIFR11 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 11.1 Reset type: SYSRSn

3.16.14.25 PIEIER12 Register (Offset = 18h) [Reset = 0h]

PIEIER12 is shown in Figure 3-330 and described in Table 3-352.

Return to the Summary Table.

Interrupt Group 12 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-330. PIEIER12 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-352. PIEIER12 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 12.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 12.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 12.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 12.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 12.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 12.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 12.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 12.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 12.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 12.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 12.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 12.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 12.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 12.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 12.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 12.1 Reset type: SYSRSn

3.16.14.26 PIEIFR12 Register (Offset = 19h) [Reset = 0h]

PIEIFR12 is shown in Figure 3-331 and described in Table 3-353.

Return to the Summary Table.

Interrupt Group 12 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-331. PIEIFR12 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-353. PIEIFR12 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 12.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 12.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 12.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 12.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 12.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 12.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 12.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 12.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 12.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 12.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 12.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 12.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 12.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 12.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 12.2 Reset type: SYSRSn

Table 3-353. PIEIFR12 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 12.1 Reset type: SYSRSn

3.16.15 ROM_PREFETCH_REGS Registers

Table 3-354 lists the memory-mapped registers for the ROM_PREFETCH_REGS registers. All register offset addresses not listed in Table 3-354 should be considered as reserved locations and the register contents should not be modified.

Table 3-354. ROM_PREFETCH_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	ROMPREFETCH	ROM Prefetch Configuration Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-355 shows the codes that are used for access types in this section.

Table 3-355. ROM_PREFETCH_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.15.1 ROMPREFETCH Register (Offset = 0h) [Reset = 1h]

ROMPREFETCH is shown in Figure 3-332 and described in Table 3-356.

Return to the Summary Table.

ROM Prefetch Configuration Register

Figure 3-332. ROMPREFETCH Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							PFENABLE
R-0h							R/W-1h

Table 3-356. ROMPREFETCH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	PFENABLE	R/W	1h	0: Prefetch is disabled for secure ROM and boot ROM. 1: Prefetch is enabled for secure ROM and boot ROM. Reset type: SYSRSn

3.16.16 ROM_WAIT_STATE_REGS Registers

Table 3-357 lists the memory-mapped registers for the ROM_WAIT_STATE_REGS registers. All register offset addresses not listed in Table 3-357 should be considered as reserved locations and the register contents should not be modified.

Table 3-357. ROM_WAIT_STATE_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	ROMWAITSTATE	ROM Wait State Configuration Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-358 shows the codes that are used for access types in this section.

Table 3-358. ROM_WAIT_STATE_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.16.1 ROMWAITSTATE Register (Offset = 0h) [Reset = 0h]

ROMWAITSTATE is shown in Figure 3-333 and described in Table 3-359.

Return to the Summary Table.

ROM Wait State Configuration Register

Figure 3-333. ROMWAITSTATE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							WSDISABLE
R-0h							R/W-0h

Table 3-359. ROMWAITSTATE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	WSDISABLE	R/W	0h	0: ROM Wait State is enabled. CPU accesses to secure ROM and boot ROM are are 1-wait. 1: ROM Wait State is disabled. CPU accesses to secure ROM and boot ROM are 0-wait. Reset type: SYSRSn

3.16.17 SYNC_SOC_REGS Registers

Table 3-360 lists the memory-mapped registers for the SYNC_SOC_REGS registers. All register offset addresses not listed in Table 3-360 should be considered as reserved locations and the register contents should not be modified.

Table 3-360. SYNC_SOC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	SYNCSELECT	Sync Input and Output Select Register	EALLOW	Go
2h	ADCSOCOUTSELECT	External ADC (Off Chip) SOC Select Register	EALLOW	Go
4h	SYNCSOCLOCK	SYNCSEL and EXTADCSOC Select Lock register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-361 shows the codes that are used for access types in this section.

Table 3-361. SYNC_SOC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.17.1 SYNCSELECT Register (Offset = 0h) [Reset = 0h]

SYNCSELECT is shown in Figure 3-334 and described in Table 3-362.

Return to the Summary Table.

Sync Input and Output Select Register

Figure 3-334. SYNCSELECT Register

31	30	29	28	27	26	25	24
RESERVED				SYNCOUT			
R/W-0h				R/W-0h			
23	22	21	20	19	18	17	16
RESERVED				RESERVED			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED	RESERVED			RESERVED			RESERVED
R-0-0h	R/W-0h			R/W-0h			R/W-0h
7	6	5	4	3	2	1	0
RESERVED		RESERVED			RESERVED		
R/W-0h		R/W-0h			R/W-0h		

Table 3-362. SYNCSELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R/W	0h	Reserved

Table 3-362. SYNCSELECT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
28-24	SYNCOUT	R/W	0h	Select Syncout Source: 00000: EPWM1SYNCOUT selected to drive the SYNCOUT pin. 00001: EPWM2SYNCOUT selected to drive the SYNCOUT pin. 00010: EPWM3SYNCOUT selected to drive the SYNCOUT pin. 00011: EPWM4SYNCOUT selected to drive the SYNCOUT pin. 00100: EPWM5SYNCOUT selected to drive the SYNCOUT pin. 00101: EPWM6SYNCOUT selected to drive the SYNCOUT pin. 00110: EPWM7SYNCOUT selected to drive the SYNCOUT pin. 00111: EPWM8SYNCOUT selected to drive the SYNCOUT pin. 01000: EPWM9SYNCOUT selected to drive the SYNCOUT pin. 01001: EPWM10SYNCOUT selected to drive the SYNCOUT pin. 01010: EPWM11SYNCOUT selected to drive the SYNCOUT pin. 01011: EPWM12SYNCOUT selected to drive the SYNCOUT pin. 01100: EPWM13SYNCOUT selected to drive the SYNCOUT pin. 01101: EPWM14SYNCOUT selected to drive the SYNCOUT pin. 01110: EPWM15SYNCOUT selected to drive the SYNCOUT pin. 01111: EPWM16SYNCOUT selected to drive the SYNCOUT pin. 10000: Reserved 10001: Reserved 10010: Reserved 10011: Reserved 10100: Reserved 10101: Reserved 10110: Reserved 10111: Reserved 11000: ECAP1SYNCOUT selected to drive the SYNCOUT pin. 11001: ECAP2SYNCOUT selected to drive the SYNCOUT pin. 11010: ECAP3SYNCOUT selected to drive the SYNCOUT pin. 11011: ECAP4SYNCOUT selected to drive the SYNCOUT pin. 11100: ECAP5SYNCOUT selected to drive the SYNCOUT pin. 11101: ECAP6SYNCOUT selected to drive the SYNCOUT pin. 11110: ECAP7SYNCOUT selected to drive the SYNCOUT pin. 11111: Reserved Notes: [1] Reserved position defaults to 00 selection Reset type: CPU1.SYSRSn
23-20	RESERVED	R/W	0h	Reserved
19-16	RESERVED	R/W	0h	Reserved
15	RESERVED	R-0	0h	Reserved
14-12	RESERVED	R/W	0h	Reserved
11-9	RESERVED	R/W	0h	Reserved
8-6	RESERVED	R/W	0h	Reserved
5-3	RESERVED	R/W	0h	Reserved
2-0	RESERVED	R/W	0h	Reserved

3.16.17.2 ADCSOCOUTSELECT Register (Offset = 2h) [Reset = 0h]

ADCSOCOUTSELECT is shown in Figure 3-335 and described in Table 3-363.

Return to the Summary Table.

External ADC (Off Chip) SOC Select Register

Figure 3-335. ADCSOCOUTSELECT Register

31	30	29	28	27	26	25	24
PWM16SOCBEN	PWM15SOCBEN	PWM14SOCBEN	PWM13SOCBEN	PWM12SOCBEN	PWM11SOCBEN	PWM10SOCBEN	PWM9SOCBEN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
PWM8SOCBEN	PWM7SOCBEN	PWM6SOCBEN	PWM5SOCBEN	PWM4SOCBEN	PWM3SOCBEN	PWM2SOCBEN	PWM1SOCBEN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
PWM16SOCAEN	PWM15SOCAEN	PWM14SOCAEN	PWM13SOCAEN	PWM12SOCAEN	PWM11SOCAEN	PWM10SOCAEN	PWM9SOCAEN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
PWM8SOCAEN	PWM7SOCAEN	PWM6SOCAEN	PWM5SOCAEN	PWM4SOCAEN	PWM3SOCAEN	PWM2SOCAEN	PWM1SOCAEN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-363. ADCSOCOUTSELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	PWM16SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
30	PWM15SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
29	PWM14SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
28	PWM13SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
27	PWM12SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
26	PWM11SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
25	PWM10SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn

Table 3-363. ADCSOCOUTSELECT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24	PWM9SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
23	PWM8SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
22	PWM7SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
21	PWM6SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
20	PWM5SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
19	PWM4SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
18	PWM3SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
17	PWM2SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
16	PWM1SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
15	PWM16SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
14	PWM15SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
13	PWM14SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
12	PWM13SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
11	PWM12SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn

Table 3-363. ADCSOCOUTSELECT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	PWM11SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
9	PWM10SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
8	PWM9SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
7	PWM8SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
6	PWM7SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
5	PWM6SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
4	PWM5SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
3	PWM4SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
2	PWM3SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
1	PWM2SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
0	PWM1SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn

3.16.17.3 SYNCSOCLOCK Register (Offset = 4h) [Reset = 0h]

SYNCSOCLOCK is shown in Figure 3-336 and described in Table 3-364.

Return to the Summary Table.

SYNCSEL and EXTADCSOC Select Lock register

Figure 3-336. SYNCSOCLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						ADCSOCOUTSELECT	SYNCSELECT
R-0-0h						R/WSonce-0h	R/WSonce-0h

Table 3-364. SYNCSOCLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-2	RESERVED	R-0	0h	Reserved
1	ADCSOCOUTSELECT	R/WSonce	0h	ADCSOCOUTSELECT Register Lock bit: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any bit in this register, once set can only be creaed through a CPU1.SYSRSn. Write of 0 to any bit of this regtister has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: CPU1.SYSRSn
0	SYNCSELECT	R/WSonce	0h	SYNCSELECT Register Lock bit: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any bit in this register, once set can only be creaed through a CPU1.SYSRSn. Write of 0 to any bit of this regtister has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: CPU1.SYSRSn

3.16.18 SYS_STATUS_REGS Registers

Table 3-365 lists the memory-mapped registers for the SYS_STATUS_REGS registers. All register offset addresses not listed in Table 3-365 should be considered as reserved locations and the register contents should not be modified.

Table 3-365. SYS_STATUS_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CM_STATUS_INT_FLG	Status of interrupts due to multiple sources of Cortex-M4 reset.		Go
2h	CM_STATUS_INT_CLR	CM_STATUS_INT_FLG clear register		Go
4h	CM_STATUS_INT_SET	CM_STATUS_INT_FLG set register	EALLOW	Go
6h	CM_STATUS_MASK	CM_STATUS_MASK register	EALLOW	Go
10h	SYS_ERR_INT_FLG	Status of interrupts due to multiple different errors in the system.		Go
12h	SYS_ERR_INT_CLR	SYS_ERR_INT_FLG clear register		Go
14h	SYS_ERR_INT_SET	SYS_ERR_INT_FLG set register	EALLOW	Go
16h	SYS_ERR_MASK	SYS_ERR_MASK register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-366 shows the codes that are used for access types in this section.

Table 3-366. SYS_STATUS_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.18.1 CM_STATUS_INT_FLG Register (Offset = 0h) [Reset = 0h]

CM_STATUS_INT_FLG is shown in Figure 3-337 and described in Table 3-367.

Return to the Summary Table.

Status of interrupts due to multiple sources of Cortex-M4 reset.
Note: This register is present only on CPU1.

Figure 3-337. CM_STATUS_INT_FLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CMVECTRESET	CMSYSRESETREQ	CMNMIWDRST	GINT
R-0h				R-0h	R-0h	R-0h	R-0h

Table 3-367. CM_STATUS_INT_FLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	CMVECTRESET	R	0h	0:CMVECTRESET has not caused a reset of CM 1:CMVECTRESET had caused a reset of CM, an interrupt will be fired if GINT flag is not set. Reset type: SYSRSn
2	CMSYSRESETREQ	R	0h	0:CMSYSRESETREQ has not caused a reset of CM 1:CMSYSRESETREQ had caused a reset of CM, an interrupt will be fired if GINT flag is not set. Reset type: SYSRSn
1	CMNMIWDRST	R	0h	0:CMNMIWDRST has not caused a reset of CM 1:CMNMIWDRST had caused a reset of CM, an interrupt will be fired if GINT flag is not set. Reset type: SYSRSn
0	GINT	R	0h	Global Interrupt flag: 0: On any of the flags of CM_STATUS_INT_FLG register being set, CM_STATUS_INT is pulsed and GINT flag would be set 1: No further interrupts would be fired until GINT flag is cleared Reset type: SYSRSn

3.16.18.2 CM_STATUS_INT_CLR Register (Offset = 2h) [Reset = 0h]

CM_STATUS_INT_CLR is shown in Figure 3-338 and described in Table 3-368.

Return to the Summary Table.

CM_STATUS_INT_FLG clear register
Note: This register is present only on CPU1.

Figure 3-338. CM_STATUS_INT_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CMVECTRESET	CMSYSRESETREQ	CMNMIWDRST	GINT
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-368. CM_STATUS_INT_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	CMVECTRESET	R-0/W1S	0h	0: No effect 1: CMVECTRESET flag of CM_STATUS_INT_FLG reister will be cleared. Reset type: SYSRSn
2	CMSYSRESETREQ	R-0/W1S	0h	0: No effect 1: CMSYSRESETREQ flag of CM_STATUS_INT_FLG reister will be cleared. Reset type: SYSRSn
1	CMNMIWDRST	R-0/W1S	0h	0: No effect 1: CMNMIWDRST flag of CM_STATUS_INT_FLG reister will be cleared. Reset type: SYSRSn
0	GINT	R-0/W1S	0h	0: No effect 1: GINT flag of CM_STATUS_INT_FLG reister will be cleared. Reset type: SYSRSn

3.16.18.3 CM_STATUS_INT_SET Register (Offset = 4h) [Reset = 0h]

CM_STATUS_INT_SET is shown in Figure 3-339 and described in Table 3-369.

Return to the Summary Table.

CM_STATUS_INT_FLG set register
Note: This register is present only on CPU1.

Figure 3-339. CM_STATUS_INT_SET Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CMVECTRESET	CMSYSRESETREQ	CMNMIWDRST	RESERVED
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0h

Table 3-369. CM_STATUS_INT_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	A value of 0xa5a5 to this field would enable write to the other bit fields of this register. Any other value written to KEY field would block the write to the other fields of this register. Note: Only a 32 bit write to this register will succeed in updating the fields of this rigister, provided the correct value written to the KEY field simultaneously Reset type: SYSRSn
15-4	RESERVED	R	0h	Reserved
3	CMVECTRESET	R-0/W1S	0h	0: No effect 1: CMVECTRESET flag of CM_STATUS_INT_FLG reister will be set. Reset type: SYSRSn
2	CMSYSRESETREQ	R-0/W1S	0h	0: No effect 1: CMSYSRESETREQ flag of CM_STATUS_INT_FLG reister will be set. Reset type: SYSRSn
1	CMNMIWDRST	R-0/W1S	0h	0: No effect 1: CMNMIWDRST flag of CM_STATUS_INT_FLG reister will be set. Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

3.16.18.4 CM_STATUS_MASK Register (Offset = 6h) [Reset = 0h]

CM_STATUS_MASK is shown in Figure 3-340 and described in Table 3-370.

Return to the Summary Table.

CM_STATUS_MASK register
Note: This register is present only on CPU1.

Figure 3-340. CM_STATUS_MASK Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CMVECTRESET	CMSYSRESETREQ	CMNMIWDRST	RESERVED
R-0h				R/W-0h	R/W-0h	R/W-0h	R-0h

Table 3-370. CM_STATUS_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	A value of 0xa5a5 to this field would enable write to the other bit fields of this register. Any other value written to KEY field would block the write to the other fields of this register. Note: Only a 32 bit write to this register will succeed in updating the fields of this rigister, provided the correct value written to the KEY field simultaneously Reset type: SYSRSn
15-4	RESERVED	R	0h	Reserved
3	CMVECTRESET	R/W	0h	0: CMVECTRESET flag of CM_STATUS_INT_FLG reister will be set on a hardware event. 1: CMVECTRESET flag of CM_STATUS_INT_FLG reister will not be set on a hardware event. Reset type: SYSRSn
2	CMSYSRESETREQ	R/W	0h	0: CMSYSRESETREQ flag of CM_STATUS_INT_FLG reister will be set on a hardware event. 1: CMSYSRESETREQ flag of CM_STATUS_INT_FLG reister will not be set on a hardware event. Reset type: SYSRSn
1	CMNMIWDRST	R/W	0h	0: CMNMIWDRST flag of CM_STATUS_INT_FLG reister will be set on a hardware event. 1: CMNMIWDRST flag of CM_STATUS_INT_FLG reister will not be set on a hardware event. Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

3.16.18.5 SYS_ERR_INT_FLG Register (Offset = 10h) [Reset = 0h]

SYS_ERR_INT_FLG is shown in Figure 3-341 and described in Table 3-371.

Return to the Summary Table.

Status of interrupts due to multiple different errors in the system.

Figure 3-341. SYS_ERR_INT_FLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						DCC2	DCC1
R-0h						R-0h	R-0h
7	6	5	4	3	2	1	0
DCC0	AUX_PLL_SLIP_NOTSUPPORTED	SYS_PLL_SLIP_NOTSUPPORTED	RAM_ACC_VIOL	FLASH_CORRECTABLE_ERR	RAM_CORRECTABLE_ERR	EMIF_ERR	GINT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 3-371. SYS_ERR_INT_FLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	DCC2	R	0h	0: DCC2 has not fired an interrupt. 1: DCC2 has fired an interrupt Reset type: SYSRSn
8	DCC1	R	0h	0: DCC1 has not fired an interrupt. 1: DCC1 has fired an interrupt Reset type: SYSRSn
7	DCC0	R	0h	0: DCC0 has not fired an interrupt. 1: DCC0 has fired an interrupt Reset type: SYSRSn
6	AUX_PLL_SLIP_NOTSUPPORTED	R	0h	RESERVED: This bit is reserved and the value read should be ignored. Reset type: SYSRSn
5	SYS_PLL_SLIP_NOTSUPPORTED	R	0h	RESERVED: This bit is reserved and the value read should be ignored. Reset type: SYSRSn
4	RAM_ACC_VIOL	R	0h	0: None of the Masters have violated the set protection rules 1: At least one of the master accesses has violated one or more of the access protection rules Reset type: SYSRSn
3	FLASH_CORRECTABLE_ERR	R	0h	0: Number of correctable errors detected has not exceeded the set threshold in FLASH. 1:Number of correctable errors detected has exceeded the set threshold in FLASH. Reset type: SYSRSn
2	RAM_CORRECTABLE_ERR	R	0h	0: Number of correctable errors detected has not exceeded the set threshold in any of the RAMs. 1:Number of correctable errors detected has exceeded the set threshold in atleast one of the RAMs. Reset type: SYSRSn

Table 3-371. SYS_ERR_INT_FLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	EMIF_ERR	R	0h	0: EMIF error has not occurred. 1: EMIF error has occurred. Reset type: SYSRSn
0	GINT	R	0h	Global Interrupt flag: 0: On any of the flags of SYS_ERR_INT_FLG register being set, SYS_ERR_INT is pulsed and GINT flag would be set 1: No further interrupts would be fired until GINT flag is cleared Reset type: SYSRSn

3.16.18.6 SYS_ERR_INT_CLR Register (Offset = 12h) [Reset = 0h]

SYS_ERR_INT_CLR is shown in Figure 3-342 and described in Table 3-372.

Return to the Summary Table.

SYS_ERR_INT_FLG clear register

Figure 3-342. SYS_ERR_INT_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						DCC2	DCC1
R-0h						R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
DCC0	AUX_PLL_SLIP_NOTSUPPORTED	SYS_PLL_SLIP_NOTSUPPORTED	RAM_ACC_VIOL	FLASH_CORRECTABLE_ERR	RAM_CORRECTABLE_ERR	EMIF_ERR	GINT
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-372. SYS_ERR_INT_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	DCC2	R-0/W1S	0h	0: No effect 1: DCC2 flag of SYS_ERR_INT_FLG reister will be cleared. Reset type: SYSRSn
8	DCC1	R-0/W1S	0h	0: No effect 1: DCC1 flag of SYS_ERR_INT_FLG reister will be cleared. Reset type: SYSRSn
7	DCC0	R-0/W1S	0h	0: No effect 1: DCC0 flag of SYS_ERR_INT_FLG reister will be cleared. Reset type: SYSRSn
6	AUX_PLL_SLIP_NOTSUPPORTED	R-0/W1S	0h	RESERVED: This bit is reserved and the value set should always be "0" Reset type: SYSRSn
5	SYS_PLL_SLIP_NOTSUPPORTED	R-0/W1S	0h	RESERVED: This bit is reserved and the value set should always be "0" Reset type: SYSRSn
4	RAM_ACC_VIOL	R-0/W1S	0h	0: No effect 1: RAM_ACC_VIOL flag of SYS_ERR_INT_FLG reister will be cleared. Reset type: SYSRSn
3	FLASH_CORRECTABLE_ERR	R-0/W1S	0h	0: No effect 1: FLASH_CORRECTABLE_ERR flag of SYS_ERR_INT_FLG reister will be cleared. Reset type: SYSRSn
2	RAM_CORRECTABLE_ERR	R-0/W1S	0h	0: No effect 1: RAM_CORRECTABLE_ERR flag of SYS_ERR_INT_FLG reister will be cleared. Reset type: SYSRSn
1	EMIF_ERR	R-0/W1S	0h	0: No effect 1: EMIF_ERR flag of SYS_ERR_INT_FLG reister will be cleared. Reset type: SYSRSn

Table 3-372. SYS_ERR_INT_CLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	GINT	R-0/W1S	0h	0: No effect 1: GINT flag of SYS_ERR_INT_FLG reister will be cleared. Reset type: SYSRSn

3.16.18.7 SYS_ERR_INT_SET Register (Offset = 14h) [Reset = 0h]

SYS_ERR_INT_SET is shown in Figure 3-343 and described in Table 3-373.

Return to the Summary Table.

SYS_ERR_INT_FLG set register

Figure 3-343. SYS_ERR_INT_SET Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED						DCC2	DCC1
R-0h						R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
DCC0	AUX_PLL_SLIP_NOTSUPPORTED	SYS_PLL_SLIP_NOTSUPPORTED	RAM_ACC_VIOL	FLASH_CORRECTABLE_ERR	RAM_CORRECTABLE_ERR	EMIF_ERR	RESERVED
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0h

Table 3-373. SYS_ERR_INT_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	A value of 0xa5a5 to this field would enable write to the other bit fields of this register. Any other value written to KEY field would block the write to the other fields of this register. Note: Only a 32 bit write to this register will succeed in updating the fields of this rigister, provided the correct value written to the KEY field simultaneously Reset type: SYSRSn
15-10	RESERVED	R	0h	Reserved
9	DCC2	R-0/W1S	0h	0: No effect 1: DCC2 flag of SYS_ERR_INT_FLG reister will be set. Reset type: SYSRSn
8	DCC1	R-0/W1S	0h	0: No effect 1: DCC1 flag of SYS_ERR_INT_FLG reister will be set. Reset type: SYSRSn
7	DCC0	R-0/W1S	0h	0: No effect 1: DCC0 flag of SYS_ERR_INT_FLG reister will be set. Reset type: SYSRSn
6	AUX_PLL_SLIP_NOTSUPPORTED	R-0/W1S	0h	RESERVED: This bit is reserved and the value set should always be "0" Reset type: SYSRSn
5	SYS_PLL_SLIP_NOTSUPPORTED	R-0/W1S	0h	RESERVED: This bit is reserved and the value set should always be "0" Reset type: SYSRSn
4	RAM_ACC_VIOL	R-0/W1S	0h	0: No effect 1: RAM_ACC_VIOL flag of SYS_ERR_INT_FLG reister will be set. Reset type: SYSRSn
3	FLASH_CORRECTABLE_ERR	R-0/W1S	0h	0: No effect 1: FLASH_CORRECTABLE_ERR flag of SYS_ERR_INT_FLG reister will be set. Reset type: SYSRSn

Table 3-373. SYS_ERR_INT_SET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	RAM_CORRECTABLE_ERR	R-0/W1S	0h	0: No effect 1: RAM_CORRECTABLE_ERR flag of SYS_ERR_INT_FLG reister will be set. Reset type: SYSRSn
1	EMIF_ERR	R-0/W1S	0h	0: No effect 1: EMIF_ERR flag of SYS_ERR_INT_FLG reister will be set. Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

3.16.18.8 SYS_ERR_MASK Register (Offset = 16h) [Reset = 60h]

SYS_ERR_MASK is shown in Figure 3-344 and described in Table 3-374.

Return to the Summary Table.

SYS_ERR_MASK register

Figure 3-344. SYS_ERR_MASK Register

31	30	29	28	27	26	25	24
KEY							
R/W-0h							
23	22	21	20	19	18	17	16
KEY							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED						DCC2	DCC1
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DCC0	AUX_PLL_SLIP	SYS_PLL_SLIP	RAM_ACC_VIOL	FLASH_CORRECTABLE_ERR	RAM_CORRECTABLE_ERR	EMIF_ERR	RESERVED
R/W-0h	R/W-1h	R/W-1h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

Table 3-374. SYS_ERR_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R/W	0h	A value of 0xa5a5 to this field would enable write to the other bit fields of this register. Any other value written to KEY field would block the write to the other fields of this register. Note: Only a 32 bit write to this register will succeed in updating the fields of this rigister, provided the correct value written to the KEY field simultaneously Reset type: SYSRSn
15-10	RESERVED	R	0h	Reserved
9	DCC2	R/W	0h	0: DCC2 flag of SYS_ERR_INT_FLG reister will be set on a hardware event. 1: DCC2 flag of SYS_ERR_INT_FLG reister will not be set on a hardware event. Reset type: SYSRSn
8	DCC1	R/W	0h	0: DCC1 flag of SYS_ERR_INT_FLG reister will be set on a hardware event. 1: DCC1 flag of SYS_ERR_INT_FLG reister will not be set on a hardware event. Reset type: SYSRSn
7	DCC0	R/W	0h	0: DCC0 flag of SYS_ERR_INT_FLG reister will be set on a hardware event. 1: DCC0 flag of SYS_ERR_INT_FLG reister will not be set on a hardware event. Reset type: SYSRSn
6	AUX_PLL_SLIP	R/W	1h	RESERVED: This bit is reserved and the value set should always be "1" Note: This bit must always be set to 1. Reset type: SYSRSn
5	SYS_PLL_SLIP	R/W	1h	RESERVED: This bit is reserved and the value set should always be "1" Note: This bit must always be set to 1. Reset type: SYSRSn

Table 3-374. SYS_ERR_MASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	RAM_ACC_VIOL	R/W	0h	0: RAM_ACC_VIOL flag of SYS_ERR_INT_FLG reister will be set on a hardware event. 1: RAM_ACC_VIOL flag of SYS_ERR_INT_FLG reister will not be set on a hardware event. Reset type: SYSRSn
3	FLASH_CORRECTABLE_ERR	R/W	0h	0: FLASH_CORRECTABLE_ERR flag of SYS_ERR_INT_FLG reister will be set on a hardware event. 1: FLASH_CORRECTABLE_ERR flag of SYS_ERR_INT_FLG reister will not be set on a hardware event. Reset type: SYSRSn
2	RAM_CORRECTABLE_ERR	R/W	0h	0: RAM_CORRECTABLE_ERR flag of SYS_ERR_INT_FLG reister will be set on a hardware event. 1: RAM_CORRECTABLE_ERR flag of SYS_ERR_INT_FLG reister will not be set on a hardware event. Reset type: SYSRSn
1	EMIF_ERR	R/W	0h	0: EMIF_ERR flag of SYS_ERR_INT_FLG reister will be set on a hardware event. 1: EMIF_ERR flag of SYS_ERR_INT_FLG reister will not be set on a hardware event. Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

3.16.19 TEST_ERROR_REGS Registers

Table 3-375 lists the memory-mapped registers for the TEST_ERROR_REGS registers. All register offset addresses not listed in Table 3-375 should be considered as reserved locations and the register contents should not be modified.

Table 3-375. TEST_ERROR_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CPU_RAM_TEST_ERROR_STS	Ram Test: Error Status Register		Go
2h	CPU_RAM_TEST_ERROR_STS_CLR	Ram Test: Error Status Clear Register		Go
4h	CPU_RAM_TEST_ERROR_ADDR	Ram Test: Error address register		Go

Complex bit access types are encoded to fit into small table cells. Table 3-376 shows the codes that are used for access types in this section.

Table 3-376. TEST_ERROR_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.19.1 CPU_RAM_TEST_ERROR_STS Register (Offset = 0h) [Reset = 0h]

CPU_RAM_TEST_ERROR_STS is shown in Figure 3-345 and described in Table 3-377.

Return to the Summary Table.

Ram Test: Error Status Register

Figure 3-345. CPU_RAM_TEST_ERROR_STS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						UNC_ERROR	COR_ERROR
R-0h						R-0h	R-0h

Table 3-377. CPU_RAM_TEST_ERROR_STS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	UNC_ERROR	R	0h	0: Indicates that there were no "un-correctable errors" generated in the RAM/ROM test mode. 1: Indicates that "un-correctable errors" wer generated in the RAM/ROM test mode. Reset type: SYSRSn
0	COR_ERROR	R	0h	0: Indicates that there were no "correctable errors" generated in the RAM/ROM test mode. 1: Indicates that "correctable errors" wer generated in the RAM/ROM test mode. Reset type: SYSRSn

3.16.19.2 CPU_RAM_TEST_ERROR_STS_CLR Register (Offset = 2h) [Reset = 0h]

CPU_RAM_TEST_ERROR_STS_CLR is shown in Figure 3-346 and described in Table 3-378.

Return to the Summary Table.

Ram Test: Error Status Clear Register

Figure 3-346. CPU_RAM_TEST_ERROR_STS_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						UNC_ERROR	COR_ERROR
R-0h						R-0/W1S-0h	R-0/W1S-0h

Table 3-378. CPU_RAM_TEST_ERROR_STS_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	UNC_ERROR	R-0/W1S	0h	0: No effect. 1: Clears the corresponding bit in CPU_RAM_TEST_ERROR_STS register. Reset type: SYSRSn
0	COR_ERROR	R-0/W1S	0h	0: No effect. 1: Clears the corresponding bit in CPU_RAM_TEST_ERROR_STS register. Reset type: SYSRSn

3.16.19.3 CPU_RAM_TEST_ERROR_ADDR Register (Offset = 4h) [Reset = 0h]

CPU_RAM_TEST_ERROR_ADDR is shown in Figure 3-347 and described in Table 3-379.

Return to the Summary Table.

Ram Test: Error address register

Figure 3-347. CPU_RAM_TEST_ERROR_ADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R-0h																															

Table 3-379. CPU_RAM_TEST_ERROR_ADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R	0h	Address of the location where error was detected in RAM/ROM test modes. Reset type: SYSRSn

3.16.20 UID_REGS Registers

Table 3-380 lists the memory-mapped registers for the UID_REGS registers. All register offset addresses not listed in Table 3-380 should be considered as reserved locations and the register contents should not be modified.

Table 3-380. UID_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	UID_PSRAND0	UID Psuedo-random 192 bit number		Go
2h	UID_PSRAND1	UID Psuedo-random 192 bit number		Go
4h	UID_PSRAND2	UID Psuedo-random 192 bit number		Go
6h	UID_PSRAND3	UID Psuedo-random 192 bit number		Go
8h	UID_PSRAND4	UID Psuedo-random 192 bit number		Go
Ah	UID_PSRAND5	UID Psuedo-random 192 bit number		Go
Ch	UID_UNIQUE	UID Unique 32 bit number		Go
Eh	UID_CHECKSUM	UID Checksum		Go

Complex bit access types are encoded to fit into small table cells. Table 3-381 shows the codes that are used for access types in this section.

Table 3-381. UID_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.20.1 UID_PSRAND0 Register (Offset = 0h) [Reset = X]

UID_PSRAND0 is shown in Figure 3-348 and described in Table 3-382.

Return to the Summary Table.

UID Psuedo-random 192 bit number

Figure 3-348. UID_PSRAND0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RandomID																															
R-X																															

Table 3-382. UID_PSRAND0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RandomID	R	X	Psuedorandom portion of the UID Reset type: N/A

3.16.20.2 UID_PSRAND1 Register (Offset = 2h) [Reset = X]

UID_PSRAND1 is shown in Figure 3-349 and described in Table 3-383.

Return to the Summary Table.

UID Psuedo-random 192 bit number

Figure 3-349. UID_PSRAND1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RandomID																															
R-X																															

Table 3-383. UID_PSRAND1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RandomID	R	X	Psuedorandom portion of the UID Reset type: N/A

3.16.20.3 UID_PSRAND2 Register (Offset = 4h) [Reset = X]

UID_PSRAND2 is shown in Figure 3-350 and described in Table 3-384.

Return to the Summary Table.

UID Psuedo-random 192 bit number

Figure 3-350. UID_PSRAND2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RandomID																															
R-X																															

Table 3-384. UID_PSRAND2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RandomID	R	X	Psuedorandom portion of the UID Reset type: N/A

3.16.20.4 UID_PSRAND3 Register (Offset = 6h) [Reset = X]

UID_PSRAND3 is shown in Figure 3-351 and described in Table 3-385.

Return to the Summary Table.

UID Psuedo-random 192 bit number

Figure 3-351. UID_PSRAND3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RandomID																															
R-X																															

Table 3-385. UID_PSRAND3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RandomID	R	X	Psuedorandom portion of the UID Reset type: N/A

3.16.20.5 UID_PSRAND4 Register (Offset = 8h) [Reset = X]

UID_PSRAND4 is shown in Figure 3-352 and described in Table 3-386.

Return to the Summary Table.

UID Psuedo-random 192 bit number

Figure 3-352. UID_PSRAND4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RandomID																															
R-X																															

Table 3-386. UID_PSRAND4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RandomID	R	X	Psuedorandom portion of the UID Reset type: N/A

3.16.20.6 UID_PSRAND5 Register (Offset = Ah) [Reset = X]

UID_PSRAND5 is shown in Figure 3-353 and described in Table 3-387.

Return to the Summary Table.

UID Psuedo-random 192 bit number

Figure 3-353. UID_PSRAND5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RandomID																															
R-X																															

Table 3-387. UID_PSRAND5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RandomID	R	X	Psuedorandom portion of the UID Reset type: N/A

3.16.20.7 UID_UNIQUE Register (Offset = Ch) [Reset = X]

UID_UNIQUE is shown in Figure 3-354 and described in Table 3-388.

Return to the Summary Table.

UID Unique 32 bit number

Figure 3-354. UID_UNIQUE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UniqueID																															
R-X																															

Table 3-388. UID_UNIQUE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UniqueID	R	X	Unique portion of the UID. This identifier will be unique across all devices with the same PARTIDH. Reset type: N/A

3.16.20.8 UID_CHECKSUM Register (Offset = Eh) [Reset = X]

UID_CHECKSUM is shown in Figure 3-355 and described in Table 3-389.

Return to the Summary Table.

Fletcher checksum of UID_PSRAND and UID_UNIQUE registers

Figure 3-355. UID_CHECKSUM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Checksum																															
R-X																															

Table 3-389. UID_CHECKSUM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Checksum	R	X	Fletcher checksum of UID_PSRANDx and UID_UINIQUE Reset type: N/A

3.16.21 WD_REGS Registers

Table 3-390 lists the memory-mapped registers for the WD_REGS registers. All register offset addresses not listed in Table 3-390 should be considered as reserved locations and the register contents should not be modified.

Table 3-390. WD_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
22h	SCSR	System Control & Status Register	EALLOW	Go
23h	WDCNTR	Watchdog Counter Register	EALLOW	Go
25h	WDKEY	Watchdog Reset Key Register	EALLOW	Go
29h	WDCR	Watchdog Control Register	EALLOW	Go
2Ah	WDWCR	Watchdog Windowed Control Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-391 shows the codes that are used for access types in this section.

Table 3-391. WD_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.21.1 SCSR Register (Offset = 22h) [Reset = 5h]

SCSR is shown in Figure 3-356 and described in Table 3-392.

Return to the Summary Table.

System Control & Status Register
It is recommended to only use 16 bit accesses to write to this register. Use a read-modify-write instruction may inadvertently clear other bits.

Figure 3-356. SCSR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED					WDINTS	WDENINT	WDOVERRIDE
R-0-0h					R-1h	R/W-0h	R/W1C-1h

Table 3-392. SCSR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R-0	0h	Reserved
2	WDINTS	R	1h	Watchdog Interrupt Status This bit indicates the state of the active-low watchdog interrupt signal (synchronized to SYSCLK). If the watchdog interrupt is used to wake the system from a low-power mode, then that mode should only be entered while this bit is high. Likewise, this bit must go high before the watchdog can be safely disabled and re-enabled. Reset type: SYSRSn 0h (R/W) = The watchdog interrupt signal is active. 1h (R/W) = The watchdog interrupt signal is inactive.
1	WDENINT	R/W	0h	Watchdog Interrupt Enable/Reset Disable This bit determines whether the watchdog triggers an interrupt (WAKE/WDOG) or a reset (WDRS) when the counter expires. Reset type: SYSRSn 0h (R/W) = Counter expiration triggers a reset. This is the default state on power-up and after any system reset. 1h (R/W) = Counter expiration triggers an interrupt.
0	WDOVERRIDE	R/W1C	1h	If this bit is set to 1, the user is allowed to change the state of the Watchdog disable (WDDIS) bit in the Watchdog Control (WDCR) register. If the WDOVERRIDE bit is cleared, by writing a 1 the WDDIS bit cannot be modified by the user. Writing a 0 will have no effect. If this bit is cleared, then it will remain in this state until a reset occurs. The current state of this bit is readable by the user. Reset type: SYSRSn

3.16.21.2 WDCNTR Register (Offset = 23h) [Reset = 0h]

WDCNTR is shown in Figure 3-357 and described in Table 3-393.

Return to the Summary Table.

Watchdog Counter Register

Figure 3-357. WDCNTR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
WDCNTR							
R-0h							

Table 3-393. WDCNTR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R-0	0h	Reserved
7-0	WDCNTR	R	0h	Watchdog Counter These bits contain the current value of the watchdog counter. This counter increments with each WDCLK (INTOSC1) cycle. If the counter overflows, either an interrupt or a reset is generated based on the value of the WDINTEN bit in the SCSR register. If the correct value is written to the WDKEY register, this counter is reset to zero. Reset type: IORSn

3.16.21.3 WDKEY Register (Offset = 25h) [Reset = 0h]

WDKEY is shown in Figure 3-358 and described in Table 3-394.

Return to the Summary Table.

Watchdog Reset Key Register

Figure 3-358. WDKEY Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
WDKEY							
R/W-0h							

Table 3-394. WDKEY Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R-0	0h	Reserved
7-0	WDKEY	R/W	0h	Watchdog Counter Reset Writing 0x55 followed by 0xAA will cause the watchdog counter to reset to zero, preventing an overflow. Writing other values has no effect. Reads of this register return the value of the WDCR register. Reset type: IORSn

3.16.21.4 WDCR Register (Offset = 29h) [Reset = 0h]

WDCR is shown in Figure 3-359 and described in Table 3-395.

Return to the Summary Table.

Watchdog Control Register
This memory mapped register requires a delay between subsequent writes to the register, otherwise a second write can be lost. The required delay is 69 SYSCLK cycles for a 200 MHz device, 45 SYSCLK cycles for a 120 MHz device, and 39 SYSCLK cycles for a 100 MHz device. This delay can be realized by adding NOP instructions corresponding to the required delay cycles.

Figure 3-359. WDCR Register

15	14	13	12	11	10	9	8
RESERVED				WDPRECLKDIV			
R-0-0h				R/W-0h			
7	6	5	4	3	2	1	0
WDFLG	WDDIS	WDCHK			WDPS		
R/W1S-0h	R/W-0h	R-0/W-0h			R/W-0h		

Table 3-395. WDCR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R-0	0h	Reserved
11-8	WDPRECLKDIV	R/W	0h	Watchdog Clock Pre-divider These bits determine the watchdog clock pre-divider, which is the first of the two dividers between INTOSC1 and the watchdog counter clock (WDCLK). The frequency of WDCLK is given by the formulas: PREDIVCLK = INTOSC1 / Pre-divider WDCLK = PREDIVCLK / Prescaler Reset type: IORSn 0h (R/W) = PREDIVCLK = INTOSC1 / 512 1h (R/W) = PREDIVCLK = INTOSC1 / 1024 2h (R/W) = PREDIVCLK = INTOSC1 / 2048 3h (R/W) = PREDIVCLK = INTOSC1 / 4096 4h (R/W) = Reserved 5h (R/W) = Reserved 6h (R/W) = Reserved 7h (R/W) = Reserved 8h (R/W) = PREDIVCLK = INTOSC1 / 2 9h (R/W) = PREDIVCLK = INTOSC1 / 4 Ah (R/W) = PREDIVCLK = INTOSC1 / 8 Bh (R/W) = PREDIVCLK = INTOSC1 / 16 Ch (R/W) = PREDIVCLK = INTOSC1 / 32 Dh (R/W) = PREDIVCLK = INTOSC1 / 64 Eh (R/W) = PREDIVCLK = INTOSC1 / 128 Fh (R/W) = PREDIVCLK = INTOSC1 / 256
7	WDFLG	R/W1S	0h	Watchdog reset status flag bit. This bit, if set, indicates a watchdog reset (WDRSTn) generated the reset condition. If 0, then it was en external device or power-up reset condition. This bit remains latched until the user writes a 1 to clear the condition. Writes of 0 will be ignored. Reset type: IORSn
6	WDDIS	R/W	0h	Watchdog Disable Setting this bit disables the watchdog module. Clearing this bit enables the watchdog module. This bit can be locked by the WDOVERRIDE bit in the SCSR register. The watchdog is enabled on reset. Reset type: IORSn

Table 3-395. WDCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-3	WDCHK	R-0/W	0h	Watchdog Check Bits During any write to this register, these bits must be written with the value 101 (binary). Writing any other value will immediately trigger the watchdog reset or interrupt. Reset type: IORSn
2-0	WDPS	R/W	0h	Watchdog Clock Prescaler These bits determine the watchdog clock prescaler, which is the second of the two dividers between INTOSC1 and the watchdog counter clock (WDCLK). The frequency of WDCLK is given by the formulas: PREDIVCLK = INTOSC1 / Pre-divider WDCLK = PREDIVCLK / Prescaler The watchdog reset or interrupt pulse is 512 INTOSC1 cycles long, so the counter period must be longer. To guarantee this, the product of the prescaler and pre-divider must be greater than or equal to four. The default prescaler value is 1. Reset type: IORSn 0h (R/W) = WDCLK = PREDIVCLK / 1 1h (R/W) = WDCLK = PREDIVCLK / 1 2h (R/W) = WDCLK = PREDIVCLK / 2 3h (R/W) = WDCLK = PREDIVCLK / 4 4h (R/W) = WDCLK = PREDIVCLK / 8 5h (R/W) = WDCLK = PREDIVCLK / 16 6h (R/W) = WDCLK = PREDIVCLK / 32 7h (R/W) = WDCLK = PREDIVCLK / 64

3.16.21.5 WDWCR Register (Offset = 2Ah) [Reset = 0h]

WDWCR is shown in Figure 3-360 and described in Table 3-396.

Return to the Summary Table.

Watchdog Windowed Control Register

Figure 3-360. WDWCR Register

15	14	13	12	11	10	9	8
RESERVED							FIRSTKEY
R-0-0h							R-0h
7	6	5	4	3	2	1	0
MIN							
R/W-0h							

Table 3-396. WDWCR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	RESERVED	R-0	0h	Reserved
8	FIRSTKEY	R	0h	This bit indicates if the 1st valid WDKEY (0x55 + 0xAA) got detected after MIN was configured to a non-zero value 0: First Valid Key after non-zero MIN configuration has not happened yet 1: First Valid key after non-zero MIN configuration got detected Notes: [1] If MIN = 0, this bit is never set [2] If MIN is changed back to 0x0 from a non-zero value, this bit is auto-cleared [3] This bit is added for debug purposes only Reset type: IORSn
7-0	MIN	R/W	0h	Watchdog Window Threshold These bits specify the lower limit of the watchdog counter reset window. If the counter is reset via the WDKEY register before the counter value reaches the value in this register, the watchdog immediately triggers a reset or interrupt. Reset type: IORSn

3.16.22 XINT_REGS Registers

Table 3-397 lists the memory-mapped registers for the XINT_REGS registers. All register offset addresses not listed in Table 3-397 should be considered as reserved locations and the register contents should not be modified.

Table 3-397. XINT_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	XINT1CR	XINT1 configuration register		Go
1h	XINT2CR	XINT2 configuration register		Go
2h	XINT3CR	XINT3 configuration register		Go
3h	XINT4CR	XINT4 configuration register		Go
4h	XINT5CR	XINT5 configuration register		Go
8h	XINT1CTR	XINT1 counter register		Go
9h	XINT2CTR	XINT2 counter register		Go
Ah	XINT3CTR	XINT3 counter register		Go

Complex bit access types are encoded to fit into small table cells. Table 3-398 shows the codes that are used for access types in this section.

Table 3-398. XINT_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.16.22.1 XINT1CR Register (Offset = 0h) [Reset = 0h]

XINT1CR is shown in Figure 3-361 and described in Table 3-399.

Return to the Summary Table.

XINT1 configuration register

Figure 3-361. XINT1CR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				POLARITY		RESERVED	ENABLE
R-0-0h				R/W-0h		R-0-0h	R/W-0h

Table 3-399. XINT1CR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R-0	0h	Reserved
3-2	POLARITY	R/W	0h	00: Interrupt is selected as negative edge triggered 01: Interrupt is selected as positive edge triggered 10: Interrupt is selected as negative edge triggered 11: Interrupt is selected as positive or negative edge triggered Reset type: SYSRSn
1	RESERVED	R-0	0h	Reserved
0	ENABLE	R/W	0h	0: Interrupt Disabled 1: Interrupt Enabled Reset type: SYSRSn

3.16.22.2 XINT2CR Register (Offset = 1h) [Reset = 0h]

XINT2CR is shown in Figure 3-362 and described in Table 3-400.

Return to the Summary Table.

XINT2 configuration register

Figure 3-362. XINT2CR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				POLARITY		RESERVED	ENABLE
R-0-0h				R/W-0h		R-0-0h	R/W-0h

Table 3-400. XINT2CR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R-0	0h	Reserved
3-2	POLARITY	R/W	0h	00: Interrupt is selected as negative edge triggered 01: Interrupt is selected as positive edge triggered 10: Interrupt is selected as negative edge triggered 11: Interrupt is selected as positive or negative edge triggered Reset type: SYSRSn
1	RESERVED	R-0	0h	Reserved
0	ENABLE	R/W	0h	0: Interrupt Disabled 1: Interrupt Enabled Reset type: SYSRSn

3.16.22.3 XINT3CR Register (Offset = 2h) [Reset = 0h]

XINT3CR is shown in Figure 3-363 and described in Table 3-401.

Return to the Summary Table.

XINT3 configuration register

Figure 3-363. XINT3CR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				POLARITY		RESERVED	ENABLE
R-0-0h				R/W-0h		R-0-0h	R/W-0h

Table 3-401. XINT3CR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R-0	0h	Reserved
3-2	POLARITY	R/W	0h	00: Interrupt is selected as negative edge triggered 01: Interrupt is selected as positive edge triggered 10: Interrupt is selected as negative edge triggered 11: Interrupt is selected as positive or negative edge triggered Reset type: SYSRSn
1	RESERVED	R-0	0h	Reserved
0	ENABLE	R/W	0h	0: Interrupt Disabled 1: Interrupt Enabled Reset type: SYSRSn

3.16.22.4 XINT4CR Register (Offset = 3h) [Reset = 0h]

XINT4CR is shown in Figure 3-364 and described in Table 3-402.

Return to the Summary Table.

XINT4 configuration register

Figure 3-364. XINT4CR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				POLARITY		RESERVED	ENABLE
R-0-0h				R/W-0h		R-0-0h	R/W-0h

Table 3-402. XINT4CR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R-0	0h	Reserved
3-2	POLARITY	R/W	0h	00: Interrupt is selected as negative edge triggered 01: Interrupt is selected as positive edge triggered 10: Interrupt is selected as negative edge triggered 11: Interrupt is selected as positive or negative edge triggered Reset type: SYSRSn
1	RESERVED	R-0	0h	Reserved
0	ENABLE	R/W	0h	0: Interrupt Disabled 1: Interrupt Enabled Reset type: SYSRSn

3.16.22.5 XINT5CR Register (Offset = 4h) [Reset = 0h]

XINT5CR is shown in Figure 3-365 and described in Table 3-403.

Return to the Summary Table.

XINT5 configuration register

Figure 3-365. XINT5CR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				POLARITY		RESERVED	ENABLE
R-0-0h				R/W-0h		R-0-0h	R/W-0h

Table 3-403. XINT5CR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R-0	0h	Reserved
3-2	POLARITY	R/W	0h	00: Interrupt is selected as negative edge triggered 01: Interrupt is selected as positive edge triggered 10: Interrupt is selected as negative edge triggered 11: Interrupt is selected as positive or negative edge triggered Reset type: SYSRSn
1	RESERVED	R-0	0h	Reserved
0	ENABLE	R/W	0h	0: Interrupt Disabled 1: Interrupt Enabled Reset type: SYSRSn

3.16.22.6 XINT1CTR Register (Offset = 8h) [Reset = 0h]

XINT1CTR is shown in Figure 3-366 and described in Table 3-404.

Return to the Summary Table.

XINT1 counter register

Figure 3-366. XINT1CTR Register

15	14	13	12	11	10	9	8
INTCTR							
R-0h							
7	6	5	4	3	2	1	0
INTCTR							
R-0h							

Table 3-404. XINT1CTR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	INTCTR	R	0h	This is a free running 16-bit up-counter that is clocked at the SYSCLKOUT rate. The counter value is reset to 0x0000 when a valid interrupt edge is detected and then continues counting until the next valid interrupt edge is detected. The counter must only be reset by the selected POLARITY edge as selected in the respective interrupt control register. When the interrupt is disabled, the counter will stop. The counter is a free-running counter and will wrap around to zero when the max value is reached. The counter is a read only register and can only be reset to zero by a valid interrupt edge or by reset. Reset type: SYSRSn

3.16.22.7 XINT2CTR Register (Offset = 9h) [Reset = 0h]

XINT2CTR is shown in Figure 3-367 and described in Table 3-405.

Return to the Summary Table.

XINT2 counter register

Figure 3-367. XINT2CTR Register

15	14	13	12	11	10	9	8
INTCTR							
R-0h							
7	6	5	4	3	2	1	0
INTCTR							
R-0h							

Table 3-405. XINT2CTR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	INTCTR	R	0h	This is a free running 16-bit up-counter that is clocked at the SYSCLKOUT rate. The counter value is reset to 0x0000 when a valid interrupt edge is detected and then continues counting until the next valid interrupt edge is detected. The counter must only be reset by the selected POLARITY edge as selected in the respective interrupt control register. When the interrupt is disabled, the counter will stop. The counter is a free-running counter and will wrap around to zero when the max value is reached. The counter is a read only register and can only be reset to zero by a valid interrupt edge or by reset. Reset type: SYSRSn

3.16.22.8 XINT3CTR Register (Offset = Ah) [Reset = 0h]

XINT3CTR is shown in Figure 3-368 and described in Table 3-406.

Return to the Summary Table.

XINT3 counter register

Figure 3-368. XINT3CTR Register

15	14	13	12	11	10	9	8
INTCTR							
R-0h							
7	6	5	4	3	2	1	0
INTCTR							
R-0h							

Table 3-406. XINT3CTR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	INTCTR	R	0h	This is a free running 16-bit up-counter that is clocked at the SYSCLKOUT rate. The counter value is reset to 0x0000 when a valid interrupt edge is detected and then continues counting until the next valid interrupt edge is detected. The counter must only be reset by the selected POLARITY edge as selected in the respective interrupt control register. When the interrupt is disabled, the counter will stop. The counter is a free-running counter and will wrap around to zero when the max value is reached. The counter is a read only register and can only be reset to zero by a valid interrupt edge or by reset. Reset type: SYSRSn

3.16.23 Register to Driverlib Function Mapping

3.16.23.1 ASYSCTL Registers to Driverlib Functions

Table 3-407. ASYSCTL Registers to Driverlib Functions

File	Driverlib Function
INTOSC1TRIM	
-	
INTOSC2TRIM	
-	
TSNSCTL	
asysctl.h	ASysCtl_enableTemperatureSensor
asysctl.h	ASysCtl_disableTemperatureSensor
LOCK	
asysctl.h	ASysCtl_lockTemperatureSensor
ANAREFTRIMA	
-	
ANAREFTRIMB	
-	
ANAREFTRIMC	
-	
ANAREFTRIMD	
-	

3.16.23.2 CPUTIMER Registers to Driverlib Functions

Table 3-408. CPUTIMER Registers to Driverlib Functions

File	Driverlib Function
TIM	
cputimer.h	CPUTimer_getTimerCount
PRD	
cputimer.h	CPUTimer_setPeriod
TCR	
cputimer.c	CPUTimer_setEmulationMode
cputimer.h	CPUTimer_clearOverflowFlag
cputimer.h	CPUTimer_disableInterrupt
cputimer.h	CPUTimer_enableInterrupt
cputimer.h	CPUTimer_reloadTimerCounter
cputimer.h	CPUTimer_stopTimer
cputimer.h	CPUTimer_resumeTimer
cputimer.h	CPUTimer_startTimer
cputimer.h	CPUTimer_getTimerOverflowStatus
TPR	
cputimer.h	CPUTimer_setPreScaler
TPRH	
cputimer.h	CPUTimer_setPreScaler

3.16.23.3 DCSM Registers to Driverlib Functions

Table 3-409. DCSM Registers to Driverlib Functions

File	Driverlib Function
Z1OTP_LINKPOINTER1	
-	
Z1OTP_LINKPOINTER2	
-	
Z1OTP_LINKPOINTER3	
-	
Z1OTP_JLM_ENABLE	
-	
Z1OTP_GPREG1	
-	
Z1OTP_GPREG2	
-	
Z1OTP_GPREG3	
-	
Z1OTP_GPREG4	
-	
Z1OTP_PSWDLOCK	
-	
Z1OTP_CRCLOCK	
-	
Z1OTP_JTAGPSWDH0	
-	
Z1OTP_JTAGPSWDH1	
-	
Z1OTP_CMACKEY0	
-	
Z1OTP_CMACKEY1	
-	
Z1OTP_CMACKEY2	
-	
Z1OTP_CMACKEY3	
-	
Z2OTP_LINKPOINTER1	
-	
Z2OTP_LINKPOINTER2	
-	
Z2OTP_LINKPOINTER3	
-	
Z2OTP_GPREG1	
-	
Z2OTP_GPREG2	
-	
Z2OTP_GPREG3	
-	

Table 3-409. DCSM Registers to Driverlib Functions (continued)

File	Driverlib Function
Z2OTP_GPREG4	
-	
Z2OTP_PSWDLOCK	
-	
Z2OTP_CRCLOCK	
-	
Z1_LINKPOINTER	
dcsm.c	DCSM_unlockZone1CSM
dcsm.c	DCSM_readZone1CSMPwd
dcsm.h	DCSM_getZone1LinkPointerError
Z1_OTPSECLOCK	
dcsm.h	DCSM_getZone1OTPSecureLockStatus
Z1_JLM_ENABLE	
-	
Z1_LINKPOINTERERR	
dcsm.h	DCSM_getZone1LinkPointerError
Z1_GPREG1	
-	
Z1_GPREG2	
-	
Z1_GPREG3	
-	
Z1_GPREG4	
-	
Z1_CSMKEY0	
dcsm.c	DCSM_unlockZone1CSM
dcsm.c	DCSM_writeZone1CSM
Z1_CSMKEY1	
dcsm.c	DCSM_unlockZone1CSM
dcsm.c	DCSM_writeZone1CSM
Z1_CSMKEY2	
dcsm.c	DCSM_unlockZone1CSM
dcsm.c	DCSM_writeZone1CSM
Z1_CSMKEY3	
dcsm.c	DCSM_unlockZone1CSM
dcsm.c	DCSM_writeZone1CSM
Z1_CR	
dcsm.h	DCSM_secureZone1
dcsm.h	DCSM_getZone1CSMSecurityStatus
dcsm.h	DCSM_getZone1ControlStatus
Z1_GRABSECT1R	
-	
Z1_GRABSECT2R	
-	
Z1_GRABSECT3R	

Table 3-409. DCSM Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
Z1_GRABRAM1R	
-	
Z1_GRABRAM2R	
-	
Z1_GRABRAM3R	
-	
Z1_EXEONLYSECT1R	
dcsm.c	DCSM_getZone1FlashEXEStatus
Z1_EXEONLYSECT2R	
dcsm.c	DCSM_getZone1FlashEXEStatus
Z1_EXEONLYRAM1R	
dcsm.c	DCSM_getZone1RAMEXEStatus
Z1_JTAGKEY0	
-	
Z1_JTAGKEY1	
-	
Z1_JTAGKEY2	
-	
Z1_JTAGKEY3	
-	
Z1_CMACKEY0	
-	
Z1_CMACKEY1	
-	
Z1_CMACKEY2	
-	
Z1_CMACKEY3	
-	
Z2_LINKPOINTER	
dcsm.c	DCSM_unlockZone2CSM
dcsm.c	DCSM_readZone2CSMPwd
dcsm.h	DCSM_getZone2LinkPointerError
Z2_OTPSECLOCK	
dcsm.h	DCSM_getZone2OTPSecureLockStatus
Z2_LINKPOINTERERR	
dcsm.h	DCSM_getZone2LinkPointerError
Z2_GPREG1	
-	
Z2_GPREG2	
-	
Z2_GPREG3	
-	
Z2_GPREG4	
-	

Table 3-409. DCSM Registers to Driverlib Functions (continued)

File	Driverlib Function
Z2_CSMKEY0	
dcsm.c	DCSM_unlockZone2CSM
dcsm.c	DCSM_writeZone2CSM
Z2_CSMKEY1	
dcsm.c	DCSM_unlockZone2CSM
dcsm.c	DCSM_writeZone2CSM
Z2_CSMKEY2	
dcsm.c	DCSM_unlockZone2CSM
dcsm.c	DCSM_writeZone2CSM
Z2_CSMKEY3	
dcsm.c	DCSM_unlockZone2CSM
dcsm.c	DCSM_writeZone2CSM
Z2_CR	
dcsm.h	DCSM_secureZone2
dcsm.h	DCSM_getZone2CSMSecurityStatus
dcsm.h	DCSM_getZone2ControlStatus
Z2_GRABSECT1R	
-	
Z2_GRABSECT2R	
-	
Z2_GRABSECT3R	
-	
Z2_GRABRAM1R	
-	
Z2_GRABRAM2R	
-	
Z2_GRABRAM3R	
-	
Z2_EXEONLYSECT1R	
dcsm.c	DCSM_getZone2FlashEXEStatus
Z2_EXEONLYSECT2R	
dcsm.c	DCSM_getZone2FlashEXEStatus
Z2_EXEONLYRAM1R	
dcsm.c	DCSM_getZone2RAMEXEStatus
FLSEM	
dcsm.c	DCSM_claimZoneSemaphore
dcsm.c	DCSM_releaseZoneSemaphore
SECTSTAT1	
dcsm.h	DCSM_getFlashSectorZone
SECTSTAT2	
-	
SECTSTAT3	
-	
RAMSTAT1	
dcsm.h	DCSM_getRAMZone

Table 3-409. DCSM Registers to Driverlib Functions (continued)

File	Driverlib Function
RAMSTAT2	
-	
RAMSTAT3	
-	
SECERRSTAT	
dcsm.h	DCSM_getFlashErrorStatus
SECERRCLR	
dcsm.h	DCSM_clearFlashErrorStatus
SECERRFRC	
dcsm.h	DCSM_forceFlashErrorStatus

3.16.23.4 MEMCFG Registers to Driverlib Functions

Table 3-410. MEMCFG Registers to Driverlib Functions

File	Driverlib Function
DXLOCK	
memcfg.c	MemCfg_lockConfig
memcfg.c	MemCfg_unlockConfig
DXCOMMIT	
memcfg.c	MemCfg_commitConfig
DXACCPROT0	
memcfg.c	MemCfg_setProtection
DXTEST	
memcfg.c	MemCfg_setTestMode
DXINIT	
memcfg.c	MemCfg_initSections
memcfg.c	MemCfg_getInitStatus
DXINITDONE	
memcfg.c	MemCfg_getInitStatus
DXRAMTEST_LOCK	
memcfg.c	MemCfg_lockTestConfig
memcfg.c	MemCfg_unlockTestConfig
LSXLOCK	
memcfg.c	MemCfg_lockConfig
memcfg.c	MemCfg_unlockConfig
LSXCOMMIT	
memcfg.c	MemCfg_commitConfig
LSXMSEL	
memcfg.c	MemCfg_setLSRAMControllerSel
LSXCLAPGM	
memcfg.h	MemCfg_setCLAMemType
LSXACCPROT0	
memcfg.c	MemCfg_setProtection
LSXACCPROT1	
-	
LSXTEST	

Table 3-410. MEMCFG Registers to Driverlib Functions (continued)

File	Driverlib Function
memcfg.c	MemCfg_setTestMode
LSXINIT	
memcfg.c	MemCfg_initSections
memcfg.c	MemCfg_getInitStatus
LSXINITDONE	
memcfg.c	MemCfg_getInitStatus
LSXRAMTEST_LOCK	
memcfg.c	MemCfg_lockTestConfig
memcfg.c	MemCfg_unlockTestConfig
GSXLOCK	
memcfg.c	MemCfg_lockConfig
memcfg.c	MemCfg_unlockConfig
GSXCOMMIT	
memcfg.c	MemCfg_commitConfig
GSXMSEL	
memcfg.c	MemCfg_setGSRAMControllerSel
GSXACCPROT0	
memcfg.c	MemCfg_setProtection
GSXACCPROT1	
-	See GSXACCPROT0
GSXACCPROT2	
-	See GSXACCPROT0
GSXACCPROT3	
-	See GSXACCPROT0
GSXTEST	
memcfg.c	MemCfg_setTestMode
GSXINIT	
memcfg.c	MemCfg_initSections
memcfg.c	MemCfg_getInitStatus
GSXINITDONE	
memcfg.c	MemCfg_getInitStatus
GSXRAMTEST_LOCK	
memcfg.c	MemCfg_lockTestConfig
memcfg.c	MemCfg_unlockTestConfig
MSGXLOCK	
memcfg.c	MemCfg_lockConfig
memcfg.c	MemCfg_unlockConfig
MSGXCOMMIT	
memcfg.c	MemCfg_commitConfig
MSGXACCPROT0	
memcfg.c	MemCfg_setProtection
MSGXACCPROT1	
-	
MSGXACCPROT2	
-	

Table 3-410. MEMCFG Registers to Driverlib Functions (continued)

File	Driverlib Function
MSGXTEST	
memcfg.c	MemCfg_setTestMode
MSGXINIT	
memcfg.c	MemCfg_initSections
memcfg.c	MemCfg_getInitStatus
MSGXINITDONE	
memcfg.c	MemCfg_getInitStatus
MSGXRAMTEST_LOCK	
memcfg.c	MemCfg_lockTestConfig
memcfg.c	MemCfg_unlockTestConfig
ROM_LOCK	
memcfg.c	MemCfg_lockTestConfig
memcfg.c	MemCfg_unlockTestConfig
ROM_TEST	
memcfg.c	MemCfg_setTestMode
ROM_FORCE_ERROR	
memcfg.c	MemCfg_forceMemError
PERI_MEM_TEST_LOCK	
memcfg.c	MemCfg_lockTestConfig
memcfg.c	MemCfg_unlockTestConfig
PERI_MEM_TEST_CONTROL	
memcfg.c	MemCfg_forceMemError
memcfg.c	MemCfg_enablePeriMemTestMode
memcfg.c	MemCfg_disablePeriMemTestMode
EMIF1LOCK	
emif.h	EMIF_lockAccessConfig
emif.h	EMIF_unlockAccessConfig
EMIF1COMMIT	
emif.h	EMIF_commitAccessConfig
EMIF1MSEL	
emif.h	EMIF_selectController
EMIF1ACCPROT0	
emif.h	EMIF_setAccessProtection
EMIF2LOCK	
-	
EMIF2COMMIT	
-	
EMIF2ACCPROT0	
-	
NMAVFLG	
memcfg.h	MemCfg_getViolationInterruptStatus
NMAVSET	
memcfg.h	MemCfg_forceViolationInterrupt
NMAVCLR	
memcfg.h	MemCfg_clearViolationInterruptStatus

Table 3-410. MEMCFG Registers to Driverlib Functions (continued)

File	Driverlib Function
NMAVINTEN	
memcfg.h	MemCfg_enableViolationInterrupt
memcfg.h	MemCfg_disableViolationInterrupt
NMCPURDAVADDR	
memcfg.c	MemCfg_getViolationAddress
NMCPUWRAVADDR	
memcfg.c	MemCfg_getViolationAddress
NMCPUFAVADDR	
-	
NMDMAWRAVADDR	
-	
NMCLA1RDAVADDR	
-	
NMCLA1WRAVADDR	
-	
NMCLA1FAVADDR	
-	
NMDMARDAVADDR	
-	
MAVFLG	
memcfg.h	MemCfg_getViolationInterruptStatus
MAVSET	
memcfg.h	MemCfg_forceViolationInterrupt
MAVCLR	
memcfg.h	MemCfg_clearViolationInterruptStatus
MAVINTEN	
memcfg.h	MemCfg_enableViolationInterrupt
memcfg.h	MemCfg_disableViolationInterrupt
MCPUFAVADDR	
memcfg.c	MemCfg_getViolationAddress
MCPUWRAVADDR	
-	
MDMAWRAVADDR	
-	
UCERRFLG	
memcfg.h	MemCfg_getUncorrErrorStatus
UCERRSET	
memcfg.h	MemCfg_forceUncorrErrorStatus
UCERRCLR	
memcfg.h	MemCfg_clearUncorrErrorStatus
UCCPUREADDR	
memcfg.c	MemCfg_getUncorrErrorAddress
UCDMAREADDR	
memcfg.c	MemCfg_getUncorrErrorAddress
UCCLA1READDR	

Table 3-410. MEMCFG Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
UCECATRAMADDR	
-	
CERRFLG	
memcfg.h	MemCfg_getCorrErrorStatus
CERRSET	
memcfg.c	MemCfg_getCorrErrorAddress
memcfg.h	MemCfg_forceCorrErrorStatus
CERRCLR	
memcfg.c	MemCfg_getCorrErrorAddress
memcfg.h	MemCfg_clearCorrErrorStatus
CCPUREADDR	
memcfg.c	MemCfg_getCorrErrorAddress
CCLA1READDR	
-	
CERRCNT	
memcfg.h	MemCfg_getCorrErrorCount
CERRTHRES	
memcfg.h	MemCfg_setCorrErrorThreshold
CEINTFLG	
memcfg.h	MemCfg_getCorrErrorInterruptStatus
CEINTCLR	
memcfg.h	MemCfg_clearCorrErrorInterruptStatus
CEINTSET	
memcfg.h	MemCfg_forceCorrErrorInterrupt
CEINTEN	
memcfg.h	MemCfg_enableCorrErrorInterrupt
memcfg.h	MemCfg_disableCorrErrorInterrupt
ROMWAITSTATE	
memcfg.h	MemCfg_enableROMWaitState
memcfg.h	MemCfg_disableROMWaitState
ROMPREFETCH	
memcfg.h	MemCfg_enableROMPrefetch
memcfg.h	MemCfg_disableROMPrefetch
CPU_RAM_TEST_ERROR_STS	
memcfg.h	MemCfg_getDiagErrorStatus
memcfg.h	MemCfg_clearDiagErrorStatus
CPU_RAM_TEST_ERROR_STS_CLR	
memcfg.h	MemCfg_clearDiagErrorStatus
CPU_RAM_TEST_ERROR_ADDR	
memcfg.h	MemCfg_getDiagErrorAddress

3.16.23.5 NMI Registers to Driverlib Functions

Table 3-411. NMI Registers to Driverlib Functions

File	Driverlib Function
CFG	
sysctl.h	SysCtl_enableNMIGlobalInterrupt
FLG	
sysctl.h	SysCtl_getNMIStatus
sysctl.h	SysCtl_getNMIFlagStatus
sysctl.h	SysCtl_isNMIFlagSet
sysctl.h	SysCtl_clearNMIStatus
sysctl.h	SysCtl_clearAllNMIFlags
sysctl.h	SysCtl_forceNMIFlags
FLGCLR	
sysctl.h	SysCtl_clearNMIStatus
sysctl.h	SysCtl_clearAllNMIFlags
FLGFRC	
sysctl.h	SysCtl_forceNMIFlags
WDCNT	
sysctl.h	SysCtl_getNMIWatchdogCounter
WDPRD	
sysctl.h	SysCtl_setNMIWatchdogPeriod
sysctl.h	SysCtl_getNMIWatchdogPeriod
SHDFLG	
sysctl.h	SysCtl_getNMIShadowFlagStatus
sysctl.h	SysCtl_isNMIShadowFlagSet
ERRORSTS	
sysctl.h	SysCtl_isErrorTriggered
sysctl.h	SysCtl_getErrorPinStatus
sysctl.h	SysCtl_forceError
sysctl.h	SysCtl_clearError
ERRORSTSCLR	
sysctl.h	SysCtl_clearError
ERRORSTSFRC	
sysctl.h	SysCtl_forceError
ERRORCTL	
sysctl.h	SysCtl_selectErrPinPolarity
ERRORLOCK	
sysctl.h	SysCtl_lockErrControl

3.16.23.6 PIE Registers to Driverlib Functions

Table 3-412. PIE Registers to Driverlib Functions

File	Driverlib Function
CTRL	
interrupt.c	Interrupt_initModule
interrupt.c	Interrupt_defaultHandler
interrupt.h	Interrupt_enablePIE
interrupt.h	Interrupt_disablePIE

Table 3-412. PIE Registers to Driverlib Functions (continued)

File	Driverlib Function
ACK	
interrupt.c	Interrupt_disable
interrupt.h	Interrupt_clearACKGroup
IER1	
interrupt.c	Interrupt_initModule
interrupt.c	Interrupt_enable
interrupt.c	Interrupt_disable
IFR1	
interrupt.c	Interrupt_initModule
IER2	
interrupt.c	Interrupt_initModule
IFR2	
interrupt.c	Interrupt_initModule
IER3	
interrupt.c	Interrupt_initModule
IFR3	
interrupt.c	Interrupt_initModule
IER4	
interrupt.c	Interrupt_initModule
IFR4	
interrupt.c	Interrupt_initModule
IER5	
interrupt.c	Interrupt_initModule
IFR5	
interrupt.c	Interrupt_initModule
IER6	
interrupt.c	Interrupt_initModule
IFR6	
interrupt.c	Interrupt_initModule
IER7	
interrupt.c	Interrupt_initModule
IFR7	
interrupt.c	Interrupt_initModule
IER8	
interrupt.c	Interrupt_initModule
IFR8	
interrupt.c	Interrupt_initModule
IER9	
interrupt.c	Interrupt_initModule
IFR9	
interrupt.c	Interrupt_initModule
IER10	
interrupt.c	Interrupt_initModule
IFR10	
interrupt.c	Interrupt_initModule

Table 3-412. PIE Registers to Driverlib Functions (continued)

File	Driverlib Function
IER11	
interrupt.c	Interrupt_initModule
IFR11	
interrupt.c	Interrupt_initModule
IER12	
interrupt.c	Interrupt_initModule
IFR12	
interrupt.c	Interrupt_initModule

3.16.23.7 SYSCTL Registers to Driverlib Functions

Table 3-413. SYSCTL Registers to Driverlib Functions

File	Driverlib Function
CLKSEM	
sysctl.c	SysCtl_setSemOwner
sysctl.h	SysCtl_getSemOwner
CLKCFGLOCK1	
sysctl.c	SysCtl_lockClkConfig
CLKSRCCTL1	
sysctl.c	SysCtl_getClock
sysctl.c	SysCtl_setClock
sysctl.c	SysCtl_selectXTAL
sysctl.c	SysCtl_selectXTALSingleEnded
sysctl.c	SysCtl_selectOscSource
sysctl.c	SysCtl_selectOscSourceAuxPLL
sysctl.h	SysCtl_turnOnOsc
sysctl.h	SysCtl_turnOffOsc
CLKSRCCTL2	
can.h	CAN_selectClockSource
sysctl.c	SysCtl_getAuxClock
sysctl.c	SysCtl_setAuxClock
sysctl.c	SysCtl_selectOscSourceAuxPLL
CLKSRCCTL3	
sysctl.h	SysCtl_selectClockOutSource
SYSPLLCTL1	
sysctl.c	SysCtl_getClock
sysctl.c	SysCtl_setClock
SYSPLLMULT	
sysctl.c	SysCtl_getClock
sysctl.c	SysCtl_setClock
SYSPLLSTS	
sysctl.c	SysCtl_setClock
AUXPLLCTL1	
sysctl.c	SysCtl_getAuxClock
sysctl.c	SysCtl_setAuxClock
AUXPLLMULT	

Table 3-413. SYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
sysctl.c	SysCtl_getAuxClock
sysctl.c	SysCtl_setAuxClock
AUXPLLSTS	
sysctl.c	SysCtl_setAuxClock
SYSCLKDIVSEL	
sysctl.c	SysCtl_getClock
sysctl.c	SysCtl_setClock
sysctl.h	SysCtl_setPLLSysClk
AUXCLKDIVSEL	
sysctl.c	SysCtl_getAuxClock
sysctl.c	SysCtl_setAuxClock
sysctl.h	SysCtl_setAuxPLLClk
sysctl.h	SysCtl_setMCANClk
PERCLKDIVSEL	
sysctl.h	SysCtl_setEPWMClockDivider
sysctl.h	SysCtl_setEMIF1ClockDivider
sysctl.h	SysCtl_setEMIF2ClockDivider
XCLKOUTDIVSEL	
sysctl.h	SysCtl_setXClk
CLBCLKCTL	
sysctl.h	SysCtl_setCLBClk
sysctl.h	SysCtl_setCLBClkDivider
sysctl.h	SysCtl_CLBClkConfig
LOSPCP	
sysctl.c	SysCtl_getLowSpeedClock
sysctl.h	SysCtl_setLowSpeedClock
MCDCR	
sysctl.h	SysCtl_enableMCD
sysctl.h	SysCtl_disableMCD
sysctl.h	SysCtl_isMCDClockFailureDetected
sysctl.h	SysCtl_resetMCD
sysctl.h	SysCtl_connectMCDClockSource
sysctl.h	SysCtl_disconnectMCDClockSource
X1CNT	
sysctl.c	SysCtl_pollX1Counter
sysctl.h	SysCtl_getExternalOscCounterValue
sysctl.h	SysCtl_clearExternalOscCounterValue
XTALCR	
sysctl.c	SysCtl_setClock
sysctl.c	SysCtl_setAuxClock
sysctl.c	SysCtl_selectXTAL
sysctl.c	SysCtl_selectXTALSingleEnded
sysctl.c	SysCtl_selectOscSourceAuxPLL
sysctl.h	SysCtl_setExternalOscMode
sysctl.h	SysCtl_turnOnOsc

Table 3-413. SYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
ETHERCATCLKCTL	
sysctl.h	SysCtl_setECatClk
CMCLKCTL	
sysctl.h	SysCtl_setCMClk
sysctl.h	SysCtl_setEnetClk
CPUSYSLOCK1	
sysctl.c	SysCtl_lockSysConfig
CPUSYSLOCK2	
-	
PIEVERRADDR	
sysctl.h	SysCtl_getPIEVErrAddr
PCLKCR0	
sysctl.h	SysCtl_enablePeripheral
sysctl.h	SysCtl_disablePeripheral
PCLKCR1	
-	See PCLKCR0
PCLKCR2	
-	See PCLKCR0
PCLKCR3	
-	See PCLKCR0
PCLKCR4	
-	See PCLKCR0
PCLKCR6	
-	See PCLKCR0
PCLKCR7	
-	See PCLKCR0
PCLKCR8	
-	See PCLKCR0
PCLKCR9	
-	See PCLKCR0
PCLKCR10	
-	See PCLKCR0
PCLKCR11	
-	See PCLKCR0
PCLKCR13	
-	See PCLKCR0
PCLKCR14	
-	See PCLKCR0
PCLKCR16	
-	See PCLKCR0
PCLKCR17	
-	
PCLKCR18	
-	
PCLKCR20	

Table 3-413. SYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
PCLKCR21	
-	
PCLKCR22	
-	
PCLKCR23	
-	
SIMRESET	
sysctl.h	SysCtl_simulateReset
LPMCR	
sysctl.h	SysCtl_enterIdleMode
sysctl.h	SysCtl_enterStandbyMode
sysctl.h	SysCtl_setStandbyQualificationPeriod
sysctl.h	SysCtl_enableWatchdogStandbyWakeup
sysctl.h	SysCtl_disableWatchdogStandbyWakeup
GPIOLPMSEL0	
sysctl.h	SysCtl_enableLPMWakeupPin
sysctl.h	SysCtl_disableLPMWakeupPin
GPIOLPMSEL1	
sysctl.h	SysCtl_enableLPMWakeupPin
sysctl.h	SysCtl_disableLPMWakeupPin
TMR2CLKCTL	
cputimer.h	CPUTimer_selectClockSource
sysctl.h	SysCtl_setCputimer2Clk
RESCCLR	
sysctl.h	SysCtl_clearWatchdogResetStatus
RESC	
sysctl.h	SysCtl_getResetCause
sysctl.h	SysCtl_clearResetCause
sysctl.h	SysCtl_getWatchdogResetStatus
sysctl.h	SysCtl_clearWatchdogResetStatus
MCANWAKESTATUS	
sysctl.h	SysCtl_isMCANWakeStatusSet
sysctl.h	SysCtl_clearMCANWakeStatus
MCANWAKESTATUSCLR	
sysctl.h	SysCtl_clearMCANWakeStatus
CLA1TASKSRCSELLOCK	
-	
DMACHSRCSELLOCK	
-	
CLA1TASKSRCSEL1	
cla.c	CLA_setTriggerSource
CLA1TASKSRCSEL2	
cla.c	CLA_setTriggerSource
DMACHSRCSEL1	

Table 3-413. SYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
dma.c	DMA_configMode
DMACHSRCSEL2	
dma.c	DMA_configMode
DEVCFGLOCK1	
sysctl.h	SysCtl_lockCPUSelectRegs
PARTIDL	
sysctl.c	SysCtl_getDeviceParametric
PARTIDH	
sysctl.c	SysCtl_getDeviceParametric
REVID	
sysctl.h	SysCtl_getDeviceRevision
PERCNF1	
sysctl.h	SysCtl_readADCWrapper
sysctl.h	SysCtl_isPresentUSBPHY
FUSEERR	
sysctl.h	SysCtl_getEfuseError
SOFTPRES0	
sysctl.h	SysCtl_resetPeripheral
SOFTPRES1	
-	See SOFTPRES0
SOFTPRES2	
-	See SOFTPRES0
SOFTPRES3	
-	See SOFTPRES0
SOFTPRES4	
-	See SOFTPRES0
SOFTPRES6	
-	See SOFTPRES0
SOFTPRES7	
-	See SOFTPRES0
SOFTPRES8	
-	See SOFTPRES0
SOFTPRES9	
-	See SOFTPRES0
SOFTPRES10	
-	See SOFTPRES0
SOFTPRES11	
-	See SOFTPRES0
SOFTPRES13	
-	See SOFTPRES0
SOFTPRES14	
-	See SOFTPRES0
SOFTPRES16	
-	See SOFTPRES0
SOFTPRES17	

Table 3-413. SYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
SOFTPRES18	
-	
SOFTPRES20	
-	
SOFTPRES21	
-	
SOFTPRES23	
-	
CPUSEL0	
sysctl.h	SysCtl_selectCPUForPeripheralInstance
sysctl.h	SysCtl_selectCPUForPeripheral
CPUSEL1	
-	See CPUSEL0
CPUSEL2	
-	See CPUSEL0
CPUSEL4	
-	See CPUSEL0
CPUSEL5	
-	See CPUSEL0
CPUSEL6	
-	See CPUSEL0
CPUSEL7	
-	See CPUSEL0
CPUSEL8	
-	See CPUSEL0
CPUSEL9	
-	See CPUSEL0
CPUSEL11	
-	See CPUSEL0
CPUSEL12	
-	See CPUSEL0
CPUSEL14	
-	See CPUSEL0
CPUSEL15	
-	See CPUSEL0
CPUSEL16	
-	See CPUSEL0
CPUSEL18	
-	
CPUSEL25	
-	
CPU2RESCTL	
sysctl.c	SysCtl_controlCPU2Reset
RSTSTAT	

Table 3-413. SYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
sysctl.h	SysCtl_isCPU2Reset
sysctl.h	SysCtl_getCPU2ResetStatus
sysctl.h	SysCtl_clearCPU2ResetStatus
LPMSTAT	
sysctl.h	SysCtl_getCPU2LPMStatus
USBTYPE	
sysctl.c	SysCtl_configureType
sysctl.c	SysCtl_isConfigTypeLocked
ECAPTYPE	
sysctl.c	SysCtl_configureType
sysctl.c	SysCtl_isConfigTypeLocked
SDFMTYPE	
sysctl.c	SysCtl_configureType
sysctl.c	SysCtl_isConfigTypeLocked
MEMMAPTYPE	
sysctl.c	SysCtl_configureType
sysctl.c	SysCtl_isConfigTypeLocked
ADCA_AC	
-	
ADCB_AC	
-	
ADCC_AC	
-	
ADCD_AC	
-	
CMPSS1_AC	
-	
CMPSS2_AC	
-	
CMPSS3_AC	
-	
CMPSS4_AC	
-	
CMPSS5_AC	
-	
CMPSS6_AC	
-	
CMPSS7_AC	
-	
CMPSS8_AC	
-	
DACA_AC	
-	
DACB_AC	
-	

Table 3-413. SYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
DACC_AC	
-	
EPWM1_AC	
-	
EPWM2_AC	
-	
EPWM3_AC	
-	
EPWM4_AC	
-	
EPWM5_AC	
-	
EPWM6_AC	
-	
EPWM7_AC	
-	
EPWM8_AC	
-	
EPWM9_AC	
-	
EPWM10_AC	
-	
EPWM11_AC	
-	
EPWM12_AC	
-	
EPWM13_AC	
-	
EPWM14_AC	
-	
EPWM15_AC	
-	
EPWM16_AC	
-	
EQEP1_AC	
-	
EQEP2_AC	
-	
EQEP3_AC	
-	
ECAP1_AC	
-	
ECAP2_AC	
-	
ECAP3_AC	

Table 3-413. SYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
ECAP4_AC	
-	
ECAP5_AC	
-	
ECAP6_AC	
-	
ECAP7_AC	
-	
SDFM1_AC	
-	
SDFM2_AC	
-	
CLB1_AC	
-	
CLB2_AC	
-	
CLB3_AC	
-	
CLB4_AC	
-	
CLB5_AC	
-	
CLB6_AC	
-	
CLB7_AC	
-	
CLB8_AC	
-	
SPIA_AC	
-	
SPIB_AC	
-	
SPIC_AC	
-	
SPID_AC	
-	
PMBUS_A_AC	
-	
CAN_A_AC	
-	
CAN_B_AC	
-	
MCBSPA_AC	
-	

Table 3-413. SYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
MCBSPB_AC	
-	
USBA_AC	
-	
HRPWM_AC	
-	
ETHERCAT_AC	
-	
FSIATX_AC	
-	
FSIARX_AC	
-	
FSIBTX_AC	
-	
FSIBRX_AC	
-	
FSICRX_AC	
-	
FSIDRX_AC	
-	
FSIERX_AC	
-	
FSIFRX_AC	
-	
FSIGRX_AC	
-	
FSIHRX_AC	
-	
MCANA_AC	
-	
PERIPH_AC_LOCK	
sysctl.h	SysCtl_lockAccessControlRegs
CMRESCTL	
sysctl.c	SysCtl_controlCMReset
sysctl.h	SysCtl_isCMReset
CMTOCPU1NMICTL	
sysctl.h	SysCtl_enableCMtoCPUNMI
sysctl.h	SysCtl_disableCMtoCPUNMI
sysctl.h	SysCtl_getCMtoCPUNMI
CMTOCPU1INTCTL	
sysctl.h	SysCtl_enableCMtoCPUInterrupt
sysctl.h	SysCtl_disableCMtoCPUInterrupt
sysctl.h	SysCtl_getCMtoCPUInterrupt
PALLOCATE0	
sysctl.h	SysCtl_allocateSharedPeripheral

Table 3-413. SYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
CM_CONF_REGS_LOCK	
sysctl.h	SysCtl_lockCMConfig
CM_STATUS_INT_FLG	
sysctl.h	SysCtl_getCMInterruptStatus
CM_STATUS_INT_CLR	
sysctl.h	SysCtl_clearCMInterruptStatus
CM_STATUS_INT_SET	
sysctl.h	SysCtl_setCMInterruptStatus
CM_STATUS_MASK	
sysctl.h	SysCtl_getCMInterruptStatusMask
sysctl.h	SysCtl_setCMInterruptStatusMask
SYS_ERR_INT_FLG	
sysctl.h	SysCtl_getInterruptStatus
SYS_ERR_INT_CLR	
sysctl.h	SysCtl_clearInterruptStatus
SYS_ERR_INT_SET	
sysctl.h	SysCtl_setInterruptStatus
SYS_ERR_MASK	
sysctl.h	SysCtl_getInterruptStatusMask
sysctl.h	SysCtl_setInterruptStatusMask
SYNCSELECT	
sysctl.h	SysCtl_setSyncOutputConfig
ADCSOCOUTSELECT	
sysctl.h	SysCtl_enableExtADCSOCSource
sysctl.h	SysCtl_disableExtADCSOCSource
SYNCSOCLOCK	
sysctl.h	SysCtl_lockExtADCSOCSelect
sysctl.h	SysCtl_lockSyncSelect

3.16.23.8 WWD Registers to Driverlib Functions

Table 3-414. WWD Registers to Driverlib Functions

File	Driverlib Function
SCSR	
sysctl.h	SysCtl_setWatchdogMode
sysctl.h	SysCtl_isWatchdogInterruptActive
sysctl.h	SysCtl_clearWatchdogOverride
WDCNTR	
sysctl.h	SysCtl_getWatchdogCounterValue
WDKEY	
sysctl.h	SysCtl_serviceWatchdog
sysctl.h	SysCtl_enableWatchdogReset
sysctl.h	SysCtl_resetWatchdog
WDCR	
sysctl.h	SysCtl_resetDevice
sysctl.h	SysCtl_disableWatchdog

Table 3-414. WWD Registers to Driverlib Functions (continued)

File	Driverlib Function
sysctl.h	SysCtl_enableWatchdog
sysctl.h	SysCtl_setWatchdogPredivider
sysctl.h	SysCtl_setWatchdogPrescaler
WDWCR	
sysctl.h	SysCtl_setWatchdogWindowValue

3.16.23.9 XINT Registers to Driverlib Functions

Table 3-415. XINT Registers to Driverlib Functions

File	Driverlib Function
1CR	
gpio.c	GPIO_setInterruptPin
gpio.h	GPIO_setInterruptType
gpio.h	GPIO_getInterruptType
gpio.h	GPIO_enableInterrupt
gpio.h	GPIO_disableInterrupt
gpio.h	GPIO_getInterruptCounter
2CR	
-	See 1CR
3CR	
-	See 1CR
4CR	
-	See 1CR
5CR	
-	See 1CR
1CTR	
gpio.h	GPIO_getInterruptCounter
2CTR	
-	
3CTR	
-	

This page intentionally left blank.

Chapter 4

C28x Processor

This chapter contains a short description of the C28x processor and extended instruction sets.

Further information can be found in the following documents:

- *TMS320C28x CPU and Instruction Set Reference Guide*
- *TMS320C28x Extended Instruction Sets Technical Reference Manual*
- *Accelerators: Enhancing the Capabilities of the C2000 MCU Family Technical Brief*
- *TMS320C28x FPU Primer Application Report*

4.1 Introduction

The C28x CPU is a 32-bit fixed-point processor. This device draws from the best features of digital signal processing, reduced instruction set computing (RISC), microcontroller architectures, firmware, and tool sets.

For more information on CPU architecture and instruction set, see the *TMS320C28x CPU and Instruction Set Reference Guide*.

4.2 C28X Related Collateral

Foundational Materials

- C2000 Academy - C28x
- C2000 C28x Optimization Guide
- C2000 Software Guide
- *Enhancing the Computational Performance of the C2000™ Microcontroller Family Application Report*

Getting Started Materials

- C2000 Multicore Development User Guide
- C2000Ware - CLAMath
- C2000Ware - FPU Fast RTS
- C2000Ware - FPU Library
- C2000Ware - Fast Integer Division
- C2000Ware - Fixed Point Library
- C2000Ware - IQMath
- C2000Ware - VCU Library
- C2000Ware Libraries Overview
- *CRC Engines in C2000 Devices Application Report*
- *TMS320C28x Extended Instruction Sets Application Report*
- *TMS320C28x FPU Primer Application Report*

Expert Materials

- *Fast Integer Division - A Differentiated Offering From C2000 Product Family Application Report*

4.3 Features

The CPU features include a modified Harvard architecture and circular addressing. The RISC features are single-cycle instruction execution, register-to-register operations, and modified Harvard architecture. The microcontroller features include ease of use through an intuitive instruction set, byte packing and unpacking, and bit manipulation. The modified Harvard architecture of the CPU enables instruction and data fetches to be performed in parallel. The CPU can read instructions and data while it writes data simultaneously to maintain the single-cycle instruction operation across the pipeline.

4.4 Floating-Point Unit

The C28x plus floating-point (C28x+FPU64) processor extends the capabilities of the C28x fixed-point CPU by adding registers and instructions to support IEEE single-precision and double-precision floating point operations.

Devices with the C28x+FPU64 include the standard C28x register set plus an additional set of floating-point unit registers. The additional floating-point unit registers are the following:

- Eight floating-point result registers, RnH (where n = 0–7)
- Floating-point Status Register (STF)
- Repeat Block Register (RB)

All of the floating-point registers, except the repeat block register, are shadowed. This shadowing can be used in high-priority interrupts for fast context save and restore of the floating-point registers.

For more information, see the *TMS320C28x Extended Instruction Sets Technical Reference Manual*.

4.5 Trigonometric Math Unit (TMU)

The trigonometric math unit (TMU) extends the capabilities of a C28x+FPU64 by adding instructions and leveraging existing FPU64 instructions to speed up the execution of common trigonometric and arithmetic operations listed in Table 4-1.

Table 4-1. TMU Supported Instructions

Instructions	C Equivalent Operation	Pipeline Cycles
MPY2PIF32/64 RaH,RbH	a = b * 2pi	2/3
DIV2PIF32/64 RaH,RbH	a = b / 2pi	2/3
DIVF32/64 RaH,RbH,RcH	a = b/c	5
SQRTF32/64 RaH,RbH	a = sqrt(b)	5
SINPUF32/64 RaH,RbH	a = sin(b*2pi)	4
COSPUF32/64 RaH,RbH	a = cos(b*2pi)	4
ATANPUF32/64 RaH,RbH	a = atan(b)/2pi	4
QUADF32/64 RaH,RbH,RcH,RdH	Operation to assist in calculating ATANPU2	5

No changes have been made to existing instructions, pipeline or memory bus architecture. All TMU instructions use the existing FPU register set (R0H to R7H) to carry out their operations.

For more information, see the *TMS320C28x Extended Instruction Sets Technical Reference Manual*.

4.6 VCRC Unit

Cyclic redundancy check (CRC) algorithms provide a straightforward method for verifying data integrity over large data blocks, communication packets, or code sections. The C28x+VCRC can perform 8-bit, 16-bit, 24-bit, and 32-bit CRCs. A CRC result register contains the current CRC, which is updated whenever a CRC instruction is executed.

The following are the CRC polynomials used by the CRC calculation logic of VCRC:

- CRC8 polynomial = 0x07
- CRC16 polynomial1 = 0x8005
- CRC16 polynomial2 = 0x1021
- CRC24 polynomial = 0x5d6dcb
- CRC32 polynomial1 = 0x04c11db7
- CRC32 polynomial2 = 0x1edc6f41

This module can calculate CRCs for a byte of data in a single cycle. The CRC calculation for CRC8, CRC16, CRC24 and CRC32 is done byte-wise (instead of computing on a complete 16-bit or 32-bit data read by the C28x core) to match the byte-wise computation requirement mandated by various standards.

The VCRC Unit also allows the user to provide the size (1b-32b) and value of any polynomial to fit custom CRC requirements. The CRC execution time increases to 3 cycles when using a custom polynomial.

For more information, see the *TMS320C28x Extended Instruction Sets Technical Reference Manual*.

This page intentionally left blank.

Chapter 5

ROM Code and Peripheral Booting

This chapter describes the boot flow and functionality of the CPU1, CPU2, and Connectivity Manager (CM) subsystems.

5.1 Introduction

The purpose of this chapter is to explain the boot read-only memory (ROM) code functionality for CPU1, CPU2, and CM cores, including the boot procedure. It also discusses the functions and features of the boot ROM code, and provides details about the ROM memory map contents. On every reset, the device executes a boot sequence in the ROM depending on the reset type and boot configuration. This sequence will initialize the device to run the application code. For CPU1, the boot ROM also contains peripheral bootloaders which can be used to load an application into RAM. These bootloaders can be disabled for safety or security purposes.

Refer to Table 5-1 for details on available boot features across CPU1, CPU2, and CM. Additionally, Table 5-2 shows the sizes of the various ROMs on the device.

Table 5-1. Boot System Overview

Boot Feature	CPU1 (Master)	CPU2	CM
Initiate boot process	Device Reset	CPU1 Application	CPU1 Application
Boot mode selection	GPIOs	IPC Register	IPC Register
Supported boot modes: • Flash boot • Secure Flash boot • RAM boot	Yes	Yes	Yes
Boot to User OTP	No	Yes	Yes
Copy from IPC Message RAM and boot to RAM	No	Yes	Yes
Peripheral boot loader support	Yes	No	No

Table 5-2. ROM Memory

ROM	CPU1 Size	CPU2 Size	CM Size
Unsecure boot ROM	192 KB	64 KB	64 KB
Secure ROM	64 KB	64 KB	32 KB
CLA data ROM	8 KB	8 KB	N/A

5.1.1 ROM Related Collateral

Foundational Materials

- Bootloading 101 (Video)

Getting Started Materials

- *Secure BOOT On C2000 Device Application Report*

Expert Materials

- *C2000 Software Controlled Firmware Update Process Application Report*

5.2 Device Boot Sequence

The boot sequence describes the general boot ROM procedure each time a CPU core is reset. CPU1 is the master and always boots first. Once CPU1 boots to the application, then the user's application code in CPU1 can configure CPU2/CM boot IPC registers and release CPU2/CM from reset to boot. Table 5-3, Table 5-4, and Table 5-5 detail the general overview of the boot-up procedures for each core.

During boot, each CPU's boot ROM code updates a boot status location in RAM that details the actions taken during this process. Additionally, CPU2 writes the boot status to the CPU2TOCPU1IPCBOOTSTS register and CM writes to CMTOCPU1IPCBOOTSTS to communicate the statuses to CPU1.

Refer to Section 5.7.11 for more details.

Table 5-3. CPU1 Boot ROM Sequence

Step	CPU1 Action
1	After reset, check for HWBIST reset. If it is a HWBIST reset, immediately branch and return to the user application. If it is not a HWBIST reset, then continue boot and check the FUSE error register for any errors and handle accordingly.
2	Clock configuration and Flash power-up
3	Peripheral trimming and device configuration registers are loaded from OTP.
4	On power-on reset (POR), all CPU1 RAMs are initialized.
5	Non-maskable interrupt (NMI) handling is enabled and DCSM initialization is performed.
6	Device calibration is performed; trimming the specified peripherals with set OTP values.
7	Determine if polling the GPIO pins are needed for determining the boot mode and, if so, read the boot mode GPIO pins to determine the boot mode to run.
8	Based on the boot mode and options, the appropriate boot sequence is executed. Refer to Section 5.5.1 for a flow chart of the CPU1 boot sequences.

Table 5-4. CPU2 Boot ROM Sequence

Step	CPU2 Action
1	CPU2 is released from reset by CPU1 application.
2	Once CPU1TOCPU2IPCFLG0 is set, read the CPU1TOCPU2IPCBOOTMODE register. If it is not set correctly or has an invalid value, the IPC error command is sent to CPU1 and the CPU2 core will enter an infinite loop and will not continue booting until the user corrects the register values and reset the CPU2.
3	Flash power-up
4	On POR, all CPU2 RAMs are initialized.
5	NMI handling is enabled.
6	Based on the boot mode set in CPU1TOCPU2IPCBOOTMODE register, CPU2 either enters "wait for command" mode to wait for a future CPU1 boot mode command or CPU2 executes the requested boot sequence. Refer to Section 5.5.2 for a flow chart of the CPU2 boot sequences.

Table 5-5. CM Boot ROM Sequence

Step	CM Action
1	CM is released from reset by the CPU1 application.
2	Once CPU1TOCMIPCFLG0 is set, read the CPU1TOCMIPCBOOTMODE register.If it is not set correctly or has an invalid value, the IPC error command is sent to CPU1 and the CM will enter an infinite loop and will not continue booting until the user corrects the register values and reset the CM.
3	Flash power-up
4	On POR, all CM RAMs are initialized.
5	NMI handling is enabled.
6	Based on the boot mode set in CPU1TOCPU2IPCBOOTMODE register, CM either enters "wait for command" mode to wait for a future CPU1 boot mode command or CM executes the requested boot sequence. Refer to Section 5.5.3 for a flow chart of the CM boot sequences.

5.3 Device Boot Modes

This section explains the default boot modes, as well as all the available boot modes supported on this device. The CPU1 boot ROM uses the boot mode select, general purpose input/output (GPIO) pins to determine the boot mode configuration. CPU2 boot ROM uses the CPU1TOCPU2IPCBOOTMODE register to determine the boot mode configuration and CM boot ROM uses the CPU1TOCMIPCBOOTMODE register to determine the boot mode configuration.

Table 5-6 shows the CPU1 boot mode options available for selection by the default boot mode select pins. Users have the option to program the device to customize the boot modes selectable in the boot-up table as well as the boot mode select pin GPIOs used.

All the available boot modes on the device are described in Table 5-8.

Table 5-6. Device Default Boot Modes for CPU1

Boot Mode	GPIO72 (Default boot mode select pin 1)	GPIO84 (Default boot mode select pin 0)
Parallel IO	0	0
SCI / Wait Boot(1)	0	1
CAN	1	0
Flash / USB(2)	1	1

(1) SCI boot mode can be used as a wait boot mode as long as SCI continues to wait for an 'A' or 'a' during the SCI autobaud lock process.
(2) On an unprogrammed device, selecting Flash boot when the default Flash entry address is unprogrammed switches the boot mode from Flash boot to USB boot. See Table 5-7 for more details.

Table 5-7. CPU1 Flash-to-USB Boot Decision Table

Value at Flash Entry Point Address	Reason for Value	Realized Boot Mode
0x00000000	Flash is locked/secured	Boot to Flash
0xFFFFFFFF	Flash is not programmed	USB Boot
Any other value	Flash is programmed	Boot to Flash

Note

The switch of Flash boot mode to USB boot mode when Flash is not programmed is only available as part of the default boot mode table on an unprogrammed device. Once a custom boot table is programmed in OTP or RAM, a selection of Flash boot mode does not switch to USB boot even when Flash is unprogrammed.

Table 5-8. All Available Boot Modes

Boot Mode	CPU Support	Details
Parallel IO	CPU1	
SCI / Wait	CPU1	
CAN	CPU1	
Flash	CPU1, CPU2, CM	
Wait	CPU1, CPU2, CM	Refer to Section 5.7.7 for functional details of the boot modes.
RAM	CPU1, CPU2, CM	
SPI	CPU1	Refer to Section 5.7.8 for boot table values and GPIOs for the boot modes.
I2C	CPU1	
USB	CPU1	
Secure Flash	CPU1, CPU2, CM	
User OTP	CPU2, CM	
IPC Message Copy to RAM	CPU2, CM	

Note

All the peripheral boot modes that are supported use the first instance of the peripheral module (SCIA, SPIA, I2CA, CANA, and so forth). Whenever these boot modes are referred to in this chapter, such as SCI boot, it is actually referring to the first module instance, which means the SCI boot on the SCIA port. The same applies to the other peripheral boots.

5.4 Device Boot Configurations

This section details what boot configurations are available and how to configure them. This device supports from 0 boot mode select pins up to 3 boot mode select pins as well as from 1 configured boot mode up to 8 configured boot modes.

To change and configure the device from the default settings to custom settings for your application, use the following process:

1. Determine all the various ways you want application to be able to boot. (For example: Primary boot option of Flash boot for your main application, secondary boot option of CAN boot for firmware updates, tertiary boot option of SCI boot for debugging, etc)
2. Based on the number of boot modes needed, determine how many boot mode select pins (BMSPs) are required to select between your selected boot modes. (For example: 2 BMSPs are required to select between 3 boot mode options)
3. Assign the required BMSPs to a physical GPIO pin. (For example, BMSP0 to GPIO50, BMSP1 to GPIO51, and BMSP2 left as default which is disabled). Refer to Section 5.4.1 for all the details on performing these configurations.
4. Assign the determined boot mode definitions to indexes in your custom boot table that correlate to the decoded value of the BMSPs. For example, BOOTDEF0=Boot to Flash, BOOTDEF1=CAN Boot, BOOTDEF2=SCI Boot; all other BOOTDEFx are left as default/nothing). Refer to Section 5.4.2 for all the details on setting up and configuring the custom boot mode table.

Additionally, Section 5.4.3 provides some example use cases on how to configure the BMSPs and custom boot tables.

5.4.1 Configuring Boot Mode Pins for CPU1

This section explains how the boot mode select pins can be customized by the user, by programming the BOOTPINCONFIG location (refer to Table 5-9) in the user-configurable dual-zone security module (DCSM) OTP. The location in the DCSM OTP is Z1-BOOTPINCONFIG or Z2-BOOTPINCONFIG. When debugging, EMUBOOTPINCONFIG is the emulation equivalent of Z1-BOOTPINCONFIG/Z2-BOOTPINCONFIG, and can be programmed to experiment with different boot modes without writing to OTP. The device can be programmed to use 0, 1, 2, or 3 boot mode select pins as needed.

Note

When using Z2-BOOTPINCONFIG, the configurations programmed in this location will take priority over the configurations in Z1-BOOTPINCONFIG. It is **recommended** to use Z1-BOOTPINCONFIG first and then if OTP configurations need to be altered, switch to using Z2-BOOTPINCONFIG.

Table 5-9. CPU1 BOOTPINCONFIG Bit Fields

Bit	Name	Description
31:24	Key	Write 0x5A to these 8-bits to tell the boot ROM code that the bits in this register are valid For EMUBOOTPINCONFIG only, write 0xA5 to emulate the standalone boot flow and use the BMSPs/OTP BOOTDEF table
23:16	Boot Mode Select Pin 2 (BMSP2)	Refer to BMSP0 description except for BMSP2
15:8	Boot Mode Select Pin 1 (BMSP1)	Refer to BMSP0 description except for BMSP1
7:0	Boot Mode Select Pin 0 (BMSP0)	Set to the GPIO pin to be used during boot (up to 255). 0x0 = GPIO0; 0x01 = GPIO1 and so on Writing 0xFF disables BMSP0 and this pin is no longer used to select the boot mode.

Note

The following GPIOs **cannot** be used as a BMSP. If selected for a particular BMSP, the boot ROM automatically selects the factory default GPIO (the factory default for BMSP2 is 0xFF, which disables the BMSP).

- GPIO 42 and GPIO 43
- GPIO 169 to 255

Table 5-10. CPU1 Standalone Boot Mode Select Pin Decoding

BOOTPIN_CONFIG Key	BMSP0	BMSP1	BMSP2	Realized Boot Mode
!= 0x5A	Don't Care	Don't Care	Don't Care	Boot as defined by the factory default BMSPs
= 0x5A	0xFF	0xFF	0xFF	Boot as defined in the boot table for boot mode 0 (All BMSPs disabled)
	Valid GPIO	0xFF	0xFF	Boot as defined by the value of BMSP0 (BMSP1 and BMSP2 disabled)
	0xFF	Valid GPIO	0xFF	Boot as defined by the value of BMSP1 (BMSP0 and BMSP2 disabled)
	0xFF	0xFF	Valid GPIO	Boot as defined by the value of BMSP2 (BMSP0 and BMSP1 disabled)
	Valid GPIO	Valid GPIO	0xFF	Boot as defined by the values of BMSP0 and BMSP1 (BMSP2 disabled)
	Valid GPIO	0xFF	Valid GPIO	Boot as defined by the values of BMSP0 and BMSP2 (BMSP1 disabled)
	0xFF	Valid GPIO	Valid GPIO	Boot as defined by the values of BMSP1 and BMSP2 (BMSP0 disabled)
	Valid GPIO	Valid GPIO	Valid GPIO	Boot as defined by the values of BMSP0, BMSP1, and BMSP2
	Invalid GPIO	Valid GPIO	Valid GPIO	BMSP0 is reset to the factory default BMSP0 GPIO Boot as defined by the values of BMSP0, BMSP1, and BMSP2
	Valid GPIO	Invalid GPIO	Valid GPIO	BMSP1 is reset to the factory default BMSP1 GPIO Boot as defined by the values of BMSP0, BMSP1, and BMSP2
	Valid GPIO	Valid GPIO	Invalid GPIO	BMSP2 is reset to the factory default state, which is disabled Boot as defined by the values of BMSP0 and BMSP1

Note

When decoding the boot mode, BMSP0 is the least-significant-bit and BMSP2 is the most-significant-bit of the boot table index value. It is **recommended** when disabling BMSPs to start with disabling BMSP2. For example, in an instance when only using BMSP2 (BMSP1 and BMSP0 are disabled), then only the boot table indexes of 0 and 4 will be selectable. In the instance when using only BMSP0, then the selectable boot table indexes are 0 and 1.

5.4.2 Configuring Boot Mode Table Options for CPU1

This section explains how to configure the boot definition table, BOOTDEF, for CPU1. The 64-bit location is located in user-configurable DCSM OTP in the Z1-BOOTDEF-LOW and Z1-BOOTDEF-HIGH locations. When debugging, EMUBOOTDEF-LOW and EMUBOOTDEF-HIGH are the emulation equivalents of Z1-BOOTDEF-LOW and Z1-BOOTDEF-HIGH, and can be programmed to experiment with different boot mode options without writing to OTP. The range of customization to the boot definition table depends on how many boot mode select pins (BMSP) are being used. For example, 0 BMSPs equals to 1 table entry, 1 BMSP equals to 2 table entries, 2 BMSPs equals to 4 table entries, and 3 BMSPs equals to 8 table entries. Refer to Section 5.4.3 for examples on how to setup the BOOTPIN_CONFIG and BOOTDEF values.

Note

The locations Z2-BOOTDEF-LOW and Z2-BOOTDEF-HIGH will be used instead of Z1-BOOTDEF-LOW and Z1-BOOTDEF-HIGH locations when Z2-BOOTPINCONFIG is configured. Refer to Section 5.4.1 for more details on BOOTPIN_CONFIG usage.

Table 5-11. CPU1 BOOTDEF Bit Fields

BOOTDEF Name	Byte Position	Name	Description
BOOT_DEF0	7:0	BOOT_DEF0 Mode/Options	Set the boot mode for index 0 of the boot table. Different boot modes and their options can include, for example, a boot mode that uses different GPIOs for a specific bootloader or a different Flash entry point address. Any unsupported boot mode will cause the device to either go to wait boot or boot to Flash. Refer to Section 5.7.8 for valid BOOTDEF values to set in the table.
BOOT_DEF1	15:8	BOOT_DEF1 Mode/Options	Refer to BOOT_DEF0 description
BOOT_DEF2	23:16	BOOT_DEF2 Mode/Options	
BOOT_DEF3	31:24	BOOT_DEF3 Mode/Options	
BOOT_DEF4	39:32	BOOT_DEF4 Mode/Options	
BOOT_DEF5	47:40	BOOT_DEF5 Mode/Options	
BOOT_DEF6	55:48	BOOT_DEF6 Mode/Options	
BOOT_DEF7	63:56	BOOT_DEF7 Mode/Options	

5.4.3 Boot Mode Example Use Cases

This section demonstrates some use cases for configuring the boot mode select pins and boot modes.

5.4.3.1 Zero Boot Mode Select Pins

This use case demonstrates a scenario for an application that does not use any boot mode select pins and always has the device boot to Flash.

1. Program the BOOTPIN_CONFIG location in OTP as follows:
   - Set BOOTPIN_CONFIG.BMSP0 to 0xFF
   - Set BOOTPIN_CONFIG.BMSP1 to 0xFF
   - Set BOOTPIN_CONFIG.BMSP2 to 0xFF
   - Set BOOTPIN_CONFIG.KEY to 0x5A for boot ROM to treat these register bits as valid and use the custom boot table.
2. Program the BOOTDEF location options for the device. This essentially sets up a device-specific boot mode table. Refer to Section 5.7.8 for valid BOOTDEF values to set in the table.
   - Set BOOTDEF.BOOTDEF0 to 0x03 for booting to Flash (entry address option 0). This sets Flash boot to boot table index 0.
   - Refer to Section 5.7.3 for the available Flash entry points.

Table 5-12. Zero Boot Pin Boot Table Result

Boot Mode Table Number	Boot Mode
0	Flash Boot (0x03)

5.4.3.2 One Boot Mode Select Pin

This use case demonstrates a scenario for an application using one boot mode select pin to select between booting to Flash or using CAN boot.

1. Program the BOOTPIN_CONFIG location in OTP as follows:
   - Set BOOTPIN_CONFIG.BMSP0 to a user specified GPIO, such as 0x0 for GPIO0
   - Set BOOTPIN_CONFIG.BMSP1 to 0xFF
   - Set BOOTPIN_CONFIG.BMSP2 to 0xFF
   - Set BOOTPIN_CONFIG.KEY to 0x5A for boot ROM to treat these register bits as valid and use the custom boot table.
2. Program the BOOTDEF location options for the device. This essentially sets up a device-specific boot mode table. Refer to Section 5.7.8 for valid BOOTDEF values to set in the table.
   - Set BOOTDEF.BOOTDEF0 to 0x02 for CAN booting. This sets CAN boot to boot table index 0.
   - Set BOOTDEF.BOOTDEF1 to 0x03 for booting to Flash (entry address option 0). This sets Flash boot to boot table index 1.

Table 5-13. One Boot Pin Boot Table Result

Boot Mode Table Number	Boot Mode
0	CAN Boot (0x02)
1	Flash Boot (0x03)

5.4.3.3 Three Boot Mode Select Pins

This use case demonstrates a scenario for an application using three boot mode select pins to select between various boot modes in the custom boot table.

1. Program the BOOTPIN_CONFIG location in OTP as follows:
   - Set BOOTPIN_CONFIG.BMSP0 to a user specified GPIO, such as 0x0 for GPIO0
   - Set BOOTPIN_CONFIG.BMSP1 to a user specified GPIO, such as 0x1 for GPIO1
   - Set BOOTPIN_CONFIG.BMSP2 to a user specified GPIO, such as 0x2 for GPIO2
   - Set BOOTPIN_CONFIG.KEY to 0x5A for boot ROM to treat these register bits as valid and use the custom boot table.
2. Program the BOOTDEF location options for the device. This essentially sets up a device-specific boot mode table. Refer to Section 5.7.8 for valid BOOTDEF values to set in the table.
   - Set BOOTDEF.BOOTDEF0 to 0x02 for CAN booting. This sets CAN boot to boot table index 0.
   - Set BOOTDEF.BOOTDEF1 to 0x03 for booting to Flash (entry address option 0). This sets Flash boot to boot table index 1.
   - Set BOOTDEF.BOOTDEF2 to 0x24 for booting to wait boot (alternate option). This sets wait boot to boot table index 2.
   - Set BOOTDEF.BOOTDEF3 to 0x66 for SPI booting (alternate GPIO option 3). This sets SPI boot to boot table index 3.
   - Set BOOTDEF.BOOTDEF4 to 0x43 for booting to Flash (entry address option 2). This sets Flash boot to boot table index 4.
   - Set BOOTDEF.BOOTDEF5 to 0x09 for USB booting. This sets USB boot to boot table index 5.

Table 5-14. Three Boot Pins Boot Table Result

Boot Mode Table Number	Boot Mode
0	CAN Boot (0x02)
1	Flash Boot (0x03)
2	Wait Boot - Alt (0x24)
3	SPI - Alt3 (0x66)
4	Flash Boot - Alt2 (0x43)
5	USB Boot (0x09)

5.5 Device Boot Flow Diagrams

This section details the boot flow diagrams for CPU1, CPU2, and CM.

5.5.1 CPU1 Boot Flow

Upon reset, CPU1 follows the boot flow shown in Figure 5-1. Depending on whether a JTAG debugger is connected to the device, CPU1 will either continue booting following the emulation boot flow or standalone boot flow. Figure 5-2 shows the emulation boot flow when JTAG debugger is connected. Figure 5-3 shows the standalone boot flow when no JTAG debugger is connected to the device.

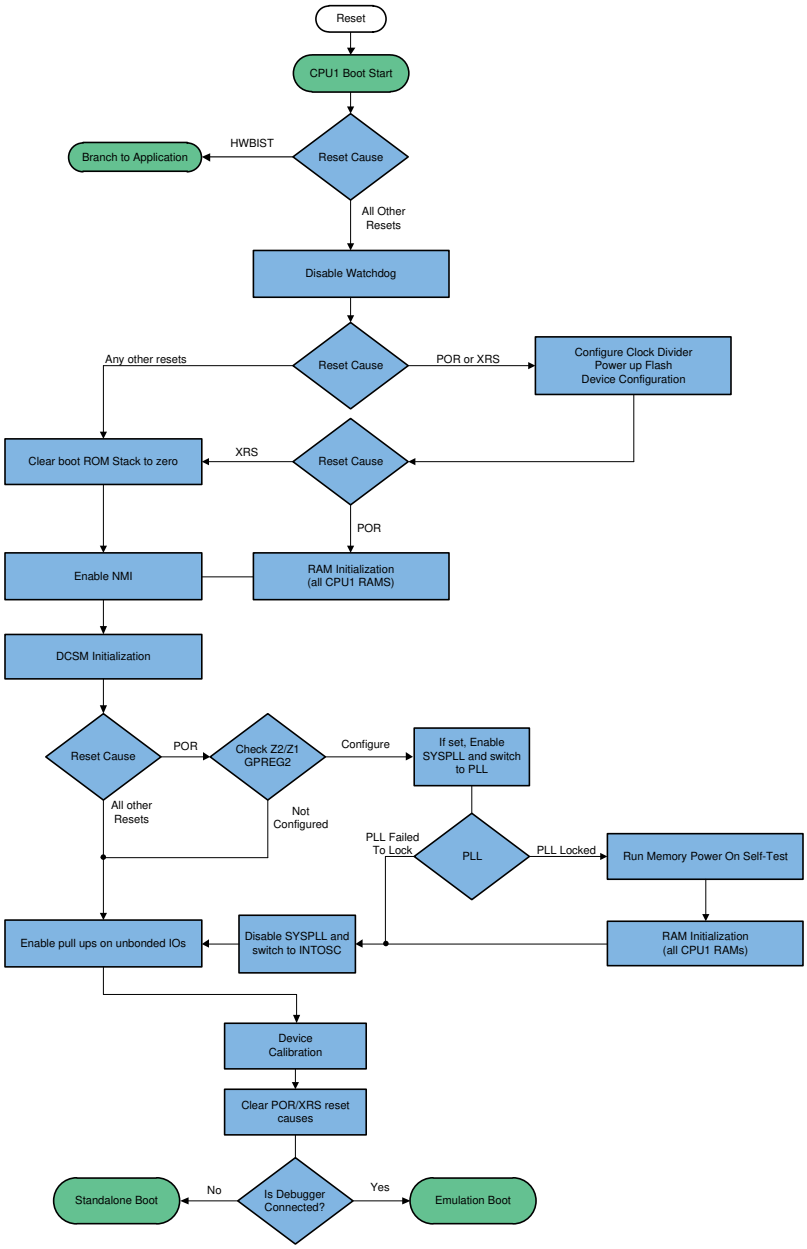


Figure 5-1. CPU1 Device Boot Flow

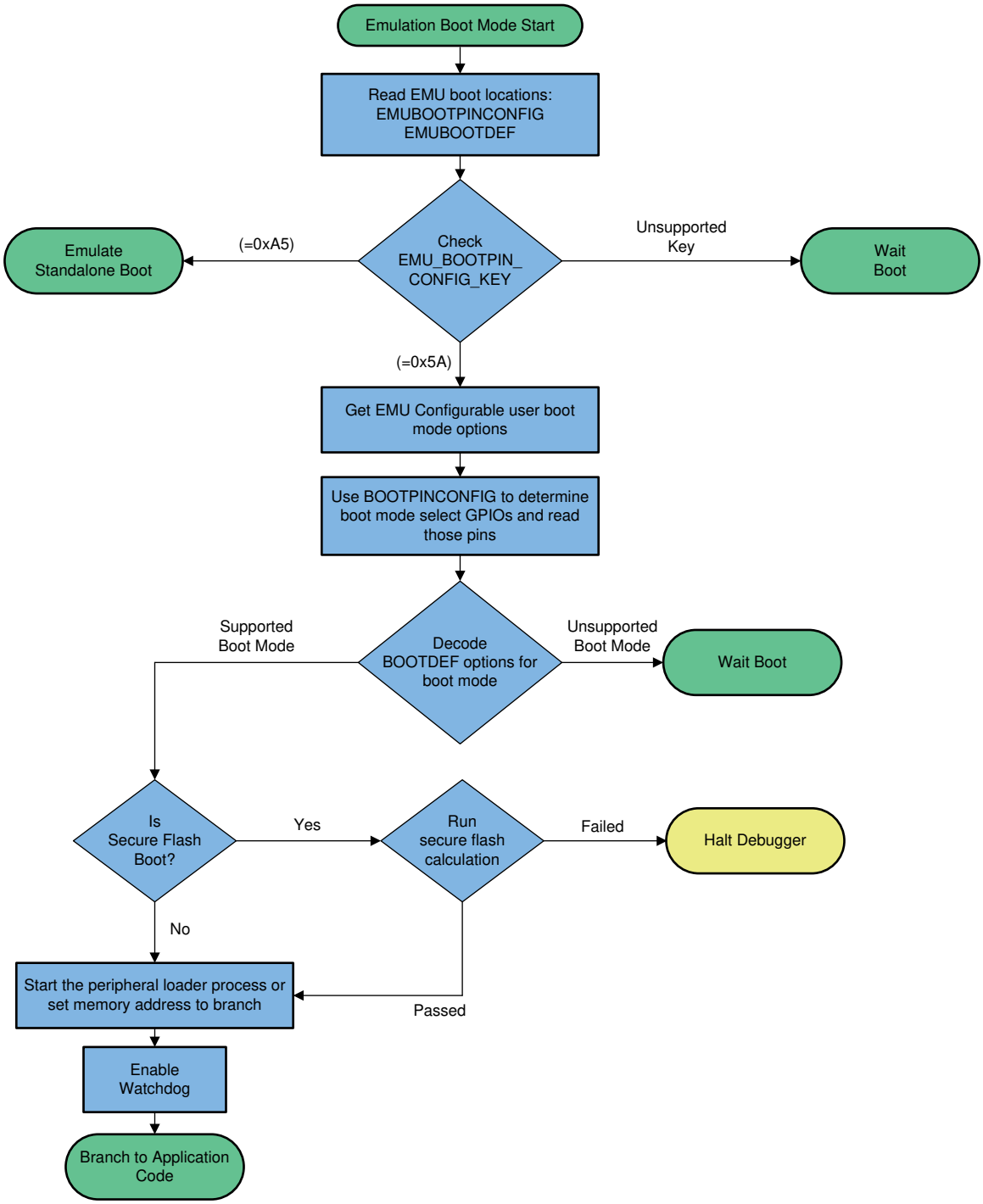


Figure 5-2. CPU1 Emulation Boot Flow

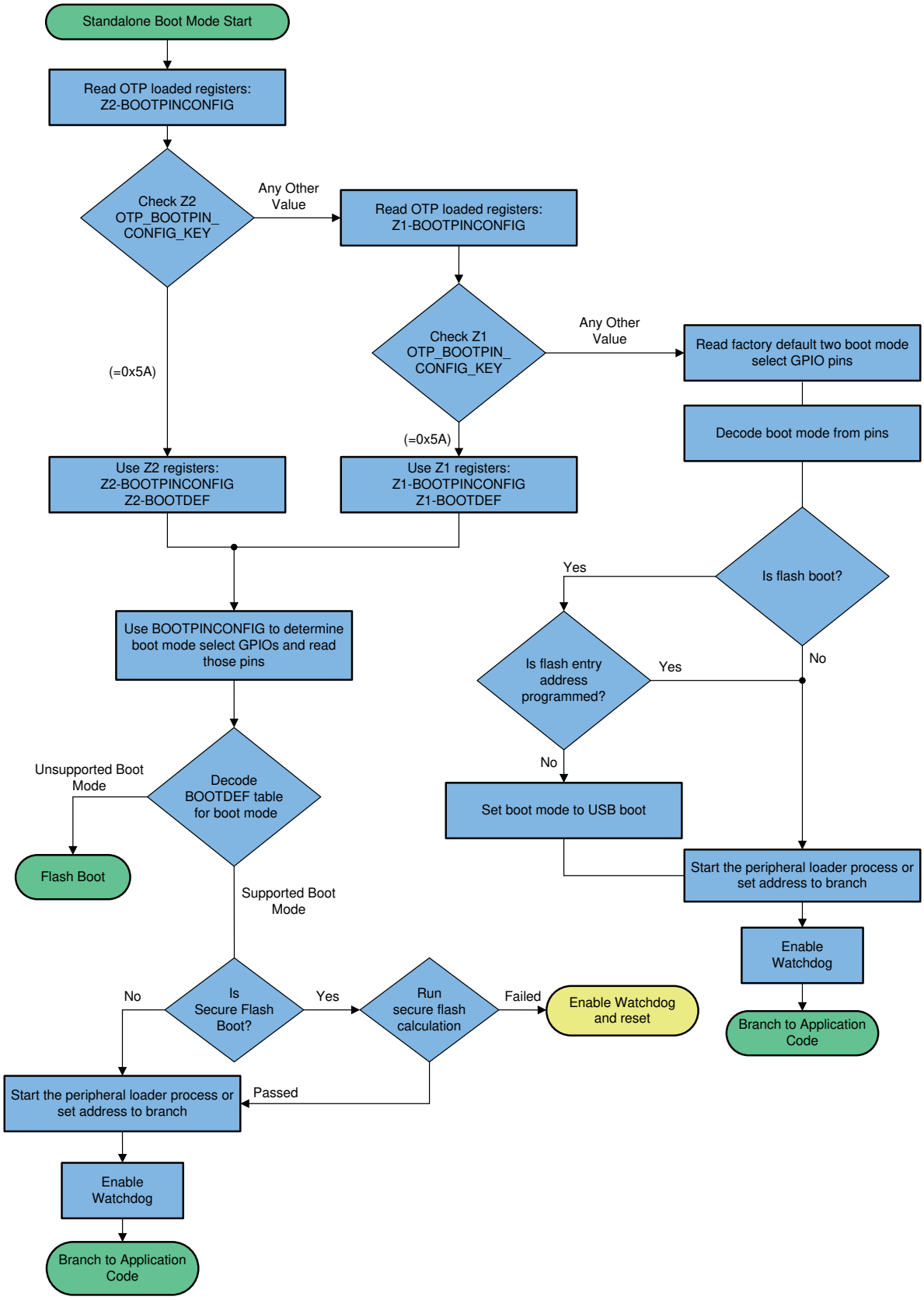


Figure 5-3. CPU1 Standalone Boot Flow

5.5.2 CPU2 Boot Flow

Upon reset, CPU2 follows the boot flow shown in Figure 5-4. CPU2 only boots once configured and released from reset by CPU1. Refer to Section 5.7.2 for more details.

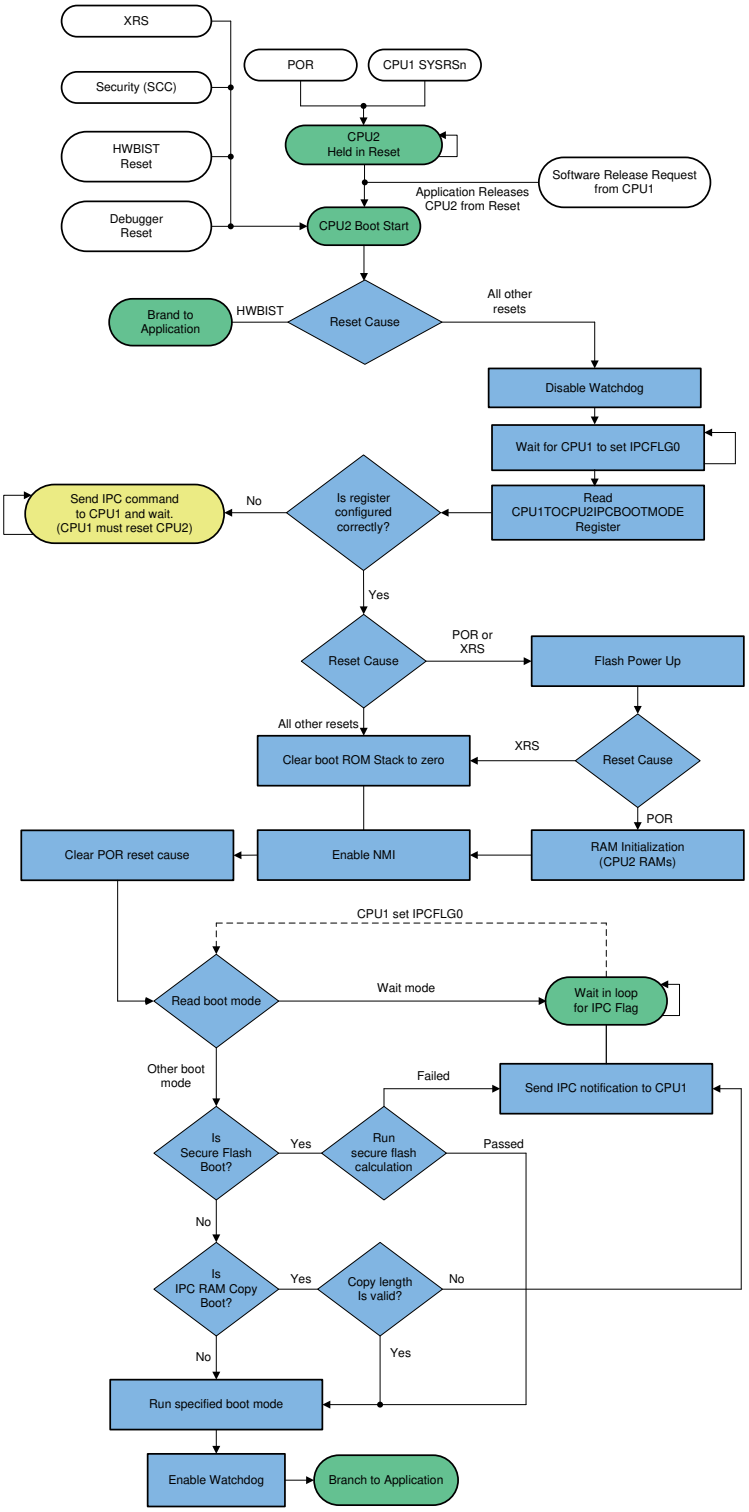


Figure 5-4. CPU2 Boot Flow

5.5.3 Connectivity Manager (CM) Boot Flow

Upon reset, CM follows the boot flow shown in Figure 5-5. CM only boots once configured and released from reset by CPU1. Refer to Section 5.7.2 for more details.

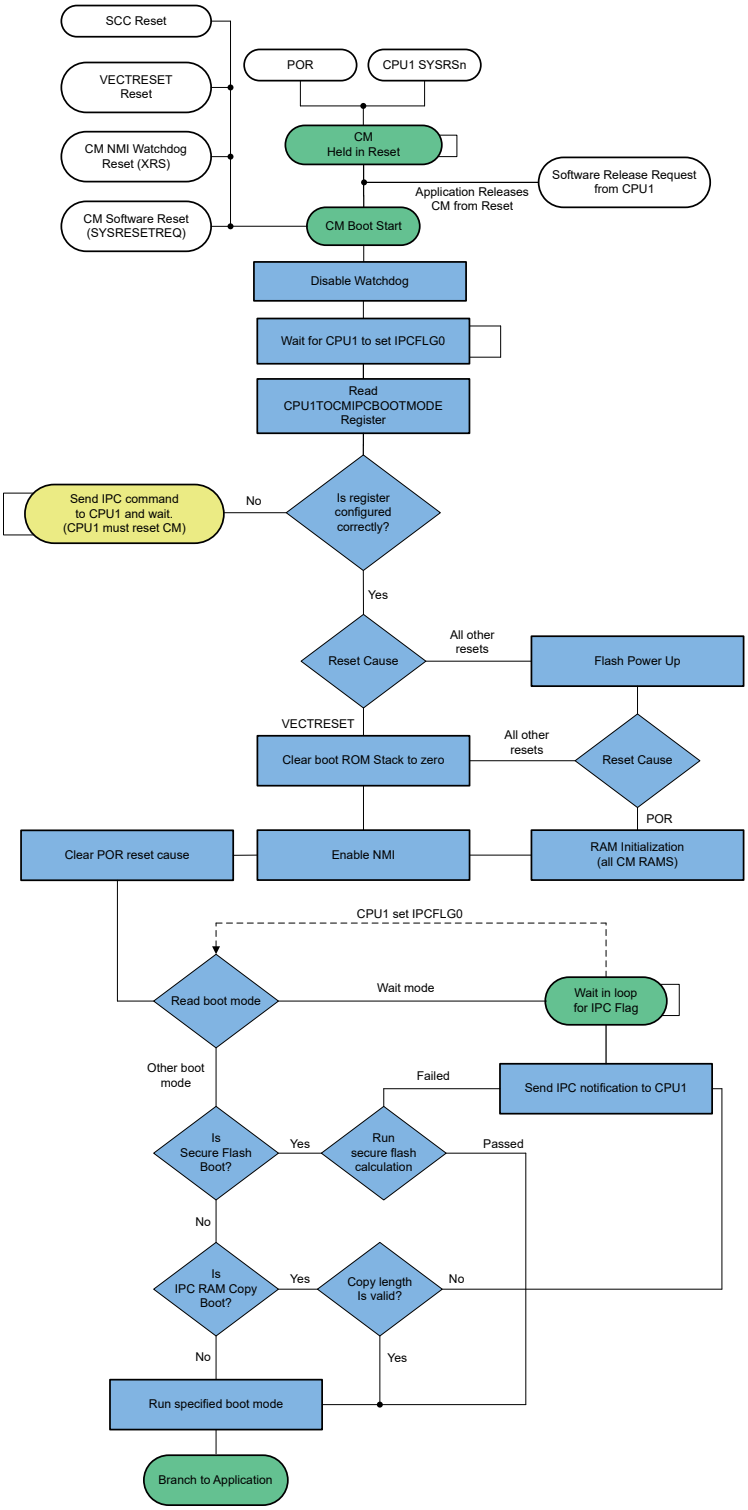


Figure 5-5. CM Boot Flow

5.6 Device Reset and Exception Handling

5.6.1 Reset Causes and Handling

Table 5-15 explains the actions each boot ROM performs upon reset for a specific reset cause.

Table 5-15. Boot ROM Reset Causes and Actions

Reset Source	CPU1 Boot ROM Action	CPU2 Boot ROM Action	CM Boot ROM Action
Power on Reset (POR)	1. Configure Clock Divider 2. Flash Power Up 3. Device configuration and trimming 4. RAM Initialization 5. Continue default boot flow	1. Flash Power Up 2. RAM Initialization 3. Continue default boot flow	1. Flash Power Up 2. RAM Initialization 3. Continue default boot flow
External Reset (XRS) For CPU1, XRS triggers include: • Watchdog Reset • NMI Watchdog Reset • EtherCAT Reset • SIMRESET XRS	1. Configure Clock Divider 2. Flash Power Up 3. Device configuration and trimming 4. Clear RAM for boot stack 5. Continue default boot flow	1. Flash Power Up 2. Clear RAM for boot stack 3. Continue default boot flow	1. Flash Power Up 2. Clear RAM for boot stack 3. Continue default boot flow
Hardware Built-In Self Test (HWBIST)	1. Read HWBIST return address 2. If set, branch to address 3. If not set, continue boot following "Debugger Reset" boot flow actions	1. Read HWBIST return address 2. If set, branch to address 3. If not set, continue boot following "Debugger Reset" boot flow actions	Not Applicable
Secure Copy Code (SCC) Reset (CPU1, CPU2) Watchdog Reset (CPU2) NMI Watchdog Reset (CPU2) Execute Override Logic (EOL) Reset (CM)	1. Clear RAM for boot stack 2. Continue default boot flow	1. Clear RAM for boot stack 2. Continue default boot flow	1. Flash Power Up 2. Clear RAM for boot stack 3. Continue default boot flow
SIMRESET_CPU1	1. Clear RAM for boot stack 2. Continue default boot flow	Not Applicable	Not Applicable
Debugger Reset (CPU1, CPU2) VECTRESET (CM)	1. Clear RAM for boot stack 2. Continue default boot flow	1. Clear RAM for boot stack 2. Continue default boot flow	1. Clear RAM for boot stack 2. Continue default boot flow

5.6.2 Exceptions and Interrupts Handling

Table 5-16 explains the actions each boot ROM performs if any exceptions that can occur happen during boot. The exception handling philosophy for CPU1, in most cases, is to log the error and continue booting to reach the application. The exception handling philosophy for CPU2 and CM is to log the error, notify CPU1, and let CPU1 handle the action.

For any CPU2 NMI event sources, CPU2 NMI handler will clear the NMI flag to stop the NMI watchdog counter and prevent CPU2 from resetting. The error pin will go low or high (depending on the polarity configuration on the error pin and the reset type) temporarily before the NMI flag is cleared. Therefore, a shorter pulse width of the error pin signal means CPU2 (or CM) is the source of the error. Following this, CPU2 sends an error IPC message to CPU1 in order for CPU1 to handle the error and reset CPU2.

For any CM NMI event sources, the CM NMI handler will clear the NMI flag to stop the NMI watchdog counter and prevent CM from resetting. The error pin will go low or high (depending on the polarity configuration on the error pin and the reset type) temporarily before the NMI flag is cleared. Therefore, a shorter pulse width of the error pin signal means CM (or CPU2) is the source of the error. Following this, the CM sends an error IPC message to CPU1 in order for CPU1 to handle the error and reset CM.

Table 5-16. Boot ROM Exceptions and Actions

Exception Event Source	CPU1 Boot ROM Action	CPU2 Boot ROM Action	CM Boot ROM Action	Event Logged
Single-bit error in FUSEERR	Ignore and continue to boot	No action	No action	No
Multi-bit error in FUSEERR	Reset the device	No action	No action	No
Clock Fail	Clear the NMI flag and continue to boot	Clear NMI flag, update boot status to CPU1, send error IPC to CPU1, and wait in loop	Clear NMI flag, update boot status to CPU1, send error IPC to CPU1, and wait in loop	Yes
RAM Uncorrectable Error ROM Parity Error	Perform RAM initialization and reset the device	Clear NMI flag, update boot status to CPU1, send error IPC to CPU1, and wait in loop	Clear NMI flag, update boot status to CPU1, send error IPC to CPU1, and wait in loop	Yes(1)
Flash Uncorrectable Error	Reset the device	Clear NMI flag, update boot status to CPU1, send error IPC to CPU1, and wait in loop	Clear NMI flag, update boot status to CPU1, send error IPC to CPU1, and wait in loop	Yes
HWBIST Error	Clear the NMI flag and continue to boot	Clear NMI flag, update boot status to CPU1, send error IPC to CPU1, and wait in loop	No action	Yes
PIE Vector Mismatch(2)	Fetch error handler address, if address is configured, call handler, else reset the device	Fetch error handler address, if address is configured, call handler, else clear NMI flag, update boot status to CPU1, send error IPC to CPU1, and wait in loop	No action	Yes
Embedded Real-time Analysis and Diagnostic (ERAD) NMI	Clear the NMI flag and continue to boot	Clear NMI flag, update boot status to CPU1, send error IPC to CPU1, and wait in loop	No action	Yes
MCAN Uncorrectable Error	Clear the NMI flag and continue to boot	No action	Clear NMI flag, update boot status to CPU1, send error IPC to CPU1, and wait in loop	Yes
EtherCAT NMI	Clear the NMI flag and continue to boot	No action	Clear NMI flag, update boot status to CPU1, send error IPC to CPU1, and wait in loop	Yes
ITRAP Exception	Record memory address of where the illegal instruction was executed and let device reset	Send IPC to CPU1 with memory address of the illegal instruction and wait in loop	No action	Yes
Invalid IPCBOOTMODE Value upon Reset	No action	Send IPC to CPU1 and wait in loop	Send IPC to CPU1 and wait in loop	Yes
Secure Flash Boot Failure	Enable watchdog and let device reset	Send IPC to CPU1 and return to wait boot mode	Send IPC to CPU1 and return to wait boot mode	Yes
Hard Fault Exception	No action	No action	Update boot status to CPU1, send error IPC to CPU1, and wait in loop	Yes
Any Unsupported Interrupt	Ignore and continue to boot	Ignore and continue to boot	Update boot status to CPU1, send error IPC to CPU1 with the interrupt exception number, and wait in loop	Yes (CM Only)

(1) For CPU1, a RAM uncorrectable error or ROM parity error will clear the boot status information stored in RAM because a RAM initialization is performed to attempt to correct the error. Since the boot status information is erased, this exception can be identified in that a NMIWD reset occurred and all the RAMs are erased.
(2) A PIE vector mismatch in one core (such as CPU1 or CPU2) will trigger the PIE vector mismatch interrupt in other cores.

5.7 Boot ROM Description

This section explains the details regarding the device boot ROMs.

5.7.1 CPU1 Boot ROM Configuration Registers

CPU1 boot ROM code involves several memory addresses and registers used during execution. There are two sets of configurations; one for emulation and one for standalone boot flow. The emulation locations located in RAM emulate the OTP configurations and can be written to as many times as needed. The user configurable DCSM OTP locations used in the standalone boot flow program the device OTP and hence can only be written once. For bit field configuration details for BOOTPIN-CONFIG and BOOTDEF, see Section 5.4.1 and Section 5.4.2.

Additionally, the CPU1 boot ROM supports boot configurations from DCSM zone 1 and zone 2 registers. Zone 2 configurations will supercede zone 1 configurations, so it is recommended to use zone 1 configurations and use zone 2 as a secondary option.

Note

All boot configurations for CPU2 and CM are set through CPU1TOCPU2IPCBOOTMODE and CPU1TOCMIPCBOOTMODE registers. See more details in Section 5.7.2.

Table 5-17. CPU1 Boot ROM Registers

Boot Flow	Register Name	Boot ROM Name	Register Address	User OTP Address
Emulation	-	EMUBOOTPINCONFIG	0x0000 0D00	-
	-	EMUBOOTDEF-LOW	0x0000 0D04	-
	-	EMUBOOTDEF-HIGH	0x0000 0D06	-
Standalone (Using Z1)	Z1-GPREG1	Z1-BOOTPINCONFIG	0x0005 F008	0x0007 8008
	Z1-GPREG2	Z1-BOOT-GPREG2	0x0005 F00A	0x0007 800A
	Z1-GPREG3	Z1-BOOTDEF-LOW	0x0005 F00C	0x0007 800C
	Z1-GPREG4	Z1-BOOTDEF-HIGH	0x0005 F00E	0x0007 800E
Standalone (Using Z2)	Z2-GPREG1	Z2-BOOTPINCONFIG	0x0005 F088	0x0007 8208
	Z2-GPREG2	Z2-BOOT-GPREG2	0x0005 F08A	0x0007 820A
	Z2-GPREG3	Z2-BOOTDEF-LOW	0x0005 F08C	0x0007 820C
	Z2-GPREG4	Z2-BOOTDEF-HIGH	0x0005 F08E	0x0007 820E

5.7.1.1 GPREG2 Usage and MPOST Configuration

Table 5-18 explains how the bit field values from the user configurable DCSM OTP location, Z1-BOOT-GPREG2 or Z2-BOOT-GPREG2, are decoded by CPU1 boot ROM. For more information, see *C2000™ Memory Power-On Self-Test (M-POST)* .

Table 5-18. CPU1 DCSM Z1/Z2 GPREG2 Bit Fields

Bit	Name	Description	Boot ROM Action
31:24	Key	Write 0x5A to these 8 bits to indicate to the boot ROM code that the bits in this register are valid	If set to 0x5A, boot ROM uses the values in this register. If set to any other value, boot ROM ignores values in this register.
23:8	Reserved	Reserved	No Action
7:4	Run MPOST(1) (2)	When configured to a valid value, MPOST (Memory Power on Self-Test) is run on all device memories	0x0 = MPOST is run with PLL enabled for high speed (110 MHz). 0xC = MPOST is run with PLL enabled for medium speed (80 MHz). 0x3 = MPOST is run with PLL enabled for low speed (60 MHz). 0x9 = MPOST is run using INTOSC2 with PLL disabled (10 MHz). Any other value = MPOST does not run.
3:0	Reserved	Reserved	No Action

(1) If MPOST is configured to run with PLL enabled and the PLL fails to lock, then the MPOST run is skipped. This does not apply if MPOST is configured to use INTOSC2 with PLL disabled.

(2) Note that EtherCAT gets de-asserted and released from reset using the EtherCATs peripheral software reset register during MPOST. The EtherCAT state is not restored back into the reset state.

5.7.2 Booting CPU2 and CM

This section details the boot up flow for CPU2 and CM. This includes the process, how to configure IPCBOOTMODE, and the error IPC commands that CPU2/CM report to CPU1.

5.7.2.1 Boot Up Procedure

The boot configurations for CPU2 and CM are set by the CPU1 application. The CPU1 application configures the clocks for CPU2/CM, sets the boot mode and other parameters in the IPCBOOTMODE register, and releases CPU2/CM from reset to boot.

CPU2 and CM have two states where CPU1 can configure their boot mode. The first state occurs before CPU2/CM boot and when they are still in reset. The second state occurs after CPU2/CM have been released from reset to wait boot mode where the cores wait for an IPC flag to be set by CPU1 to indicate that a boot mode has been set in the IPCBOOTMODE register. The procedures that CPU1 must follow are detailed in Table 5-19 and Table 5-20.

Note

Regardless of reset source, CPU2 and CM each require their respective IPCFLG0 to be set by CPU1 on **every reset** in order to confirm the contents of IPCBOOTMODE are valid and continue their boot process.

Table 5-19. CPU2 Boot Procedure

CPU2 State	CPU1 Application Actions
Held in Reset	1. Configures CPU2 clocks 2. Configures the CPU1TOCPU2IPCBOOTMODE register (Refer to Section 5.7.2.2 for configuration details) 3. Sets CPU1TOCPU2IPCFLG0(1) 4. Releases CPU2 from being held in reset
In Wait Boot Mode waiting for the IPC Flag	1. Configures the CPU1TOCPU2IPCBOOTMODE register (Refer to Section 5.7.2.2 for configuration details) 2. Sets CPU1TOCPU2IPCFLG0(1)

(1) CPU2 will ACK and clear this IPC flag during boot up.

Table 5-20. CM Boot Procedure

CM State	CPU1 Application Actions
Held in Reset	1. Configures CM clocks 2. Configures the CPU1TOCMIPCBOOTMODE register (Refer to Section 5.7.2.2 for configuration details) 3. Sets CPU1TOCMIPCFLG0(1) 4. Releases CM from being held in reset
In Wait Boot Mode waiting for IPC Flag	1. Configures the CPU1TOCMIPCBOOTMODE register (Refer to Section 5.7.2.2 for configuration details) 2. Sets CPU1TOCMIPCFLG0(1)

(1) CM will ACK and clear this IPC flag during boot up.

5.7.2.2 IPCBOOTMODE Details

This section details the CPU1TOCPU2IPCBOOTMODE and CPU1TOCMIPCBOOTMODE register bit-field configurations and requirements for booting CPU2/CM.

Note

If any of the bit-fields of CPU1TOCPU2IPCBOOTMODE or CPU1TOCMIPCBOOTMODE registers are set with invalid values, an error IPC command is sent to CPU1. CPU2/CM then enter a wait loop where CPU2/CM wait for CPU1 to re-configure the IPCBOOTMODE register correctly and issue a reset to the respective core.

Table 5-21. CPU1TOCPU2IPCBOOTMODE Register Details

Bit	Name	Valid Values	Description
31:24	Key	0x5A	Key must be set for this register to be considered valid.
23:20	Reserved	-	Reserved
19:16	IPC Message RAM Copy Length	0x0 = 0 words (Boot mode not used) 0x1 = 100 words 0x2 = 200 words ... 0x9 = 900 words 0xA = 1000 words(1)	Sets the data length (in words) for the "Copy from IPC Message RAM and Boot to M1RAM" boot mode. This is the number words to be copied from CPU1TOCPU2MSGRAM1 to CPU2 M1RAM. If not using this boot mode, set value to 0x0.
15:8	CPU2 Device Frequency	0xA = 10 MHz(2) 0xB = 11 MHz ... 0xC8 = 200 MHz(2)	Sets the clock frequency (in MHz) that CPU2 is configured at.
7:0	CPU2 Boot Mode	0x0 = None/Wait Boot 0x03 = Flash Boot Option 0 (Sector 0) 0x23 = Flash Boot Option 1 (Sector 4) 0x43 = Flash Boot Option 2 (Sector 8) 0x63 = Flash Boot Option 3 (Sector 13) 0x0A = Secure Flash Boot Option 0 (Sector 0) 0x2A = Secure Flash Boot Option 1 (Sector 4) 0x4A = Secure Flash Boot Option 2 (Sector 8) 0x6A = Secure Flash Boot Option 3 (Sector 13) 0x0C = IPC Message RAM copy and boot to M1RAM 0x05 = Boot to M0RAM 0x0B = Boot to User OTP	Sets the boot mode for CPU2

(1) Values greater than 0xA are invalid.
(2) Values less than 0xA (10 MHz) or greater than 0xC8 (200 MHz) are invalid.

Table 5-22. CPU1TOCMIPCBOOTMODE Register Details

Bit	Name	Valid Values	Description
31:24	Key	0x5A	Key must be set for this register to be considered valid.
23:20	Reserved	-	Reserved
19:16	IPC Message RAM Copy Length	0x0 = 0 words / 0 bytes (Boot mode not used) 0x1 = 100 words / 200 bytes 0x2 = 200 words / 400 bytes ... 0x9 = 900 words / 1800 bytes 0xA = 1000 words / 2000 bytes(1)	Sets the data length (in words) for the "Copy from IPC Message RAM and Boot to S0RAM" boot mode. This is the number words to be copied from CPU1TOCMMSGRAM1 to CM S0RAM. If not using this boot mode, set value to 0x0.
15:8	CM Device Frequency	0xA = 10 MHz(2) 0xB = 11 MHz ... 0x7D = 125 MHz(2)	Sets the clock frequency (in MHz) that CM is configured at.
7:0	CM Boot Mode	0x0 = None/Wait Boot 0x03 = Flash Boot Option 0 (Sector 0) 0x23 = Flash Boot Option 1 (Sector 4) 0x43 = Flash Boot Option 2 (Sector 8) 0x63 = Flash Boot Option 3 (Sector 13) 0x0A = Secure Flash Boot Option 0 (Sector 0) 0x2A = Secure Flash Boot Option 1 (Sector 4) 0x4A = Secure Flash Boot Option 2 (Sector 8) 0x6A = Secure Flash Boot Option 3 (Sector 13) 0x0C = IPC Message RAM copy and boot to S0RAM 0x05 = Boot to S0RAM 0x0B = Boot to User OTP	Sets the boot mode for CM

(1) Values greater than 0xA are invalid.
(2) Values less than 0xA (10 MHz) or greater than 0x7D (125 MHz) are invalid.

5.7.2.3 Error IPC Command Table

This section details the IPC commands that CPU2 or CM can send to CPU1 to notify CPU1 regarding an error that occurred.

Note

After CPU2 or CM sends the error IPC command to CPU1, CPU2/CM will set CPU2TOCPU1IPCFLG0/CMTOCPU1IPCFLG0.

Table 5-23. CPU2 to CPU1 Error IPC Commands

Description	IPCSENDCOM Value	IPCSENDADDR Value
No Command	0x0000 0000	Not Used
IPCBOOTMODE Value(s) Incorrect	0xFFFF FFFF	Not Used
CPU2 in ITRAP	0xFFFF FFFE	If RAM is accessible, the address for the source of the ITRAP will be provided
CPU2 got NMI	0xFFFF FFFA	Not Used
CPU2 Secure Flash CMAC Calculation Failed	0xFFFF FFF9	Not Used

Table 5-24. CM to CPU1 Error IPC Commands

Description	IPCSENDCOM Value	IPCSENDADDR Value
No Command	0x0000 0000	Not Used
IPCBOOTMODE Value(s) Incorrect	0xFFFF FFFF	Not Used
CM got Hard Fault Exception	0xFFFF FFFE	Not Used
CM got Unsupported Interrupt	0xFFFF FFFB	Active Exception Number
CM got NMI	0xFFFF FFFA	Not Used
CM Secure Flash CMAC Calculation Failed	0xFFFF FFF9	Not Used

5.7.3 Entry Points

This section gives details about the entry point addresses for various boot modes. These entry points direct the boot ROM what address to branch to at the end of booting as per the selected boot mode.

Table 5-25. Entry Point Addresses for CPU1

Entry Point	Details	Address
Flash / Secure Flash (Option 0)	Flash Sector 0	0x0008 0000
Flash / Secure Flash (Option 1)	Flash Sector 4	0x0008 8000
Flash / Secure Flash (Option 2)	Flash Sector 8	0x000A 8000
Flash / Secure Flash (Option 3)	Flash Sector 13	0x000B E000
RAM	M0RAM	0x0000 0000

Table 5-26. Entry Point Addresses for CPU2

Entry Point	Details	Address
Flash / Secure Flash (Option 0)	Flash Sector 0	0x0008 0000
Flash / Secure Flash (Option 1)	Flash Sector 4	0x0008 8000
Flash / Secure Flash (Option 2)	Flash Sector 8	0x000A 8000
Flash / Secure Flash (Option 3)	Flash Sector 13	0x000B E000
RAM	M0RAM	0x0000 0000
CPU1 IPC Message RAM Copy to CPU2 RAM	M1RAM	0x0000 0400
User OTP	CPU2 User OTP	0x0007 8000

Table 5-27. Entry Point Addresses for CM

Entry Point	Details	Address
Flash / Secure Flash (Option 0)	Flash Sector 0	0x0020 0000
Flash / Secure Flash (Option 1)	Flash Sector 4	0x0021 0000
Flash / Secure Flash (Option 2)	Flash Sector 8	0x0025 0000
Flash / Secure Flash (Option 3)	Flash Sector 13	0x0027 C000
RAM	S0RAM	0x2000 0800
CPU1 IPC Message RAM Copy to CM RAM	S0RAM	0x2000 0800
User OTP	CM User OTP	0x003C 0000

5.7.4 Wait Points

During boot ROM execution, there are situations where the CPU may enter a wait loop in the code. This state can occur for a variety of reasons. Table 5-28, Table 5-29, and Table 5-30 detail the address ranges that the CPU PC register for each core will fall between if boot has entered one of these instances.

Table 5-28. Wait Point Addresses for CPU1

Silicon Rev0 Address Range	Silicon RevA Address Range	Description
0x3FB100 – 0x3FB106	0x3FB112 – 0x3FB117	In Wait Boot Mode
0x3FBEE2 – 0x3FBF09	0x3FBEEB – 0x3FBF12	In SCI Boot waiting on autobaud lock
0x3FE839 – 0x3FE92D	0x3FE839 – 0x3FE92D	In NMI Handler
0x3FE7F1 – 0x3FE823	0x3FE7F1 – 0x3FE823	In PIE Vector Mismatch Handler
0x3FE944 – 0x3FE970	0x3FE944 – 0x3FE970	In ITRAP ISR
0x3FB12A – 0x3FB12E	0x3FB13C – 0x3FB142	Failed secure Flash CMAC verification loop

Table 5-29. Wait Point Addresses for CPU2

Silicon Rev0,A Address Range	Description
0x3FB41D – 0x3FB42B	In Wait Boot Mode waiting for boot command
0x3FB459 – 0x3FB503	In NMI Handler
0x3FB504 – 0x3FB559	In ITRAP ISR
0x3FB173 – 0x3FB1B7	In loop due to invalid CPU1TOCPU2IPCBOOTMODE value(s) and/or CPU1TOCPU2IPCFLG0 is not set

Table 5-30. Wait Point Addresses for CM

Silicon Rev0,A Address Range	Description
0x180C – 0x1818	In Wait Boot Mode waiting for boot command
0x4018 – 0x41C2	In NMI Handler
0x41C4 – 0x41E8	In Hard Fault Handler
0x41EA – 0x421A	In Default Interrupt Handler
0x1618 – 0x16C4	In loop due to invalid CPU1TOCMIPCBOOTMODE value(s) and/or CPU1TOCMIPCFLG0 is not set

5.7.5 Memory Maps

This section details the ROM memory maps.

5.7.5.1 Boot ROM Memory Maps

Table 5-31. CPU1 Boot ROM Memory Map

Memory	Origin Address	Length (Words)
ROM Signature	0x003E 8000	0x0002
AES Tables	0x003E 8002	0x1400
IQmath Tables	0x003E 9402	0x166D
FPU32 Fast Tables	0x003F 6946	0x081A
FPU64 Fast Tables	0x003F 7160	0x0D30
FPU32 Twiddle Tables	0x003F 7E90	0x0DF8
FPU64 Twiddle Tables	0x003F 8DD0	0x1BF0
Boot	0x003F A9C0	0x3E00
Interrupt Handlers	0x003F E7C0	0x01B1
CPU Fast Data(1)	0x003F EA22	0x0100
Boot Checksum	0x003F FE40	0x0042
Full ROM Checksum	0x003F FEC0	0x0042
CRC Table	0x003F FF32	0x0008
Version	0x003F FF7A	0x0002
Vector Table	0x003F FFBE	0x0042

(1) Check the data manual to determine if these are available for your device part number. If not available, treat these sections as reserved.

Table 5-32. CPU2 Boot ROM Memory Map

Memory	Origin Address	Length (Words)
ROM Signature	0x003E 8000	0x0002
AES Tables	0x003E 8002	0x1400
IQmath Tables	0x003E 9402	0x166D
FPU32 Fast Tables	0x003F 6946	0x081A
FPU64 Fast Tables	0x003F 7160	0x0D30
FPU32 Twiddle Tables	0x003F 7E90	0x0DF8
FPU64 Twiddle Tables	0x003F 8DD0	0x1BF0
Boot	0x003F A9C0	0x3E00
Interrupt Handlers	0x003F E7C0	0x0173
CPU Fast Data(1)	0x003F EA22	0x0100
Boot Checksum	0x003F FE40	0x0042
Full ROM Checksum	0x003F FEC0	0x0042
CRC Table	0x003F FF32	0x0008
Version	0x003F FF7A	0x0002
Vector Table	0x003F FFBE	0x0042

(1) Check the data manual to determine if these are available for your device part number. If not available, treat these sections as reserved.

Table 5-33. CM Boot ROM Memory Map

Memory	Origin Address	Length (Bytes)
Vector Table	0x0000 0000	0x0140
Version	0x0000 0140	0x0004
Boot Checksum	0x0000 0144	0x0084
Full ROM Checksum	0x0000 01C8	0x0084
Boot	0x0000 024C	0x3DCC
Interrupt Handlers	0x0000 4018	0x0204
CRC Table	0x0000 FBFC	0x0400
ROM Signature	0x0000 FFFC	0x0004

5.7.5.2 CLA Data ROM Memory Maps

Note

Load refers to the memory addresses where the C28x CPU can view the data. **Run** refers to the CLA memory addresses that the CLA uses to access the data.

Table 5-34. CPU1 CLA Data ROM Memory Map

Memory	Origin Address	Length (Words)
FFT Tables (Load)	0x0100 1070	0x0800
Data (Load)	0x0100 1870	0x078A
Version (Load)	0x0100 1FFA	0x0006
FFT Tables (Run)	0x0000 F070	0x0800
Data (Run)	0x0000 F870	0x078A
Version (Run)	0x0000 FFFA	0x0006

Table 5-35. CPU2 CLA Data ROM Memory Map

Memory	Origin Address	Length (Words)
FFT Tables (Load)	0x0100 1070	0x0800
Data (Load)	0x0100 1870	0x078A
Version (Load)	0x0100 1FFA	0x0006
FFT Tables (Run)	0x0000 F070	0x0800
Data (Run)	0x0000 F870	0x078A
Version (Run)	0x0000 FFFA	0x0006

5.7.5.3 Reserved RAM Memory Maps

This section details memory usage in RAM that is reserved for boot ROM to use. These memory sections must be reserved in the user application.

Table 5-36. CPU1 Reserved RAM Memory Map

Memory	Description	Origin Address	End Address	Length (Words)
RAM	Boot Status, Boot Mode, MPOST Status, Boot Stack	0x0000 0002	0x0000 01B0	0x01AF

Table 5-37. CPU2 Reserved RAM Memory Map

Memory	Description	Origin Address	End Address	Length (Words)
RAM	Boot Status, Boot Mode, Boot Stack	0x0000 0002	0x0000 01A8	0x01A7

Table 5-38. CM Reserved RAM Memory Map

Memory	Description	Origin Address	End Address	Length (Bytes)
RAM	Boot Status, Boot Mode, Boot Stack	0x2000 0000	0x2000 07FF	0x0800

5.7.6 ROM Tables

Table 5-39 details the boot ROM and CLA ROM symbol libraries that can be integrated into an application to use the available ROM functions and tables.

Table 5-39. ROM Symbol Tables

ROM Symbols	Library Name	Location
ROM Bootloaders and Functions	F2838xCPU1_BootROM_Symbols	Under */libraries/boot_rom* in C2000Ware
FPU32 and FPU64 Tables	F2838xCPU1_BootROM_Symbols F2838xCPU2_BootROM_Symbols	
AES Tables	F2838xCPU1_BootROM_Symbols F2838xCPU2_BootROM_Symbols	
CLA Data ROM	F2838xCPU1_CLADATAROM_Symbols F2838xCPU2_CLADATAROM_Symbols	
IQmath	F2838xCPU1_IQMathROM_Symbols F2838xCPU2_IQMathROM_Symbols	
Secure Zone Functions	F2838xCPU1_SecureZoneCode_Symbols F2838xCPU2_SecureZoneCode_Symbols F2838xCM_SecureZoneCode_Symbols	

5.7.7 Boot Modes and Loaders

The available boot modes and bootloaders supported on this device are detailed in this section.

5.7.7.1 Boot Modes

Table 5-40 details the available boot modes that do not involve a peripheral boot loader.

Table 5-40. Boot Mode Availability

Boot Mode	CPU Support
Wait Boot	CPU1, CPU2, CM
Flash Boot	CPU1, CPU2, CM
Secure Flash Boot	CPU1, CPU2, CM
RAM Boot	CPU1, CPU2, CM
User OTP Boot	CPU2, CM
IPC Message Copy to RAM Boot	CPU2, CM

5.7.7.1.1 Wait Boot

The wait boot mode puts the CPU in a loop and does not branch to the user application code. The device can either enter wait boot mode through configuration or because an error occurred during boot up. TI recommends using wait boot when using a debugger to avoid any JTAG complications. CPU2 and CM can exit wait boot mode and run a different boot mode when CPU1 sets the respective CPU IPCFLG0. More information regarding this can found at Section 5.7.2.

Table 5-41. Reasons for Entering Wait Boot

CPU	Actions Resulting in Wait Boot
CPU1	• Wait boot is selected by user configuration • Decoded boot mode is unrecognized/invalid when a debugger is connected to the device • The emulation BOOTPIN_CONFIG key is not set to 0xA5 or 0x5A
CPU2	• Wait boot is selected by user configuration • Decoded boot mode is unrecognized/invalid when a debugger is connected to the device • Secure Flash boot CMAC calculation returns failure • IPC Message copy to RAM length specified is invalid (out of range) when trying to use this boot mode
CM	• Wait boot is selected by user configuration • Decoded boot mode is unrecognized/invalid when a debugger is connected to the device • Secure Flash boot CMAC calculation returns failure • IPC Message copy to RAM length specified is invalid (out of range) when trying to use this boot mode

5.7.7.1.2 Flash Boot

Flash boot mode branches to the configured memory address in Flash. Refer to Section 5.7.3 for all the available Flash address options.

For CPU1, on an unprogrammed device, Flash boot can be switched to USB boot. See more details in Section 5.3.

5.7.7.1.3 Secure Flash Boot

Secure Flash boot mode is similar to Flash boot mode in that the boot flow branches to the configured memory address in Flash except only after the Flash memory contents have been authenticated. The Flash authentication uses a Cipher-based Message Authentication Protocol (CMAC) to authenticate 16-KB of Flash starting from the configured Flash entry point address. The CMAC calculation requires a user-defined 128-bit key programmed in the CPU1 User OTP Zone 1 Header OTP CMACKEY bit field. Additionally, the user must calculate the golden CMAC tag based on the 16-KB Flash memory range and store it along with the user code at a hardcoded address in Flash. During secure Flash boot, the calculated CMAC tag is compared to the user golden CMAC tag in Flash to determine the pass/fail status of the CMAC authentication. When authentication passes, boot flow continues and branches to Flash to begin executing the application. When authentication fails, the boot flow actions performed vary by core. Refer to Table 5-43 for details on failure actions for each core.

For the available secure Flash boot entry address options, refer to Section 5.7.3.

For generating the secure Flash golden CMAC tag for CPU1 or CPU2, refer to the section “Using Secure Flash Boot on TMS320F2838x Devices” in the *TMS320C28x Assembly Language Tools User’s Guide* for instructions.

For generating the secure Flash golden CMAC tag for CM, refer to the ARM Assembly Language Tools v19.6.0.STS, within section “Using Secure Flash Boot on TMS320F2838x Devices” for instructions.

Note

- User must make sure that the Flash sector that encompasses the configured Flash entry point and the first 16KB of Flash is assigned to Zone 1 for any cores setup to use secure Flash boot.
- Recommended to use device JTAGLOCK when using secure Flash boot.
- If only using secure Flash boot for CPU2/CM, then the CPU1 application must first dummy load the Z1 OTP CMACKEY before releasing CPU2/CM from reset. When dummy loading, CPU1 application must first disable Flash data caching, then perform the dummy load, and then the application can re-enable Flash data caching.

Table 5-42. Secure Flash Tag and Key Details

Name	Address	Details
CMAC Golden Tag (128-bit)	**CPU1/CPU2:** *Flash Entry Point Address* + 0x2 **CM:** *Flash Entry Point Address* + 0x4	Located in Flash, offset from the entry point address, by 2 words (CPU1/CPU2) or 4 bytes (CM). When CMAC calculations are performed, the golden tag location in memory is considered all 0xFs. Refer to Example 5-1 for an example regarding linker configuration on CPU1. Lower memory contains the tag's MSW and higher memory contains the LSW **Example (on CPU1):** Tag = 0x00112233 44556677 8899AABB CCDDEEFF Address 0x0 = 0x00112233 Address 0x2 = 0x44556677 Address 0x4 = 0x8899AABB Address 0x6 = 0xCCDDEEFF

Table 5-42. Secure Flash Tag and Key Details (continued)

Name	Address	Details
CMAC 128-Bit Key	0x0007 8018	Located in CPU1 Zone 1 User Header OTP (CMACKEY0, CMACKEY1, CMACKEY2, CMACKEY3) CMACKEY0 contains the key's MSW and CMACKEY3 contains the LSW **Example:** Key = 0x00112233 44556677 8899AABB CCDDEEFF CMACKEY0 = 0x00112233 CMACKEY1 = 0x44556677 CMACKEY2 = 0x8899AABB CMACKEY3 = 0xCCDDEEFF

Table 5-43. Secure Flash Authentication Failure Actions

CPU	Action on Failed Authentication
CPU1	Reset the device (If using debugger, device halts)
CPU2	Send IPC message to CPU1, update CPU2 boot status to indicate failure, and return to wait boot
CM	Send IPC message to CPU1, update CM boot status to indicate failure, and return to wait boot

Table 5-44. Secure Flash on all CPUs Recommended Flow

Step	Action
1	Secure Flash boot CPU1
2	CPU1 application configures CPU2 and CM to boot using *Secure Flash Boot* and releases CPU2/CM from reset
3	CPU2 and CM perform secure Flash boot
4	CPU2 and CM applications signal to CPU1 using IPC that booting is complete
5	For CPU1/CPU2/CM, any Flash beyond the first 16KB from the entry point that is planned for use must be authenticated by the user using a different CMAC golden tag embedded at an address somewhere within the already authenticated 16KB of Flash

Example 5-1. Secure Flash CPU1 Linker File Example

```
MEMORY
{
/* Code Start branch to _c_int00 */
BEGIN : origin = 0x80000, length = 0x0002
/* User calculated golden CMAC tag for Flash Sector 0 */
GOLDEN_CMAC_TAG : origin = 0x80002, length = 0x0008
/* Flash Sector 0 containing application code */
FLASH_SECTOR_0 : origin = 0x8000A, length = 0x1FF6
.
.
.
}
```

5.7.7.1.4 RAM Boot

RAM boot mode branches to the configured memory address in RAM. Refer to Section 5.7.3 for all the available RAM address options.

5.7.7.1.5 User OTP Boot

User OTP boot mode branches to the configured memory address in User OTP. Refer to Section 5.7.3 for all the available user OTP address options.

5.7.7.1.6 IPC Message Copy to RAM Boot

IPC message copy to RAM involves copying application code from CPU1 IPC message RAM 1 to CPU2/CM destination RAM for execution. The maximum copy length size is 1000 words (2000 bytes) and the minimum is 100 words (200 bytes). Refer to Table 5-45 for details regarding configuration and execution flow of this boot mode.

Table 5-45. IPC Message Copy Steps

Step	Action
1	CPU1 application links CPU2/CM code to copy in either CPU1TOCPU2MSGRAM1 or CPU1TOCMMSGRAM1.
2	CPU1 application configures CPU2/CM IPCBOOTMODE with the copy length and IPC message copy as the boot mode. See Section 5.7.2.2 for more IPCBOOTMODE details.
3	Once CPU2/CM is released from reset to boot, CPU2/CM begins copying from CPU1TOCPU2MSGRAM1/ CPU1TOCMMSGRAM1 to their destination RAM. Refer to Table 5-46 for destination RAM details.
4	Once the copying is complete, CPU2/CM boot branches to the entry address of the destination RAM and begins executing the application.

Table 5-46. IPC Message Copy Destination Address

CPU	RAM	Destination Address Range(1) (For maximum copy length of 1000 words)
CPU2	M1RAM	0x0000 0400 to 0x0000 07E6
CM	S0RAM	0x2000 0800 to 0x2000 0FCC

(1) This memory must be allocated and reserved in the CPU2/CM application linker command file when using this boot mode.

5.7.7.2 Bootloaders

This section details the available boot modes that use a peripheral boot loader. For more specific details on the supported data stream structure used by the following bootloaders, refer to Section 5.8.1.

Note

These are only available on CPU1.

5.7.7.2.1 SCI Boot Mode

The SCI boot mode asynchronously transfers code from SCI-A to internal memory. This boot mode only supports an incoming 8-bit data stream and follows the data flow as outlined in Example 5-2.

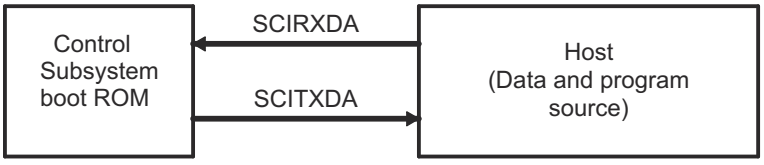


Figure 5-6. Overview of SCI Bootloader Operation

The device communicates with the external host by communication through the SCI-A peripheral. The autobaud feature of the SCI port is used to lock baud rates with the host. For this reason the SCI loader is very flexible and you can use a number of different baud rates to communicate with the device.

After each data transfer, the bootloader echoes back the 8-bit character received to the host. This allows the host to check that each character was received by the bootloader.

At higher baud rates, the slew rate of the incoming data bits can be affected by transceiver and connector performance. While normal serial communications can work well, this slew rate can limit reliable auto-baud detection at higher baud rates (typically beyond 100 kbaud) and cause the auto-baud lock feature to fail. To avoid this, the following is recommended:

1. Achieve a baud-lock between the host and SCI bootloader using a lower baud rate.
2. Load the incoming application or custom loader at this lower baud rate.
3. The host can then handshake with the loaded application to set the SCI baud rate register to the desired high baud rate.

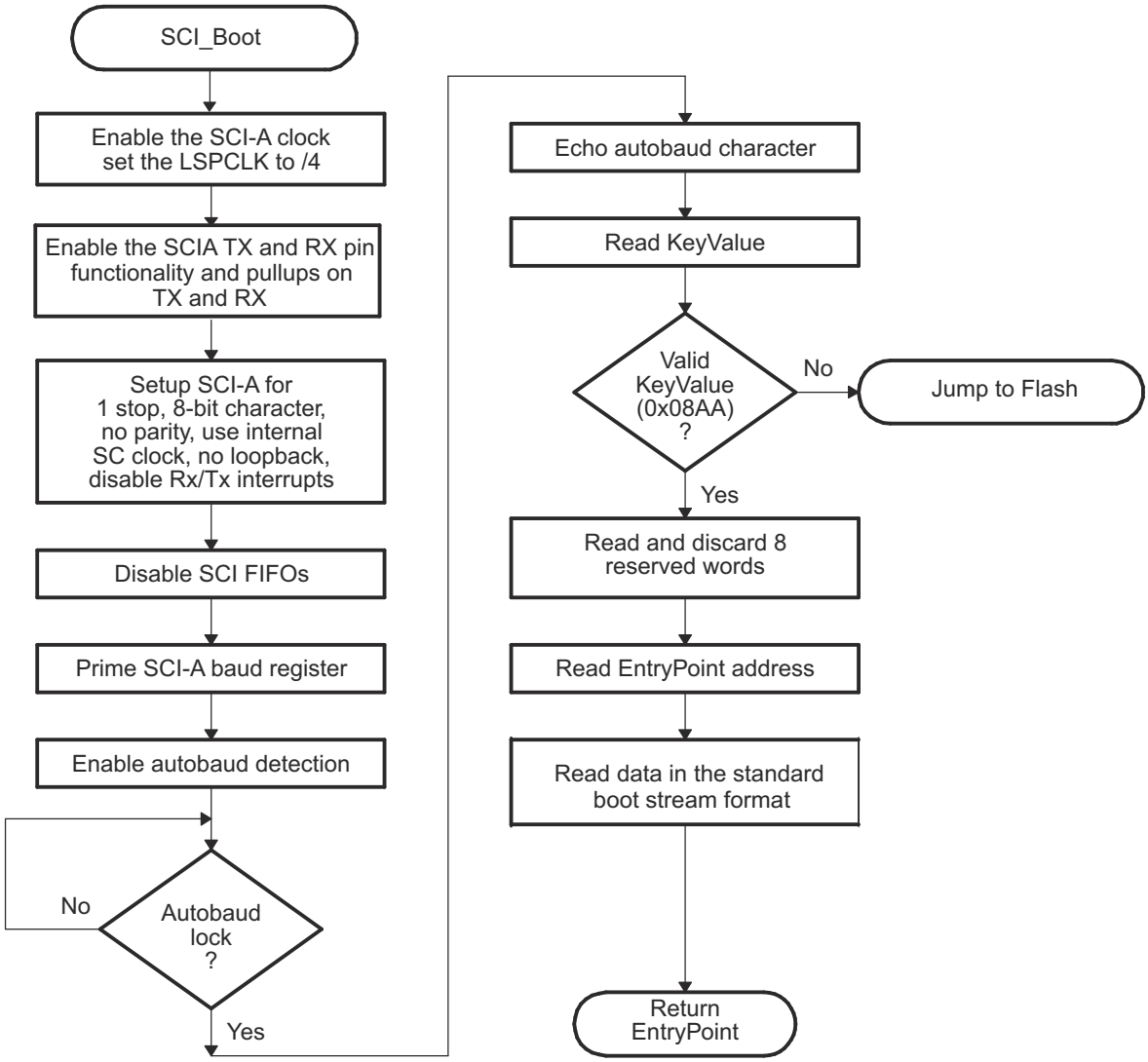


Figure 5-7. Overview of SCI Boot Function

5.7.7.2.2 SPI Boot Mode

The SPI loader expects an SPI-compatible 16-bit or 24-bit addressable serial EEPROM or serial Flash device to be present on the SPI-A pins as indicated in Figure 5-8. The SPI bootloader supports an 8-bit data stream and does not support a 16-bit data stream.

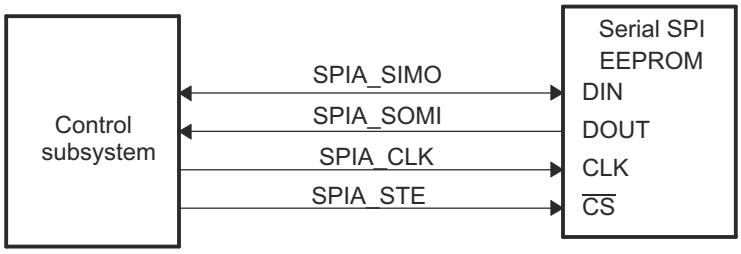


Figure 5-8. SPI Loader

The SPI boot ROM loader initializes the SPI module to interface to a serial SPI EEPROM or Flash. Devices of this type include, but are not limited to, the Xicor X25320 (4Kx8) and Xicor X25256 (32Kx8) SPI serial SPI EEPROMs and the Atmel AT25F1024A serial Flash.

The SPI boot ROM loader initializes the SPI with the following settings: FIFO enabled, 8-bit character, internal SPICLK master mode and talk mode, clock phase = 1, polarity = 0, using the slowest baud rate.

If the download is to be performed from an SPI port on another device, then that device must be set up to operate in the slave mode and mimic a serial SPI EEPROM. Immediately after entering the SPI_Boot function, the pin functions for the SPI pins are set to primary and the SPI is initialized. The initialization is done at the slowest speed possible. Once the SPI is initialized and the key value read, specify a change in baud rate or low-speed peripheral clock.

Table 5-47. SPI 8-Bit Data Stream

Byte	Contents
1	LSB: AA (KeyValue for memory width = 8-bits)
2	MSB: 08h (KeyValue for memory width = 8-bits)
3	LSB: LOSPCP
4	MSB: SPIBRR
5	LSB: reserved for future use
6	MSB: reserved for future use
...	...
...	Reserved
	...
17	LSB: reserved for future use
18	MSB: reserved for future use
19	LSB: Upper half (MSW) of Entry point PC[23:16]
20	MSB: Upper half (MSW) of Entry point PC[31:24] (Note: Always 0x00)
21	LSB: Lower half (LSW) of Entry point PC[7:0]
22	MSB: Lower half (LSW) of Entry point PC[15:8]
...	....
...	Data for this section.
	...
...	Blocks of data in the format size/destination address/data as shown in the generic data stream description
...	...
...	Data for this section.
	...
n	LSB: 00h
n+1	MSB: 00h - indicates the end of the source

The data transfer is done in "burst" mode from the serial SPI EEPROM. The transfer is carried out entirely in byte mode (SPI at 8 bits/character). A step-by-step description of the sequence follows:

1. The SPI-A port is initialized.
2. The GPIO pin, as defined by SPI option configured from Table 5-61, is used as a chip-select for the serial SPI EEPROM or Flash.
3. The SPI-A outputs a read command for the serial SPI EEPROM or Flash.
4. The SPI-A sends the serial SPI EEPROM an address 0x0000; that is, the host requires that the EEPROM or Flash must have the downloadable packet starting at address 0x0000 in the EEPROM or Flash. The loader is compatible with both 16-bit addresses and 24-bit addresses.
5. The next word fetched must match the key value for an 8-bit data stream (0x08AA). The least significant byte of this word is the byte read first and the most significant byte is the next byte fetched. This is true of all word transfers on the SPI. If the key value does not match, then the load is aborted and the bootloader jumps to Flash.
6. The next two bytes fetched can be used to change the value of the low speed peripheral clock register (LOSPCP) and the SPI baud rate register (SPIBRR). The first byte read is the LOSPCP value and the second byte read is the SPIBRR value. The next seven words are reserved for future enhancements. The SPI bootloader reads these seven words and discards them.
7. The next two words makeup the 32-bit entry point address where execution continues after the boot load process is complete. This is typically the entry point for the program being downloaded through the SPI port.
8. Multiple blocks of code and data are then copied into memory from the external serial SPI EEPROM through the SPI port. The blocks of code are organized in the standard data stream structure presented earlier. This is done until a block size of 0x0000 is encountered. At that point in time the entry point address is returned to the calling routine that then exits the bootloader and resumes execution at the address specified.

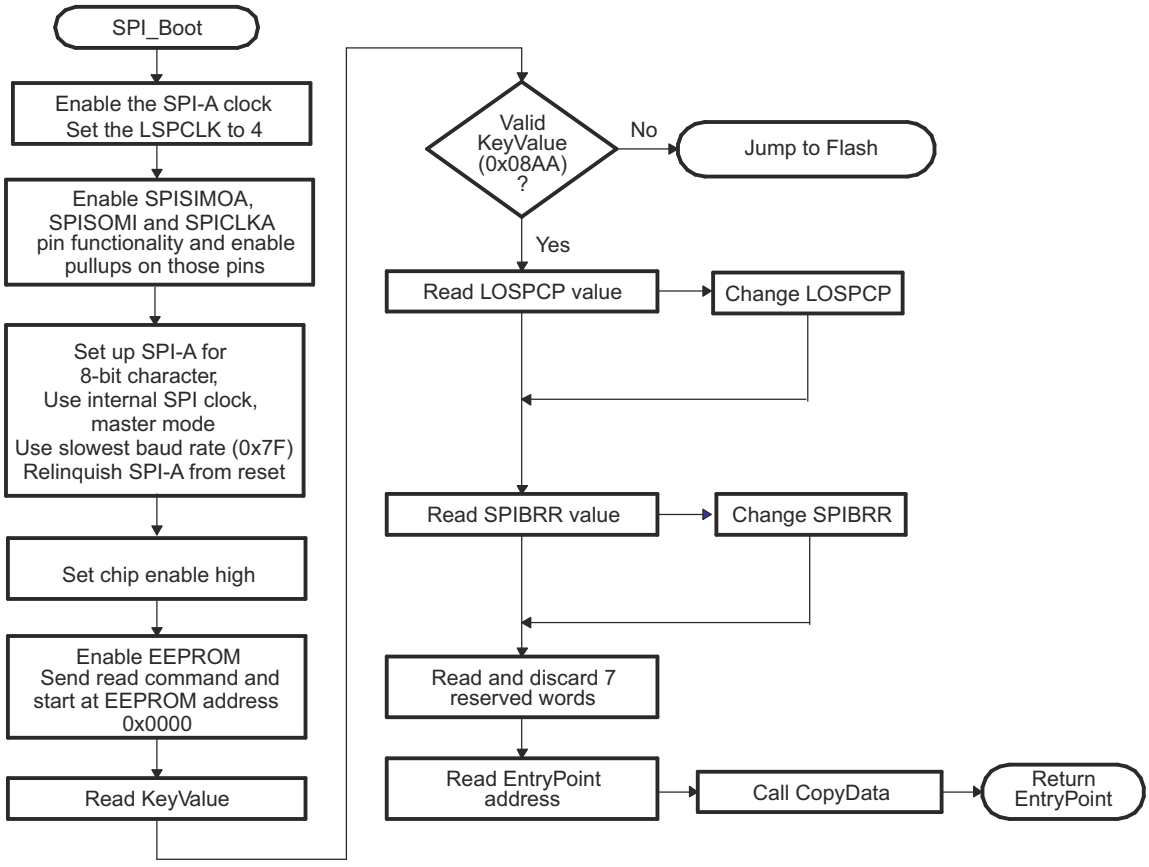


Figure 5-9. Data Transfer From EEPROM Flow

5.7.7.2.3 I2C Boot Mode

The I2C bootloader expects an 8-bit wide I2C-compatible EEPROM device to be present at address 0x50 on the I2C-A bus as indicated in Figure 5-10. The EEPROM must adhere to conventional I2C EEPROM protocol, as described in this section, with a 16-bit base address architecture.

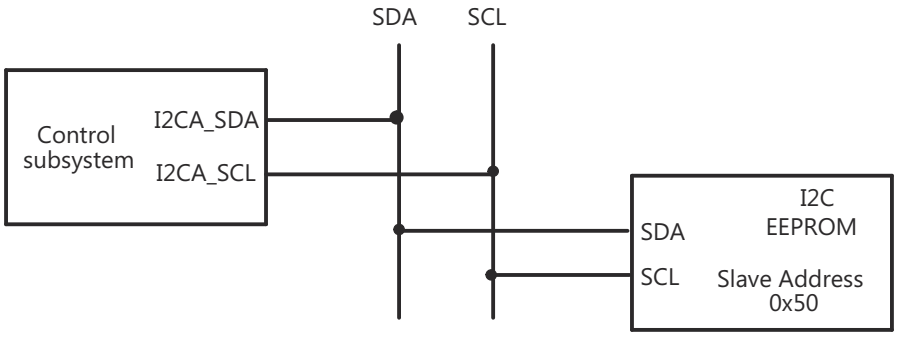


Figure 5-10. EEPROM Device at Address 0x50

If the download is to be performed from a device other than an EEPROM, then that device must be set up to operate in the slave mode and mimic the I2C EEPROM. Immediately after entering the I2C boot function, the GPIO pins are configured for I2C-A operation and the I2C is initialized. The following requirements must be met when booting from the I2C module:

- The input frequency to the device must be in the appropriate range.
- The EEPROM must be at slave address 0x50.

The bit-period prescalers (I2CCLKH and I2CCLKL) are configured by the bootloader to run the I2C at a 50 percent duty cycle at 100-kHz bit rate (standard I2C mode) when the system clock is 10 MHz. These registers can be modified after receiving the first few bytes from the EEPROM. This allows the communication to be increased up to a 400-kHz bit rate (fast I2C mode) during the remaining data reads.

Arbitration, bus busy, and slave signals are not checked. Therefore, no other master is allowed to control the bus during this initialization phase. If the application requires another master during I2C boot mode, that master must be configured to hold off sending any I2C messages until the application software signals that the application is past the bootloader portion of initialization.

The non-acknowledgment bit is checked only during the first message sent to initialize the EEPROM base address. This is to make sure that an EEPROM is present at address 0x50 before continuing. If an EEPROM is not present, the non-acknowledgment bit is not checked during the address phase of the data read messages (I2C_Get Word). If a non-acknowledgment is received during the data read messages, the I2C bus hangs. Table 5-48 shows the 8-bit data stream used by the I2C.

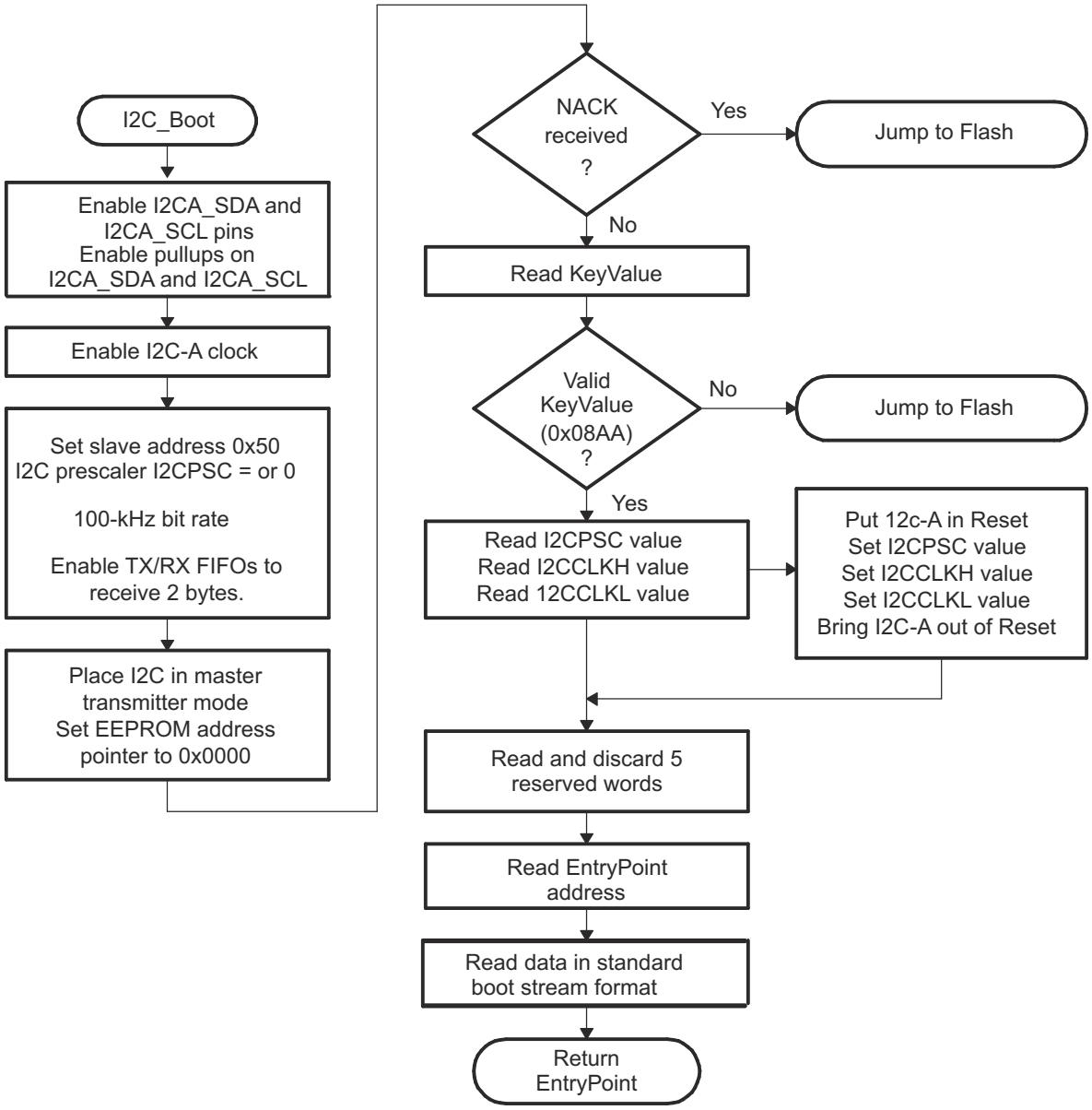


Figure 5-11. Overview of I2C Boot Function

Table 5-48. I2C 8-Bit Data Stream

Byte	Contents
1	LSB: AA (KeyValue for memory width = 8 bits)
2	MSB: 08h (KeyValue for memory width = 8 bits)
3	LSB: I2CPSC[7:0]
4	reserved
5	LSB: I2CCLKH[7:0]
6	MSB: I2CCLKH[15:8]
7	LSB: I2CCLKL[7:0]
8	MSB: I2CCLKL[15:8]
...	...
...	Data for this section.
	...
17	LSB: Reserved for future use
18	MSB: Reserved for future use
19	LSB: Upper half of entry point PC
20	MSB: Upper half of entry point PC[22:16] (Note: Always 0x00)
21	LSB: Lower half of entry point PC[15:8]
22	MSB: Lower half of entry point PC[7:0]
...	...
...	Data for this section.
	...
	Blocks of data in the format size/destination address/data as shown in the generic data stream description.
...	...
...	Data for this section.
	...
n	LSB: 00h
n+1	MSB: 00h - indicates the end of the source

The I2C EEPROM protocol required by the I2C bootloader is shown in Figure 5-12 and Figure 5-13. The first communication, which sets the EEPROM address pointer to 0x0000 and reads the KeyValue (0x08AA), is shown in Figure 5-12. All subsequent reads are shown in Figure 5-13 and are read two bytes at a time.

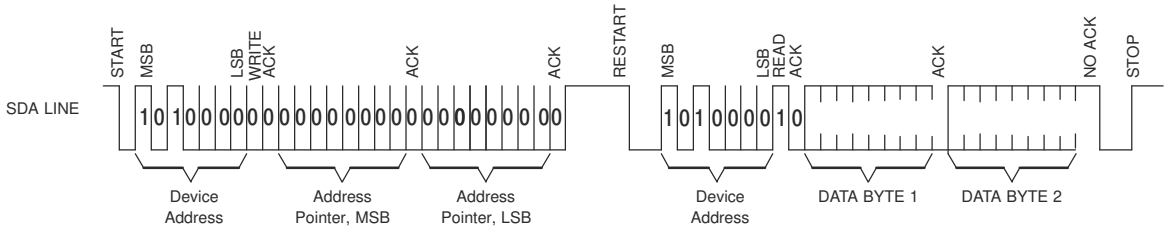


Figure 5-12. Random Read

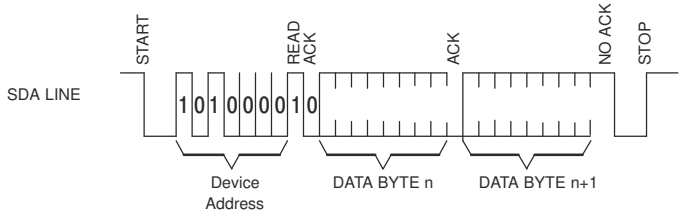


Figure 5-13. Sequential Read

5.7.7.2.4 Parallel Boot Mode

The parallel general purpose I/O (GPIO) boot mode asynchronously transfers code from GPIO88, GPIO62:GPIO58, GPIO90:GPIO89 to internal memory. Each value is eight bits long and follows the same data flow as outlined in Figure 5-14.

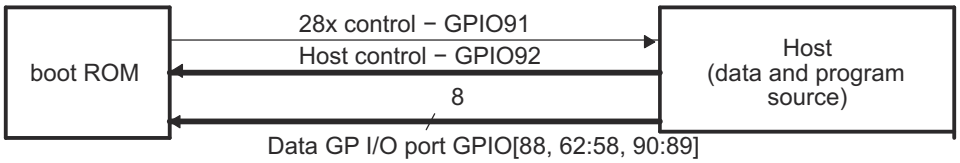


Figure 5-14. Overview of Parallel GPIO Bootloader Operation

The control subsystem communicates with the external host device by polling/driving the GPIO92 and GPIO91 lines. The handshake protocol shown in Figure 5-15 must be used to successfully transfer each word using GPIO[88, 62:58, 90:89]. This protocol is very robust and allows for a slower or faster host to communicate with the master subsystem.

Two consecutive 8-bit words are read to form a single 16-bit word. The least significant byte (LSB) is read first followed by the most significant byte (MSB). In this case, data is read from GPIO[88, 62:58, 90:89].

The 8-bit data stream is shown in Table 5-49.

Table 5-49. Parallel GPIO Boot 8-Bit Data Stream

Bytes		GPIO[88,62:58,90:89] (Byte 1 of 2)	GPIO[88,62:58,90:89] (Byte 2 of 2)	Description
1	2	AA	08	0x08AA (KeyValue for memory width = 16 bits)
3	4	00	00	8 reserved words (words 2 - 9)
...	...	...	...	...
17	18	00	00	Last reserved word
19	20	BB	00	Entry point PC[22:16]
21	22	DD	CC	Entry point PC[15:0] (PC = 0x00BBCCDD)
23	24	NN	MM	Block size of the first block of data to load = 0xMMNN words
25	26	BB	AA	Destination address of first block Addr[31:16]
27	28	DD	CC	Destination address of first block Addr[15:0] (Addr = 0xAABBCCDD)
29	30	BB	AA	First word of the first block in the source being loaded = 0xAABB
... ...				... Data for this section. ...
.		BB	AA	Last word of the first block of the source being loaded = 0xAABB
.		NN	MM	Block size of the 2nd block to load = 0xMMNN words
.		BB	AA	Destination address of second block Addr[31:16]
.		DD	CC	Destination address of second block Addr[15:0]
.		BB	AA	First word of the second block in the source being loaded
.				…
n	n+1	BB	AA	Last word of the last block of the source being loaded (More sections if required)
n+2	n+3	00	00	Block size of 0000h - indicates end of the source program

The device first signals the host that the device is ready to begin data transfer by pulling the GPIO91 pin low. The host load then initiates the data transfer by pulling the GPIO92 pin low. The complete protocol is shown in Figure 5-15.

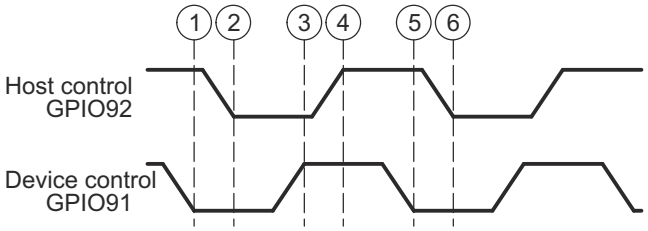


Figure 5-15. Parallel GPIO Bootloader Handshake Protocol

1. The device indicates the device is ready to start receiving data by pulling the GPIO91 pin low.
2. The bootloader waits until the host puts data on GPIO [88,62:58,90:89]. The host signals to the device that data is ready by pulling the GPIO92 pin low.
3. The device reads the data and signals the host that the read is complete by pulling GPIO91 high.
4. The bootloader waits until the host acknowledges the device by pulling GPIO92 high.
5. The device again indicates the device is ready for more data by pulling the GPIO91 pin low.

This process is repeated for each data value to be sent.

Figure 5-16 shows an overview of the Parallel GPIO bootloader flow.

Figure 5-17 shows the transfer flow from the host side. The operating speed of the CPU and host are not critical in this mode, as the host waits for the device and the device waits for the host. In this manner, the protocol works with both a host running faster and a host running slower than the device.

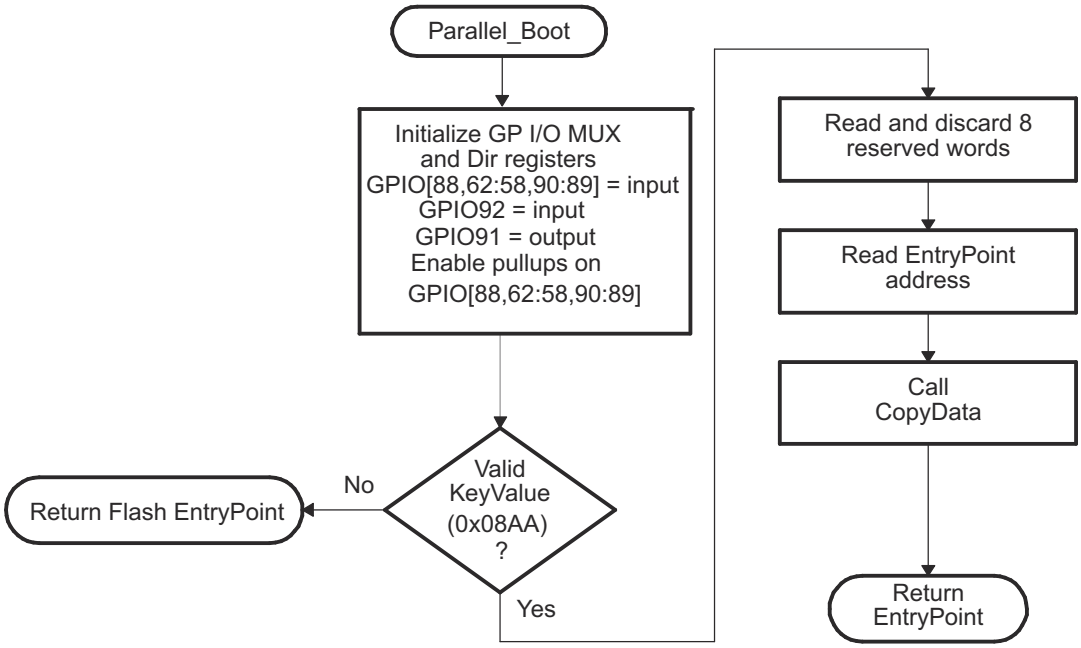


Figure 5-16. Parallel GPIO Mode Overview

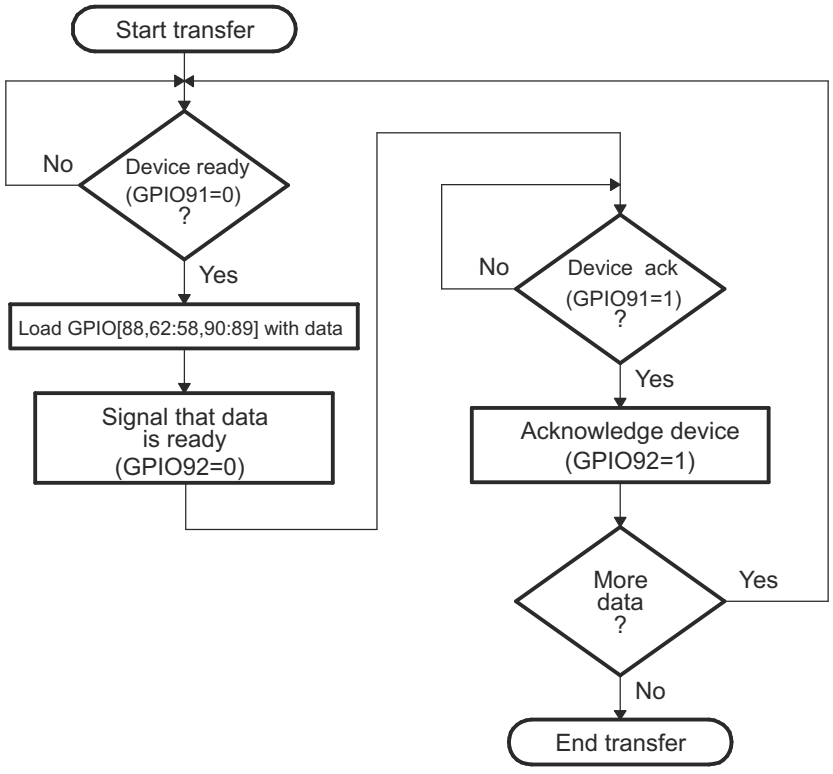


Figure 5-17. Parallel GPIO Mode - Host Transfer Flow

Figure 5-18 shows the flow used to read a single word of data from the parallel port.

- **8-bit data stream**

  The 8-bit routine, shown in Figure 5-18, discards the upper eight bits of the first read from the port and treats the lower eight bits masked with GPIO89 in bit position 7 and GPIO90 in bit position 6 as the least-significant byte (LSB) of the word to be fetched. The routine then performs a second read to fetch the most-significant byte (MSB). The routine then combines the MSB and LSB into a single 16-bit value to be passed back to the calling routine.

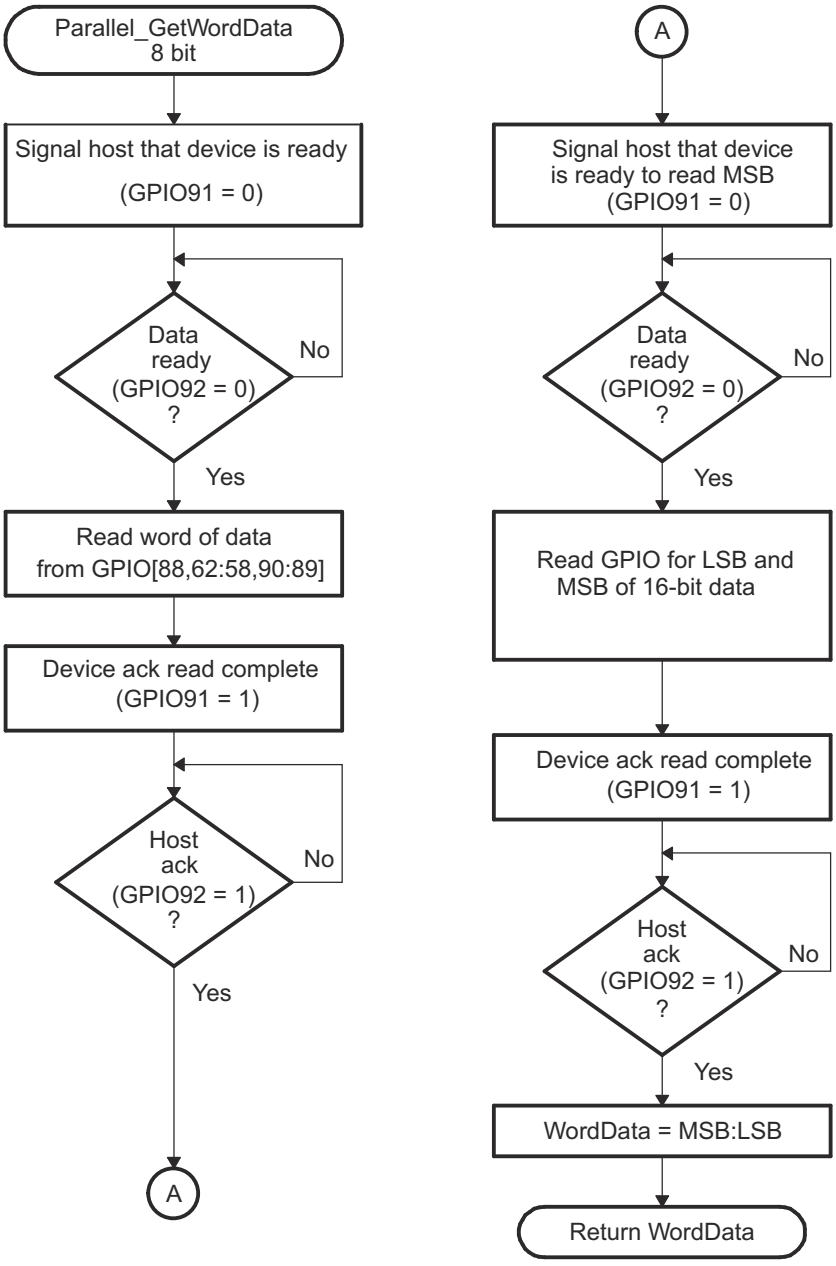


Figure 5-18. 8-Bit Parallel GetWord Function

5.7.7.2.5 CAN Boot Mode

The CAN bootloader asynchronously transfers code from CAN-A to internal memory. The host can be any CAN node. The communication is first done with 11-bit standard identifiers (with a MSGID of 0x1) using two bytes per data frame. The host can download a kernel to reconfigure the CAN if higher data throughput is desired.

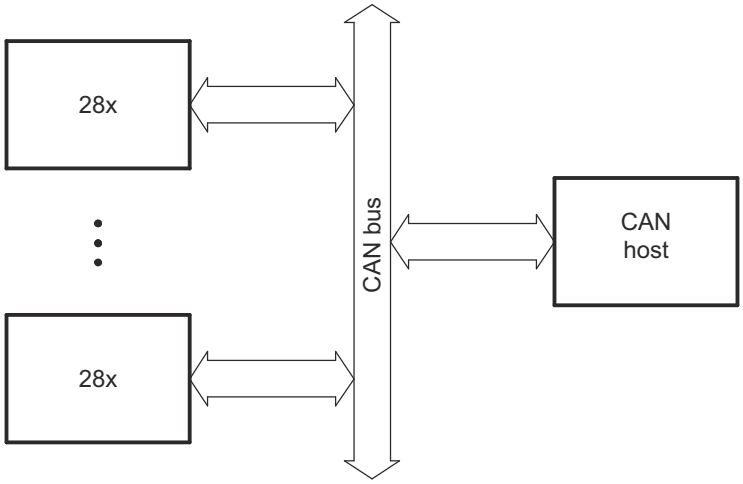


Figure 5-19. Overview of CAN-A Bootloader Operation

The bit timing registers are programmed in such a way that a 100-kbps bit rate is achieved with a 20-MHz external oscillator, a shown in Table 5-50.

Table 5-50. Bit-Rate Value for Internal Oscillators

OSCCLK	SYSCLK	Bit Rate
20 MHz	10 MHz	100 kbps

The SYSCLKOUT values shown are the reset values with the default PLL setting. The BRP and bit-time values are hard-coded to 10 and 20, respectively.

Note

The CPU1 CAN boot loader uses XTAL as the bit clock source and INTOSC2 as the system clock source.

Mailbox 1 is programmed with a standard MSGID of 0x1 for boot-loader communication. The CAN host must transmit only two bytes at a time, LSB first and MSB next. For example, to transmit the word 0x08AA to the device, transmit AA first, followed by 08. The program flow of the CAN bootloader is identical to the SCI bootloader. The data sequence for the CAN bootloader is shown in Table 5-51.

Table 5-51. CAN 8-Bit Data Stream

Bytes		Byte 1 of 2	Byte 2 of 2	Description
1	2	AA	08	0x08AA (KeyValue for memory width = 16 bits)
3	4	00	00	reserved
5	6	00	00	reserved
7	8	00	00	reserved
9	10	00	00	reserved
11	12	00	00	reserved
13	14	00	00	reserved
15	16	00	00	reserved
17	18	00	00	reserved
19	20	BB	AA	Entry point PC[22:16]
21	22	DD	CC	Entry point PC[15:0] (PC = 0xAABBCCDD)
23	24	NN	MM	Block size of the first block of data to load = 0xMMNN words
25	26	BB	AA	Destination address of first block Addr[31:16]
27	28	DD	CC	Destination address of the first block Addr[15:0] (Addr = 0xAABBCCDD)
29	30	BB	AA	First word of the first block in the source being loaded = 0xAABB
...				....
...				Data for this section. ...
.		BB	AA	Last word of the first block of the source being loaded = 0xAABB
.		NN	MM	Block size of the second block to load = 0xMMNN words
.		BB	AA	Destination address of the second block Addr[31:16]
.		DD	CC	Destination address of the second block Addr[15:0]
.		BB	AA	First word of the second block in the source being loaded
.				…
n	n+1	BB	AA	Last word of the last block of the source being loaded (More sections if required)
n+2	n+3	00	00	Block size of 0000h - indicates end of the source program

5.7.7.2.6 USB Boot Mode

In USB boot mode, the device enumerates with vendor ID 0x1CBE and product ID 0x00FF. The device descriptor and interface descriptor both show the class as 0xFF (vendor-specific), the subclass as 0x00, and the protocol as 0x00. After enumeration, the device waits for data. Data must be sent using bulk OUT transfers to endpoint 1. The data is interpreted as a series of 8-bit bytes in the standard data stream format described in Section 5.8.1, shown here in Table 5-52. No reserved bytes are used. Once the data transfer is complete (block size of 0x0000 sent), the device disconnects from the USB bus, allowing other software to make use of the module if desired. Figure 5-20 illustrates the flow for USB boot mode.

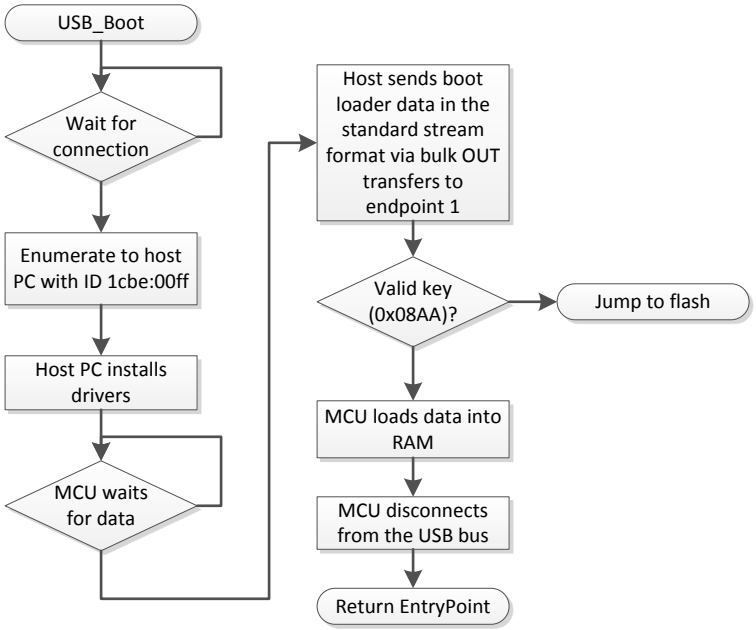


Figure 5-20. USB Boot Flow

Table 5-52. USB 8-Bit Data Stream

Bytes		First Byte (LSB)	Second Byte (MSB)	Description
1	2	AA	08	0x08AA (KeyValue for memory width = 16bits)
3	4	00	00	reserved
5	6	00	00	reserved
7	8	00	00	reserved
9	10	00	00	reserved
11	12	00	00	reserved
13	14	00	00	reserved
15	16	00	00	reserved
17	18	00	00	reserved
19	20	BB	AA	Entry point PC[22:16]
21	22	DD	CC	Entry point PC[15:0] (PC = 0xAABBCCDD)
23	24	NN	MM	Block size of the first block of data to load = 0xMMNN words
25	26	BB	AA	Destination address of the first block Addr[31:16]
27	28	DD	CC	Destination address of the first block Addr[15:0] (Addr = 0xAABBCCDD)
29	30	BB	AA	First word of the first block in the source being loaded = 0xAABB
...				....
...				Data for this section.
				...
.		BB	AA	Last word of the first block of the source being loaded = 0xAABB
.		NN	MM	Block size of the 2nd block to load = 0xMMNN words
.		BB	AA	Destination address of the second block Addr[31:16]
.		DD	CC	Destination address of the second block Addr[15:0]
.		BB	AA	First word of the second block in the source being loaded
.				…
n	n+1	BB	AA	Last word of the last block of the source being loaded (More sections if required)
n+2	n+3	00	00	Block size of 0000h - indicates end of the source program

Implementing PC-side USB software is not trivial. It is recommended to use the TI-provided tools and drivers to load data in USB boot mode. Hex and binary files for loader tools can be generated from COFF (.out) files using the hex2000 tool. To produce a plain binary file in the boot loader format, use the following command line:

hex2000 -boot -b Program_to_Load.out -o Binary_Loader_Data.dat

For more information on hex2000, see the *TMS320C28x Assembly Language Tools User's Guide*.

Note

INTOSC2 must be enabled before invoking the USB boot loader. If INTOSC2 is not enabled, the boot loader hangs. A debugger reset or SCC reset does not enable INTOSC2, if INTOSC2 has been disabled by the application.

5.7.8 GPIO Assignments for CPU1

This section details the GPIOs and boot option values used for each CPU1 boot mode set in the BOOT_DEF memory location located at Z1-BOOTDEF-LOW/ Z2-BOOTDEF-LOW and Z1-BOOTDEF-HIGH/ Z2-BOOTDEF-HIGH. Refer to Section 5.4.2 on how to configure BOOT_DEF. When selecting a boot mode option, make sure to verify that the necessary pins are available in the pin mux options for the specific device package being used.

Note

These configurations only apply to CPU1. Refer to Section 5.7.2 for details on configuring CPU2 and CM boot modes.

Table 5-53. SCI Boot Options

Option	BOOTDEF Value	SCITXDA GPIO	SCIRXDA GPIO
0 (default)	0x01	GPIO29	GPIO28
1	0x21	GPIO84	GPIO85
2	0x41	GPIO36	GPIO35
3	0x61	GPIO42	GPIO43
4	0x81	GPIO65	GPIO64
5	0xA1	GPIO135	GPIO136
6	0xC1	GPIO8	GPIO9

Table 5-54. CAN Boot Options

Option	BOOTDEF Value	CANTXA GPIO	CANRXA GPIO
0 (default)	0x02	GPIO37	GPIO36
1	0x22	GPIO71	GPIO70
2	0x42	GPIO63	GPIO62
3	0x62	GPIO19	GPIO18
4	0x82	GPIO4	GPIO5
5	0xA2	GPIO31	GPIO30

Table 5-55. I2C Boot Options

Option	BOOTDEF Value	SDAA GPIO	SCLA GPIO
0	0x07	GPIO91	GPIO92
1	0x27	GPIO32	GPIO33
2	0x47	GPIO42	GPIO43
3	0x67	GPIO0	GPIO1
4	0x87	GPIO104	GPIO105

Table 5-56. USB Boot Options

Option	BOOTDEF Value	USBDM GPIO	USBDP GPIO
0 (default)	0x09	GPIO42	GPIO43

Table 5-57. RAM Boot Options

Option	BOOTDEF Value	RAM Entry Point (Address)
0	0x05	0x0000 0000

Table 5-58. Flash Boot Options

Option	BOOTDEF Value	Flash Entry Point (Address)	Flash Sector
0 (default)	0x03	0x0008 0000	CPU1 Bank0 Sector 0
1	0x23	0x0008 8000	CPU1 Bank 0 Sector 4
2	0x43	0x000A 8000	CPU1 Bank 0 Sector 8
3	0x63	0x000B E000	CPU1 Bank 0 Sector 13

Table 5-59. Secure Flash Boot Options

Option	BOOTDEF Value	Flash Entry Point (Address)	Flash Sector
0	0x0A	0x0008 0000	CPU1 Bank0 Sector 0
1	0x2A	0x0008 8000	CPU1 Bank 0 Sector 4
2	0x4A	0x000A 8000	CPU1 Bank 0 Sector 8
3	0x6A	0x000B E000	CPU1 Bank 0 Sector 13

Table 5-60. Wait Boot Options

Option	BOOTDEF Value	Watchdog
0	0x04	Enabled
1	0x24	Disabled

Table 5-61. SPI Boot Options

Option	BOOTDEF Value	SPISIMOA	SPISOMIA	SPICLKA	SPISTEA
0	0x06	GPIO58	GPIO59	GPIO60	GPIO61
1	0x26	GPIO16	GPIO17	GPIO18	GPIO19
2	0x46	GPIO32	GPIO33	GPIO34	GPIO35
3	0x66	GPIO16	GPIO17	GPIO56	GPIO57
4	0x86	GPIO54	GPIO55	GPIO56	GPIO57

Table 5-62. Parallel Boot Options

Option	BOOTDEF Value	D0-D7 GPIO	DSP Control GPIO	Host Control GPIO
0 (default)	0x0	D0 - GPIO89 D1 - GPIO90 D2 - GPIO58 D3 - GPIO59 D4 - GPIO60 D5 - GPIO61 D6 - GPIO62 D7 - GPIO88	GPIO91	GPIO92

5.7.9 Secure ROM Function APIs

Within secure ROM of each core, functions are available to be called by the application to perform EXEONLY Flash/RAM tasks in a secure manner.

Note

The application must disable interrupts before calling one of the EXEONLY function APIs.

If a vector fetch request is given by the CPU (C28 or CM, depending on the subsystem) while the corresponding program counter (PC) is within the EXEONLY function API code of the Secure ROM, a reset occurs (RSN, if from C28; SYSRESETn, if from CM). The consequence of this is if an NMI, ITRAP, or Bus Fault occurs while the PC is executing one of the EXEONLY API functions, the NMI/ITRAP/Fault cannot be serviced because a reset occurs to that subsystem.

The **secure copy code zone 1 and zone 2 functions** allow EXEONLY Flash to be copied to EXEONLY RAM in a secure manner. The source must be from EXEONLY Flash and the destination to EXEONLY RAM. There is no support to copy EXEONLY ROM or EXEONLY RAM to RAM. Both Flash and RAM must be set to EXEONLY and configured for the same zone. Additionally, the copy size must not cross over the Flash sector boundary. Any violations of these requirements results in a failure status returned. Upon successful copy of the data, the number of 16-bit words copied is returned.

Table 5-63. Secure Copy Code Function

CPU	Function Prototype	Function Parameters	Function Return Value
CPU1, CPU2, CM	Uint16 **SecureCopyCodeZ1**(Uint32 size, Uint16 *dst, Uint16 *src) Uint16 **SecureCopyCodeZ2**(Uint32 size, Uint16 *dst, Uint16 *src)	*size* : The number of 16-bit words to copy *dst* : The destination memory address in EXEONLY RAM *src* : The source memory address in EXEONLY Flash	0xXXXX : Returns the number of 16-bit words copied 0x0000 : Indicates one of the following: Copy length is zero; Copy size crosses over Flash sector boundary; Flash and RAM do not belong to the same zone; Flash or RAM are not set to EXEONLY; Error occurred during data copy

The **secure CRC calculation zone 1 and zone 2 functions** allow a safety CRC check of EXEONLY memory in a secure manner. The CRC length provided must be a value from 1 to 8 where 1 represents a CRC size of 32 16-bit words and 8 represents a CRC size of 4096 16-bit words. The source address specifies the starting address for the CRC and the destination address is the location that the resulting CRC value is stored. The source and destination memories must be configured for the same zone. Additionally, the CRC length must not cross over the Flash sector or RAM block boundary. On the CM, there is an additional requirement that CRCLOCK is not enabled. Any violations of these requirements results in a failure status returned. Upon successful CRC, the number of 16-bit words CRC'd is returned.

Table 5-64. Secure CRC Calculation Function

CPU	Function Prototype	Function Parameters	Function Return Value
CPU1, CPU2, CM	Uint16 **SecureCRCCalcZ1**(Uint16 len_id, Uint16 *dst, Uint16 *src) Uint16 **SecureCRCCalcZ2**(Uint16 size, Uint16 *dst, Uint16 *src)	*len_id* : A number from 1 to 8 which corresponds to length options of 32, 64, 128, 256, 512, 1024, 2048, or 4096 16-bit words *dst* : The destination memory address for resulting CRC *src* : The source memory address to begin CRC calculation	0xXXXX : Returns the number of 16-bit words CRC'd 0x0000 : Indicates one of the following: Invalid length option; Source address is not modulo of length value; Destination address is not within secure RAM; CRC size crosses over Flash sector or RAM block boundary; The source and destination memory do not belong to the same zone; On CM, CRCLOCK is enabled

The **calculate CMAC (Cipher-based Message Authentication Code) function** calculates a CMAC tag for a specified memory range using the user-set CMAC key in OTP and returns pass or failure depending if the calculated tag matches the golden tag. The memory address range provided must align to a 128-bit boundary (split evenly into 128-bit blocks). If this requirement is not met, the function returns a status indicating a boundary violation. When using the CM CMAC function, there is an additional requirement that the CM must be running in privileged mode.

For generating the secure Flash golden CMAC tag for CPU1 or CPU2, refer to the section "Using Secure Flash Boot on TMS320F2838x Devices" in the *TMS320C28x Assembly Language Tools User's Guide* for instructions.

For generating the secure Flash golden CMAC tag for CM, refer to the ARM Assembly Language Tools v19.6.0.STS, within section "Using Secure Flash Boot on TMS320F2838x Devices" for instructions.

The 128-bit golden CMAC tag:

- Must be stored inside of the memory address range that the calculation is performed on.
- Another golden CMAC tag (from a different memory address range that is being authenticated) can not be nested inside a different CMAC authentication memory address range. (For example, a CMAC on addresses 0x1000 to 0x2000 can not contain the golden CMAC tag for memory address ranges 0x4000 to 0x5000).
- The starting address of the golden CMAC tag must align to a 32-bit boundary, such as 0x80002 on CPU1/CPU2 or 0x200004 on the CM.
- The CMAC calculation treats the memory addresses containing the golden tag as all ones.

Note

If calling this function, without running the secure Flash boot mode, then a dummy load must be performed for the Z1 OTP CMACKEY before calling the function. Additionally, the Flash data caching can be disabled before performing the dummy load and then the Flash data caching can be re-enabled after the dummy load.

Table 5-65. Secure Flash CMAC Calculation Function

CPU	Function Prototype	Function Parameters	Function Return Value
CPU1	uint32_t **CPU1BROM_calculateCMAC**(uint32_t startAddress, uint32_t endAddress, uint32_t tagAddress)	*startAddress* : The starting memory address for the calculation (Example: 0x80000) *endAddress* : The ending memory address for the calculation (Example: 0x82000) *tagAddress* : The starting memory address of where the golden CMAC tag is stored. (Example: 0x80002)	0xFFFFFFFF : The calculated CMAC tag does not match the golden tag (failure) 0xA5A5A5A5 : The memory address range is not aligned to a 128-bit boundary or length is zero 0x00000000 : The calculated CMAC tag matches the golden tag (pass)
CPU2	uint32_t **CPU2BROM_calculateCMAC**(uint32_t startAddress, uint32_t endAddress, uint32_t tagAddress)	*startAddress* : The starting memory address for the calculation (Example: 0x80000) *endAddress* : The ending memory address for the calculation (Example: 0x82000) *tagAddress* : The starting memory address of where the golden CMAC tag is stored. (Example: 0x80002)	0xFFFFFFFF : The calculated CMAC tag does not match the golden tag (failure) 0xA5A5A5A5 : The memory address range is not aligned to a 128-bit boundary or length is zero 0x00000000 : The calculated CMAC tag matches the golden tag (pass)

Table 5-65. Secure Flash CMAC Calculation Function (continued)

CPU	Function Prototype	Function Parameters	Function Return Value
CM	uint32_t **CMBROM_calculateCMAC**(uint32_t startAddress, uint32_t endAddress, uint32_t tagAddress)	*startAddress* : The starting memory address for the calculation (Example: 0x200000) *endAddress* : The ending memory address for the calculation (Example: 0x200000) *tagAddress* : The starting memory address of where the golden CMAC tag is stored. (Example: 0x200004)	0xFFFFFFFF : The calculated CMAC tag does not match the golden tag (failure) 0xA5A5A5A5 : The memory address range is not aligned to a 128-bit boundary or length is zero 0x5A5A5A5A : The CM is not running in privileged mode 0xE1E1E1E1: The CM AES engine timed out. AES engine can not be working as expected. 0x00000000 : The calculated CMAC tag matches the golden tag (pass)

5.7.10 Clock Initializations

During boot-up, the boot ROM initializes the device clocking, depending upon the reset source, to assist in faster boot time response. Clock configurations are performed by the boot ROM code only for POR and XRS reset types. For all other resets, the boot ROM starts executing with the clocks that were already set up before reset.

Note

Only CPU1 performs clock configurations during boot up. CPU1 application configures clocks for CPU2 and CM before releasing them from reset. Refer to Section 5.7.2 for more details.

If the PLL is used during the CPU1 boot process, it will be bypassed by the boot ROM code before branching to the user application.

Table 5-66. CPU1 Boot Clock Sources

Source	Frequency	Description
INTOSC2	10 MHz	Default clock source
INTOSC1	10 MHz	Set as clock source if missing clock is detected at power up or right after device reset
SYSPLL	110 MHz, 80 MHz, or 60 MHz	Enabled only as part of MPOST boot flow. PLL is bypassed and disabled after memory test has completed. See more details regarding enabling MPOST in Section 5.7.1.1.
SYSPLL and AUXPLL	SYSPLL = 180 MHz AUXPLL = 60 MHz	Enabled only as part of USB Bootloader. Both PLLs are bypassed and disabled after the bootloader actions complete.

Table 5-67. CPU1 Clock State After Boot

Reset Source	Clock State
POR/XRS	1. Using INTOSC2 2. System clock divider set to /1
All other Resets	Maintain clocks setup before device reset.

5.7.11 Boot Status information

Boot ROM keeps a record of the various actions and events that occur during boot ROM execution. The reason for this is because NMI and other exceptions are enabled by default in the device and must be handled accordingly. Boot ROM stores the boot status information in a RAM location so that the user application can read this boot status and take the necessary actions per application's needs to handle these events.

5.7.11.1 CPU1 Booting Status

CPU1 boot ROM health and booting status is written to a 32-bit address in M0RAM. This status is cleared on a POR or XRS reset. The previous status is retained on any other reset. For example, a user should clear the status before performing a debugger device reset in order to view the latest boot ROM actions reflected in the status.

Table 5-68. CPU1 Boot Status Address

Description	Address
CPU1 Boot ROM Status	0x0000 0002

Table 5-69. CPU1 Boot Status Bit Fields

Bit	Description
31	CPU1 Boot ROM has finished running
30	Missing clock NMI occurred
29	RAM Uncorrectable Error NMI or ROM Parity Error occurred
28	Flash Uncorrectable Error NMI occurred
27	HWBIST NMI occurred
26	PIE Vector NMI occurred
25	RL NMI occurred
24	PIE Mismatch occurred
23	ITRAP occurred
22	ERAD NMI occurred
21	EtherCAT NMI occurred
20	MCAN NMI occurred
19	SYSPLL or AUXPLL failed to enable
18	MPOST (Memory Power On Self-Test) Complete
17	RAM Initialization Complete
16	DCSM Initialization Complete
15	HWBIST Reset Handled
14	POR Reset Handled
13	XRS Reset Handled
12	All Resets Handled
11:8	Not Used
7:0	0x0 = Invalid / No Status set yet 0x1 = CPU1 Boot ROM has started running 0x2 = Running Flash Boot 0x3 = Running Secure Flash Boot 0x4 = Running Parallel Boot 0x5 = Running RAM Boot 0x6 = Running SCI Boot 0x7 = Running SPI Boot 0x8 = Running I2C Boot 0x9 = Running CAN Boot 0xA = Running USB Boot 0xB = Running Wait Boot

5.7.11.2 CPU2 Booting Status

CPU2 boot ROM health and booting status is written to a 32-bit address in M0RAM. This status is cleared on every CPU2 reset. Additionally, a copy of the status is written to CPU2TOCPU1IPCBOOTSTS for CPU1 to have access to CPU2's boot status.

Table 5-70. CPU2 Boot ROM Status Address

Description	Address
CPU2 Boot ROM Status	0x0000 0002

Table 5-71. CPU2 Boot Status Bit Fields

Bit	Description
31	CPU2 Boot ROM has finished running
30	Missing clock NMI occurred
29	RAM Uncorrectable Error NMI or ROM Parity Error occurred
28	Flash Uncorrectable Error NMI occurred
27	HWBIST NMI occurred
26	PIE Vector NMI occurred
25	RL NMI occurred
24	PIE Mismatch occurred
23	ITRAP occurred
22	ERAD NMI occurred
21	Secure Flash Boot CMAC returned failure
20	Not Used
19	Invalid length specified in CPU1TOCPU2IPCBOOTMODE for IPC message RAM copy length
18	Invalid (or missing configuration) in CPU1TOCPU2IPCBOOTMODE
17	RAM Initialization Complete
16	Not Used
15	HWBIST Reset Handled
14	POR Reset Handled
13	XRS Reset Handled
12	All Resets Handled
11:8	Not Used
7:0	0x0 = Invalid / No Status set yet 0x1 = CPU2 Boot ROM has started running 0x2 = Running Flash Boot 0x3 = Running Secure Flash Boot 0x4 = Running IPC Message Copy to RAM Boot 0x5 = Running RAM Boot 0x6 = Running User OTP Boot 0x7 = Running Wait Boot 0x8 = Waiting for CPU1 to set CPU1TOCPU2IPCFLG0 to allow CPU2 to start booting

5.7.11.3 CM Booting Status

CM boot ROM health and booting status is written to a 32-bit address in S0RAM. This status is cleared on every CM reset. Additionally, a copy of the status is written to CMTOCPU1IPCBOOTSTS for CPU1 to have access to the CM's boot status.

Table 5-72. CM Boot ROM Status Address

Description	Address
CM Boot ROM Status	0x2000 0000

Table 5-73. CM Boot Status Bit Fields

Bit	Description
31	CM Boot ROM has finished running
30	Missing clock NMI occurred
29	RAM Uncorrectable Error NMI or ROM Parity Error occurred
28	Flash Uncorrectable Error NMI occurred
27	MCAN NMI occurred
26	Windowed Watchdog NMI occurred
25	An EtherCAT NMI occurred
24	Not Used
23	Hard Fault occurred
22	Unassigned interrupt occurred
21	Secure Flash Boot CMAC returned failure
20	Not Used
19	Invalid length specified in CPU1TOCMIPCBOOTMODE for IPC message RAM copy length
18	Invalid (or missing configuration) in CPU1TOCMIPCBOOTMODE
17	RAM Initialization Complete
16:15	Not Used
14	POR Reset Handled
13	Not Used
12	All Resets Handled
11:8	Not Used
7:0	0x0 = Invalid / No Status set yet 0x1 = CM Boot ROM has started running 0x2 = Running Flash Boot 0x3 = Running Secure Flash Boot 0x4 = Running IPC Message Copy to RAM Boot 0x5 = Running RAM Boot 0x6 = Running User OTP Boot 0x7 = Running Wait Boot 0x8 = Waiting for CPU1 to set CPU1TOCMIPCFLG0 to allow CM to start booting

5.7.11.4 Boot Mode and MPOST (Memory Power On Self-Test) Status

Once the boot mode is decoded during the boot flow for each core, the boot mode value is written to RAM. Additionally, on CPU1, when running the MPOST POR memory test, the test result is written to RAM.

For more information, see the *C2000™ Memory Power-On Self-Test (M-POST) Application Report*.

Table 5-74. Boot Mode and MPOST Status Addresses

Description	Address
CPU1 Boot Mode	0x0000 0004
CPU2 Boot Mode	0x0000 0004
CM Boot Mode	0x2000 0004
MPOST Result (CPU1 Only)	0x0000 0006

5.7.12 ROM Version

The ROM revision and release date information is stored at the ROM locations specified in this section. Reading a revision number value of “0x100” represents version “1.0”, “0x101” represents version “1.1”, and so on. Reading a revision date value of “0x0916” represents “09/16” or “September 2016”.

Table 5-75. Boot ROM Version Information for CPU1

Start Address	End Address	Contents	Silicon Revision 0 Values	Silicon Revision A Values
0x003F FF7A	0x003F FF7A	Revision Number	0x0100	0x0200
0x003F FF7B	0x003F FF7B	Revision Date	0x0418	0x0819

Table 5-76. Boot ROM Version Information for CPU2

Start Address	End Address	Contents	Silicon Revision 0 Values	Silicon Revision A Values
0x003F FF7A	0x003F FF7A	Revision Number	0x0100	0x0100
0x003F FF7B	0x003F FF7B	Revision Date	0x0418	0x0418

Table 5-77. Boot ROM Version Information for CM

Start Address	End Address	Contents	Silicon Revision 0 Values	Silicon Revision A Values
0x0000 0140	0x0000 0141	Revision Number	0x0100	0x0200
0x0000 0142	0x0000 0143	Revision Date	0x0418	0x0819

5.8 Application Notes for Using the Bootloaders

5.8.1 Boot Data Stream Structure

This section details the data transfer protocols or stream structures that allow boot data transfer between boot ROM and host device. This data transfer protocol is compatible to the respective bootloaders on C2000 devices.

5.8.1.1 Bootloader Data Stream Structure

The following table and associated examples show the structure of the data stream incoming to the bootloader. The basic structure is the same for all the bootloaders and is based on the C54x source data stream generated by the C54x hex utility. The C28x hex utility (hex2000.exe) has been updated to support this structure. The hex2000.exe utility is included with the C2000 code generation tools. All values in the data stream structure are in hex. Refer to The C2000 Hex Utility for more details on using the C28x hex utility to convert a project to this format.

The first 16-bit word in the data stream is known as the key value. The key value is used to tell the bootloader the width of the incoming stream: 8 or 16 bits. Note that not all bootloaders accept both 8- and 16-bit streams. Refer to the detailed information on each loader for the valid data stream width. For an 8-bit data stream, the key value is 0x08AA and for a 16-bit stream the key value is 0x10AA. If a bootloader receives an invalid key value, then the load is aborted.

The next eight words are used to initialize register values or otherwise enhance the bootloader by passing values to the bootloader. If a bootloader does not use these values, then the values are reserved for future use and the bootloader simply reads the value and then discards the value. Currently only the SPI and I2C and parallel bootloaders use these words to initialize registers.

The tenth and eleventh words comprise the 22-bit entry point address. This address is used to initialize the PC after the boot load is complete. This address is most likely the entry point of the program downloaded by the bootloader.

The twelfth word in the data stream is the size of the first data block to be transferred. The size of the block is defined as 8-bit data stream format. For example, to transfer a block of 20 8-bit data values from an 8-bit data stream, the block size is 0x000A to indicate 10 16-bit words.

The next two words indicate to the loader the destination address of the block of data. Following the size and address is the 16-bit words that makeup that block of data.

This pattern of block size/destination address repeats for each block of data to be transferred. Once all the blocks have been transferred, a block size of 0x0000 signals to the loader that the transfer is complete. At this point, the loader returns the entry point address to the calling routine that exits. Execution continues at the entry point address as determined by the input data stream contents.

Table 5-78. LSB/MSB Loading Sequence in 8-Bit Data Stream

Byte		Contents	
		LSB (First Byte of 2)	MSB (Second Byte of 2)
1	2	LSB: AA (KeyValue for memory width = 8 bits)	MSB: 08h (KeyValue for memory width = 8 bits)
3	4	LSB: Register initialization value or reserved	MSB: Register initialization value or reserved
5	6	LSB: Register initialization value or reserved	MSB: Register initialization value or reserved
7	8	LSB: Register initialization value or reserved	MSB: Register initialization value or reserved
...	...	...	...
...	...	...	...
17	18	LSB: Register initialization value or reserved	MSB: Register initialization value or reserved
19	20	LSB: Upper half of Entry point PC[23:16]	MSB: Upper half of entry point PC[31:24] (Always 0x00)
21	22	LSB: Lower half of Entry point PC[7:0]	MSB: Lower half of Entry point PC[15:8]
23	24	LSB: Block size in words of the first block to load. If the block size is 0, this indicates the end of the source program; otherwise, another block follows. For example, a block size of 0x000A indicates 10 words or 20 bytes in the block.	MSB: block size
25	26	LSB: MSW destination address, first block Addr[23:16]	MSB: MSW destination address, first block Addr[31:24]
27	28	LSB: LSW destination address, first block Addr[7:0]	MSB: LSW destination address, first block Addr[15:8]
29	30	LSB: First word of the first block being loaded	MSB: First word of the first block being loaded
...	...	...	...
...	...	...	...
.	.	LSB: Last word of the first block to load	MSB: Last word of the first block to load
.	.	LSB: Block size of the second block	MSB: Block size of the second block
.	.	LSB: MSW destination address, second block Addr[23:16]	MSB: MSW destination address, second block Addr[31:24]
.	.	LSB: LSW destination address, second block Addr[7:0]	MSB: LSW destination address, second block Addr[15:8]
.	.	LSB: First word of the second block being loaded	MSB: First word of the second block being loaded
...	...	...	...
...	...	...	...
.	.	LSB: Last word of the second block	MSB: Last word of the second block
.	.	LSB: Block size of the last block	MSB: Block size of the last block
.	.	LSB: MSW of destination address of last block Addr[23:16]	MSB: MSW destination address, last block Addr[31:24]
.	.	LSB: LSW destination address, last block Addr[7:0]	MSB: LSW destination address, last block Addr[15:8]
.	.	LSB: First word of the last block being loaded	MSB: First word of the last block being loaded
...	...	...	...
...	...	...	...
.	.	LSB: Last word of the last block	MSB: Last word of the last block
n	n+1	LSB: 00h	MSB: 00h - indicates the end of the source

Example 5-2. Data Stream Structure 8-bit

```
AA 08        ; 0x08AA 8-bit key value
00 00 00 00  ; 8 reserved words
00 00 00 00
00 00 00 00
00 00 00 00
3F 00 00 80  ; 0x003F8000 EntryAddr, starting point after boot load completes
05 00        ; 0x0005 - First block consists of 5 16-bit words
3F 00 10 90  ; 0x003F9010 - First block will be loaded starting at 0x3F9010
01 00        ; Data loaded = 0x0001 0x0002 0x0003 0x0004 0x0005
02 00
03 00
04 00
05 00
02 00        ; 0x0002 - 2nd block consists of 2 16-bit words
3F 00 00 80  ; 0x003F8000 - 2nd block will be loaded starting at 0x3F8000
00 77        ; Data loaded = 0x7700 0x7625
25 76
00 00        ; 0x0000 - Size of 0 indicates end of data stream
After load has completed the following memory values will have been initialized as follows:
Location Value
0x3F9010 0x0001
0x3F9011 0x0002
0x3F9012 0x0003
0x3F9013 0x0004
0x3F9014 0x0005
0x3F8000 0x7700
0x3F8001 0x7625
PC Begins execution at 0x3F8000
```

5.8.2 The C2000 Hex Utility

To use the features of the bootloader, you must generate a data stream and boot table as described in Section 5.8.1.1. The hex conversion utility tool, included with the 28x code generation tools, can generate the required data stream including the required boot table. This section describes the hex2000 utility. An example of a file conversion performed by hex2000 is described in Example 5-3.

The hex utility supports creation of the boot table required for the SCI, SPI, I2C, CAN, and parallel I/O loaders. That is, the hex utility adds the required information to the file such as the key value, reserved bits, entry point, address, block start address, block length and terminating value. The contents of the boot table vary slightly depending on the boot mode and the options selected when running the hex conversion utility. The actual file format required by the host (ASCII, binary, hex, and so on) differs from one specific application to another and some additional conversion may be required.

To build the boot table, follow these steps:

1. **Assemble or compile the code.**

   This creates the object files that then is used by the linker to create a single output file.
2. **Link the file.**

   The linker combines all of the object files into a single output file in common object file format (ELF). The specified linker command file is used by the linker to allocate the code sections to different memory blocks. Each block of the boot table data corresponds to an initialized section in the ELF file. Uninitialized sections are not converted by the hex conversion utility. The following options can be useful:

The linker -m option can be used to generate a map file. This map file shows all of the sections that were created, their location in memory, and their length. It can be useful to check this file to make sure that the initialized sections are where you expect them to be.

The linker -w option configures the linker to show if the linker assigned a section to a memory region automatically. For example, if you have a section in your code called .TI.ramfunc.

3. **Run the hex conversion utility.**

   Choose the appropriate options for the desired boot mode and run the hex conversion utility to convert the ELF file produced by the linker to a boot table.

See the *TMS320C28x Assembly Language Tools User's Guide* and the *TMS320C28x Optimizing C/C++ Compiler User's Guide* for more information on the compiling and linking process.

Table 5-79 summarizes the hex conversion utility options available for the bootloader. See the *TMS320C28x Assembly Language Tools User's Guide* for a detailed description of the hex2000 operations used to generate a boot table. Updates are made to support the I2C boot. See the Codegen release notes for the latest information.

Table 5-79. Boot Loader Options

Option	Description
-boot	Convert all sections into bootable form (use instead of a SECTIONS directive)
-sci8	Specify the source of the bootloader table as the SCI-A port, 8-bit mode
-spi8	Specify the source of the bootloader table as the SPI-A port, 8-bit mode
-gpio8	Specify the source of the bootloader table as the GPIO port, 8-bit mode
-bootorg value	Specify the source address of the bootloader table
-lospcp value	Specify the initial value for the LOSPCP register. This value is used only for the spi8 boot table format and ignored for all other formats. If the value is greater than 0x7F, the value is truncated to 0x7F.
-spibrr value	Specify the initial value for the SPIBRR register. This value is used only for the spi8 boot table format and ignored for all other formats. If the value is greater than 0x7F, the value is truncated to 0x7F.
-e value	Specify the entry point at which to begin execution after boot loading. The value can be an address or a global symbol. This value is optional. The entry point can be defined at compile time using the linker -e option to assign the entry point to a global symbol. The entry point for a C program is normally _c_int00 unless defined otherwise by the -e linker option.
-i2c8	Specify the source of the bootloader table as the I2C-A port, 8-bit
-i2cpsc value	Specify the value for the I2CPSC register. This value is loaded and takes effect after all I2C options are loaded, prior to reading data from the EEPROM. This value is truncated to the least-significant eight bits and must be set to maintain an I2C module clock of 7-12 MHz.
-i2cclkh value	Specify the value for the I2CCLKH register. This value is loaded and takes effect after all I2C options are loaded, prior to reading data from the EEPROM.
-i2cclkl value	Specify the value for the I2CCLKL register. This value is loaded and takes effect after all I2C options are loaded, prior to reading data from the EEPROM.

Example 5-3. HEX2000.exe Command Syntax

```
C: HEX2000 GPIO34TOG.OUT -boot -gpio8 -a
Where:
- boot Convert all sections into bootable form.
- gpio8 Use the GPIO in 8-bit mode data format. The eCAN uses the same data format as the GPIO in
8-bit mode.
- a Select ASCII-Hex as the output format.
```

5.9 Software

5.9.1 BOOT Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/boot

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

5.9.1.1 CM Secure Flash Boot

FILE: boot_ex1_cpu1_cpu2_cm_secure_flash_cm.c

This example demonstrates how to use the secure flash boot mode for CM. (Requires CPU1 example application)

Secure flash boot performs a CMAC authentication on the entry sector of flash upon device boot up. If authentication passes, the application will begin execution. Learn more on the secure flash boot mode in the device technical reference manual.

This project shows how to use the C2000 HEX Utility to generate a CMAC Tag based on a user CMAC key and embed the value into the flash application. Additionally, the example details the method to call the CMAC API from the user application to calculate CMAC on other flash sectors beyond the the application entry flash sector.

Determining Pass/Fail without debugger connected: CM - ControlCARD LED3.
- LED off = Secure Boot failed
- LED On (Solid) = Secure Boot Passed, Full Flash CMAC failed
- LED Blinking = Secure Boot Passed and Full Flash CMAC passed

External Connections
- None.

Watch Variables
- None.

5.9.1.2 CPU1 Secure Flash Boot

FILE: boot_ex1_cpu1_cpu2_cm_secure_flash_cpu1.c

This example demonstrates how to use the secure flash boot mode for CPU1 as well as release CPU2 and CM for secure flash boot.

Secure flash boot performs a CMAC authentication on the entry sector of flash upon device boot up. If authentication passes, the application will begin execution. Learn more on the secure flash boot mode in the device technical reference manual.

This project shows how to use the C2000 HEX Utility to generate a CMAC Tag based on a user CMAC key and embed the value into the flash application. Additionally, the example details the method to call the CMAC API from the user application to calculate CMAC on other flash sectors beyond the the application entry flash sector.

How to Run:
- Load application into CPU1 flash (as well as CPU2 and CM applications)
- Disconnect and reconnect to only CPU1
- In memory window, set address 0xD00/D01 to 0x5AFFFFFF and address 0xD04 to 0x000A (This sets emulation boot to secure flash boot)
- Reset CPU1 via CCS and click resume
- Observe the LEDs

Determining Pass/Fail without debugger connected: *CPU1* - ControlCARD LED1.
- LED off = Secure Boot failed
- LED On (Solid) = Secure Boot Passed, Full Flash CMAC failed
- LED Blinking = Secure Boot Passed and Full Flash CMAC passed *CPU2* - ControlCARD LED2.
- LED off = Secure Boot failed

- LED On (Solid) = Secure Boot Passed, Full Flash CMAC failed
- LED Blinking = Secure Boot Passed and Full Flash CMAC passed *CM* - ControlCARD LED3.
- LED off = Secure Boot failed
- LED On (Solid) = Secure Boot Passed, Full Flash CMAC failed
- LED Blinking = Secure Boot Passed and Full Flash CMAC passed

External Connections

- None.

Watch Variables

- cpu1_SuccessfullyBooted - True when CPU1 full flash CMAC authentication passes. Otherwise, false.
- cpu2_SuccessfullyBooted - True when CPU2 full flash CMAC authentication passes and CPU1 receives IPC. Otherwise, false.
- cm_SuccessfullyBooted - True when CM full flash CMAC authentication passes and CPU1 receives IPC. Otherwise, false.

5.9.1.3 CPU2 Secure Flash Boot

FILE: boot_ex1_cpu1_cpu2_cm_secure_flash_cpu2.c

This example demonstrates how to use the secure flash boot mode for CPU2. (Requires CPU1 example application)

Secure flash boot performs a CMAC authentication on the entry sector of flash upon device boot up. If authentication passes, the application will begin execution. Learn more on the secure flash boot mode in the device technical reference manual.

This project shows how to use the C2000 HEX Utility to generate a CMAC Tag based on a user CMAC key and embed the value into the flash application. Additionally, the example details the method to call the CMAC API from the user application to calculate CMAC on other flash sectors beyond the the application entry flash sector.

Determining Pass/Fail without debugger connected: CPU2 - ControlCARD LED2.

- LED off = Secure Boot failed
- LED On (Solid) = Secure Boot Passed, Full Flash CMAC failed
- LED Blinking = Secure Boot Passed and Full Flash CMAC passed

External Connections

- None.

Watch Variables

- None.

This page intentionally left blank.

Chapter 6

Dual Code Security Module (DCSM)

This chapter explains the dual code security module.

6.1 Introduction

The dual code security module (DCSM) is a security feature incorporated in this device. It prevents access and visibility to on-chip secure memories (and other secure resources) by unauthorized persons. It also prevents duplication and reverse-engineering of proprietary code. The term “secure” means that access to on-chip secure memories and resources is blocked. The term “unsecure” means that access is allowed; that is, the contents of the memory could be read by any means (for example, through a debugging tool such as Code Composer Studio™ IDE.

There are two security zones, Zone1 (Z1) and Zone2 (Z2). Unlike earlier C2000 devices where each CPU subsystem had two security zones, on this device, both security zones are shared by each CPU subsystem. This means secure resources from each CPU subsystem are allocated to Zone1 or Zone2. All the security configurations are controlled by the CPU1 subsystem only (programmed in CPU1 USER OTP). Other CPU subsystems have only read access to these configurations via their own memory map registers.

6.1.1 DCSM Related Collateral

Getting Started Materials

- *C2000 DCSM Security Tool Application Report*
- *C2000 Unique Device Number Application Report*
- *Enhancing Device Security by Using JTAGLOCK Feature Application Report*
- *Secure BOOT On C2000 Device Application Report*

6.2 Functional Description

The security module restricts the CPU access to on-chip secure memory and resources without interrupting or stalling CPU execution. When a read occurs to a secure memory location, the read returns a zero value and CPU execution continues with the next instruction. This, in effect, blocks read and write access to secure memories through the JTAG port or external peripherals.

The code security mechanism offers protection for two zones, Zone1 (Z1) and Zone2 (Z2). The security mechanism for both the zones is identical. Each zone has its own dedicated secure resource and allocated secure resource. The following are different secure resources available on this device:

- **OTP:** Each zone has its own dedicated secure OTP (USER OTP) on CPU1 subsystem. This contains the security configurations for the individual zone. If a zone is secure, its USER OTP content (including CSM passwords) can be read (execution not allowed) only if the zone is unlocked using the password match flow (PMF).
- **RAM:** All Dx and LSx RAMs on C28x and C0/C1 RAMs on the Connectivity Manager(CM) can be secure RAM on this device. On this device IPC MSG RAMs between different subsystems can also be configured to be secure RAM. This enables secure message exchange between CPU subsystems. These RAMs can be allocated to either zone by configuring the respective GRABRAM locations in the CPU1 USER OTP.
- **Flash Sectors:** Flash sectors of each CPU subsystems can be made secure on this device. Each Flash sector can be allocated to either zone by configuring the respective GRABSECT locations in the CPU1 USER OTP.
- **Secure ROM:** This device also has secure ROM on each CPU subsystem which is EXEONLY-protected. These ROM contains specific function for the user, provided by TI.

Table 6-1 shows the status of a RAM block/Flash sector based on the configuration in the GRABRAM/ GRABSECT register.

The security of each zone is ensured by its own 128-bit (four 32-bit words) password (CSM password). The password for each zone is stored in CPU1 USER OTP. A zone can be unsecured by executing the password match flow (PMF), described in Section 6.7.4.

Table 6-1. RAM/Flash Status

Zone 1 GRAMRAMx/GRABSECTx Bits	Zone 2 GRAMRAMx/GRABSECT Bits	Ownership and Accessibility
01	10	RAM block/Flash Sector belongs to Zone1
01	11(2)	RAM block/Flash Sector belongs to Zone1
10	01	RAM block/Flash Sector belongs to Zone2
11(1)	01	RAM block/Flash Sector belongs to Zone2
10	10	RAM block/Flash Sector is unsecure
11	11	If both zones are unsecure, then RAM block/Flash is unsecure. RAM block/Flash Sector inaccessible if either of the zone is secure (CSM passwords are programmed). User should never leave these values default (11) if CSM passwords are programmed for even one zone.

(1) Zone1 must be unsecure. Assumption in this case is that user is not using Zone1 so none of the fields, including passwords, in Zone1 USER OTP are programmed by user hence Zone1 will always be unsecure.
(2) Zone2 must be unsecure. Assumption in this case is that user is not using Zone2 so none of the fields, including passwords, in Zone2 USER OTP are programmed by user hence Zone2 will always be unsecure.

Note

You should never program any other values in these fields. Failing any these conditions for a RAM block/Flash sector will make that RAM block/Flash sector inaccessible.

There are three types of accesses:

- Data/program reads: Data reads to a secure memory are always blocked unless the program is executing from a memory that belongs to the same zone. Data reads to unsecure memory are always allowed.
- JTAG access: JTAG accesses are always blocked when a memory is secure.
- Instruction fetches (calls, jumps, code executions, ISRs): Instruction fetches are never blocked.

Table 6-2 shows the levels of security.

Table 6-2. Security Levels

PMF Executed With Correct Password?	Operating Mode of the Zone	Program Fetch Location	Security Description
No	Secure	Outside secure memory	Only instruction fetches by the CPU are allowed to secure memory. In other words, code can still be executed, but not read.
No	Secure	Inside secure memory	CPU has full access (except for EXEONLY memories where read is not allowed). JTAG port cannot read the secured memory contents.
Yes	Unsecure	Anywhere	Full access for CPU and JTAG port to secure memory of that zone.

6.2.1 CSM Passwords

Unlike earlier C2000™ devices, on this device ALL_1 value (0xFFFFFFFF,0xFFFFFFFF,0xFFFFFFFF,0xFFFFFFFF) for CSM password for a zone does not unsecure the zone. Instead, if for any zone the CSM password values get loaded as ALL_1 from USER OTP, the device will be in BLOCKED state. Due to this reason TI will program a few bits in the second 32-bit password value (ZxOTP_CSMPSWD1) in every zone select block of each zone with value '0'. The default value for this password location is chosen in a manner that the respective ECC value remains ALL_1. Due to this, the CSMPSWD1 value programmed by TI for every zone select block is different. See Table 6-3 for ZxOTP_CSMPSWD1 value, programmed by TI on every device. Since ECC is not programmed, the user will be able to change this value by flipping the bits which are '1' to '0' but leaving the ones which are already programmed by TI as '0'. BOOTROM code will write the default password value into the KEYx register to unlock the device as part of device initialization sequence.

If the password locations of a zone have all 128 bits as zeros (ALL_0), that zone becomes permanently secure (LOCKED state), regardless of the contents of the CSMKEYx registers which means the zone cannot be unlocked using PMF, the password match flow described in Section 6.7.4. Therefore, the user should never use ALL_0 as password. A password of ALL_0 will prevent debug of secure code or reprogramming the Flash sectors. CSMKEYx registers are user-accessible registers that are used to unsecure the zones.

Table 6-3. Default Value of ZxOTP (Programmed by TI)

Zone Select Block	Zone1 USER OTP		Zone2 USER OTP	
	Address	Value	Address	Value
JLM_ENABLE (JTAGLOCK)	0x00078006	0xffff000f	NA	NA
PSWDLOCK	0x00078010	0xfb7fffff	0x00078210	0x1f7fffff
CRCLOCK	0x00078012	0x7fffffff	0x00078212	0x3fffffff
JTAGPSWDH0	0x00078014	0x4bffffff	NA	NA
JTAGPSWDH1	0x00078016	0x3fffffff	NA	NA
Zone_Select_Block0	0x00078022 (CSMPSWD1)	0x4d7fffff	0x00078222 (CSMPSWD1)	0x1f7fffff
Zone_Select_Block0	0x0007803e (JTAGPSWDL1)	0x2bffffff	NA	NA
Zone_Select_Block1	0x00078042 (CSMPSWD1)	0x5f7fffff	0x00078242 (CSMPSWD1)	0xe57fffff
Zone_Select_Block1	0x0007805e (JTAGPSWDL1)	0x27ffffff	NA	NA
Zone_Select_Block2	0x00078062 (CSMPSWD1)	0x1dffffff	0x00078262 (CSMPSWD1)	0x4fffffff
Zone_Select_Block2	0x0007807e (JTAGPSWDL1)	0x7b7fffff	NA	NA
Zone_Select_Block3	0x00078082 (CSMPSWD1)	0xaf7fffff	0x00078282 (CSMPSWD1)	0xe37fffff
Zone_Select_Block3	0x0007809e (JTAGPSWDL1)	0xc9ffffff	NA	NA
Zone_Select_Block4	0x000780a2 (CSMPSWD1)	0x1bffffff	0x000782a2 (CSMPSWD1)	0x57ffffff
Zone_Select_Block4	0x000780be (JTAGPSWDL1)	0x7d7fffff	NA	NA
Zone_Select_Block5	0x000780c2 (CSMPSWD1)	0x17ffffff	0x000782c2 (CSMPSWD1)	0x5bffffff
Zone_Select_Block5	0x000780de (JTAGPSWDL1)	0x6f7fffff	NA	NA
Zone_Select_Block6	0x000780e2 (CSMPSWD1)	0xbd7fffff	0x000782e2 (CSMPSWD1)	0xf17fffff
Zone_Select_Block6	0x000780fe (JTAGPSWDL1)	0x33ffffff	NA	NA
Zone_Select_Block7	0x00078102 (CSMPSWD1)	0x9f7fffff	0x00078302 (CSMPSWD1)	0x3b7fffff
Zone_Select_Block7	0x0007811e (JTAGPSWDL1)	0x0fffffff	NA	NA
Zone_Select_Block8	0x00078122 (CSMPSWD1)	0x2bffffff	0x00078322 (CSMPSWD1)	0x8fffffff
Zone_Select_Block8	0x0007813e (JTAGPSWDL1)	0xbb7fffff	NA	NA
Zone_Select_Block9	0x00078142 (CSMPSWD1)	0x27ffffff	0x00078342 (CSMPSWD1)	0x6bffffff
Zone_Select_Block9	0x0007815e (JTAGPSWDL1)	0x5f7fffff	NA	NA
Zone_Select_Block10	0x00078162 (CSMPSWD1)	0x7b7fffff	0x00078362 (CSMPSWD1)	0x377fffff
Zone_Select_Block10	0x0007817e (JTAGPSWDL1)	0x1dffffff	NA	NA
Zone_Select_Block11	0x00078182 (CSMPSWD1)	0xc9ffffff	0x00078382 (CSMPSWD1)	0x9bffffff
Zone_Select_Block11	0x0007819e (JTAGPSWDL1)	0xaf7fffff	NA	NA
Zone_Select_Block12	0x000781a2 (CSMPSWD1)	0x7d7fffff	0x000783a2 (CSMPSWD1)	0x2f7fffff
Zone_Select_Block12	0x000781be (JTAGPSWDL1)	0x1bffffff	NA	NA
Zone_Select_Block13	0x000781c2 (CSMPSWD1)	0x6f7fffff	0x000783c2 (CSMPSWD1)	0xcb7fffff
Zone_Select_Block13	0x000781de (JTAGPSWDL1)	0x17ffffff	NA	NA
Zone_Select_Block14	0x000781e2 (CSMPSWD1)	0x33ffffff	0x000783e2 (CSMPSWD1)	0x97ffffff
Zone_Select_Block14	0x000781fe (JTAGPSWDL1)	0xbd7fffff	NA	NA

6.2.2 Emulation Code Security Logic (ECSL)

In addition to the CSM, the emulation code security logic (ECSL) has been implemented using a 64-bit password (part of existing CSM password) for each zone to prevent unauthorized users from stepping through secure code. A halt in secure code while the emulator is connected will trip the ECSL and break the emulation connection. To allow emulation of secure code, while maintaining the CSM protection against secure memory reads, the user must write the correct 64-bit password into the CSMKEY (0/1) registers, which matches the password value stored in the USER OTP of that zone. This will disable the ECSL for the specific zone.

When initially debugging a device with the password locations in OTP programmed (secured), the emulator takes some time to take control of the CPU. During this time, the CPU will start running and may execute an instruction that performs an access to a protected ECSL area and if the CPU is halted when the program counter (PC) is pointing to a secure location, the ECSL will trip and cause the emulator connection to be broken.

The solution to this problem is:

- Use the Wait Boot Mode boot option. In this mode, the CPU will be in a loop and hence will not jump to the user application code. Using this BOOTMODE, the user can connect to CCS and debug the code.

6.2.3 CPU Secure Logic

The CPU Secure Logic (CPUSL) on this device prevents a hacker from reading the CPU registers in a watch window while code is running in a secure zone. All accesses to CPU registers when the PC points to a secure location are blocked by this logic. The only exception to this is read access to the PC. It is highly recommended not to write into the CPU register in this case, because proper code execution may get affected. If the CSM is unlocked using the CSM password match flow, the CPUSL logic also gets disabled.

6.2.4 Execute-Only Protection

To achieve a higher level of security on secure Flash sectors and RAM blocks that store critical user code (instruction opcodes), the Execute-Only protection feature is provided. When the Execute-Only protection is turned on for any secure Flash sector or RAM block, data reads to that Flash sector or RAM block are disallowed from any code (even from secure code). Execute-only protection for a Flash sector and RAM block can be turned on by configuring the bit field associated for that particular sector/RAM block in the zone's (which has ownership of that sector/RAM block) EXEONLYSECT and EXEONLYRAM register, respectively.

6.2.5 Password Lock

The password locations in USER OTP for each zone can be locked by programming the zone’s PSWDLOCK field with any value other than “1111” (0xF) at the PSWDLOCK location in OTP. Until the passwords of a zone are locked, password locations will not be secure and can be read from the debugger as well as code running from non-secure memory. This feature can be used by the user to avoid accidental locking of the zone while programming the Flash sectors during the software development phase. On a fresh device the value for password lock fields for all zones at the PSWDLOCK location in OTP will be “1111” which means the password for all zones will be unlocked.

Note

Password unlock only makes password locations non-secure. All other secure memories remains secure as per security settings. Since password locations are non-secure, anyone can read the password and make the zone un-secure by running through PMF, user must program PSWDLOCK locations to lock the password before sending the device in field.

6.2.6 JTAGLOCK

Sometimes you want to disable the JTAG access on a device to avoid any debug access to it. This can be done by using the JTAGLOCK feature on this device. You need to follow a two step process to enable the JTAGLOCK feature (both steps can be performed at the same time).

1. Program the JTAG passwords. This device has a 128-bit JTAG password that needs to be programmed in Z1 USER OTP of CPU1. JTAG passwords are split into two parts, JTAGPSWDH and JTAGPSWDL. JTAGPSWDH is part of the Z1 USER OTP header and JTAGPSWDL is part of the Z1 Zone Select Block (ZSB). What this means is program JTAGPSWDH once and change the JTAGPSWDL multiple times, if needed. The Code Composer Studio™ IDE has an integrated tool that you need to use to unlock the JTAGLOCK on the device.
2. After programming the JTAG passwords, you need to enable the JTAGLOCK module (JLM) by programming bit [3:0] of Z1OTP_JLM_ENABLE with any value other than 0xF. It is recommended to program all four bits with a value 0x0.

For more details on how to enable and disable the JTAGLOCK feature, refer to the *Enhancing Device Security by Using JTAGLOCK Feature Application Report*.

6.2.7 Link Pointer and Zone Select

For each of the two security zones, a dedicated OTP block exists on CPU1 that holds the configuration related to zone's security. The following are user programmable configurations:

- ZxOTP_LINKPOINTER1
- ZxOTP_LINKPOINTER2
- ZxOTP_LINKPOINTER3
- Z1OTP_JLM_ENABLE
- ZxOTP_GPREG1
- ZxOTP_GPREG2
- ZxOTP_GPREG3
- ZxOTP_GPREG4
- ZxOTP_PSWDLOCK
- ZxOTP_CRCLOCK
- Z1OTP_JTAGPSWDH
- Z1OTP_CMACKEY
- ZxOTP_CSMPSWD0
- ZxOTP_CSMPSWD1
- ZxOTP_CSMPSWD2
- ZxOTP_CSMPSWD3
- ZxOTP_GRABSECT1
- ZxOTP_GRABSECT2
- ZxOTP_GRABSECT3
- ZxOTP_GRABRAM1
- ZxOTP_GRABRAM2
- ZxOTP_GRABRAM3
- ZxOTP_EXEONLYSECT1
- ZxOTP_EXEONLYSECT2
- ZxOTP_EXEONLYRAM1
- Z1OTP_JTAGPSWDL

Since OTP cannot be erased, the following configurations are placed in zone select blocks of each zone's OTP Flash of both the banks:

- ZxOTP_CSMPSWD0
- ZxOTP_CSMPSWD1
- ZxOTP_CSMPSWD2
- ZxOTP_CSMPSWD3
- ZxOTP_GRABSECT1
- ZxOTP_GRABSECT2
- ZxOTP_GRABSECT3
- ZxOTP_GRABRAM1
- ZxOTP_GRABRAM2
- ZxOTP_GRABRAM3
- ZxOTP_EXEONLYSECT1
- ZxOTP_EXEONLYSECT2
- ZxOTP_EXEONLYRAM1
- Z1OTP_JTAGPSWDL

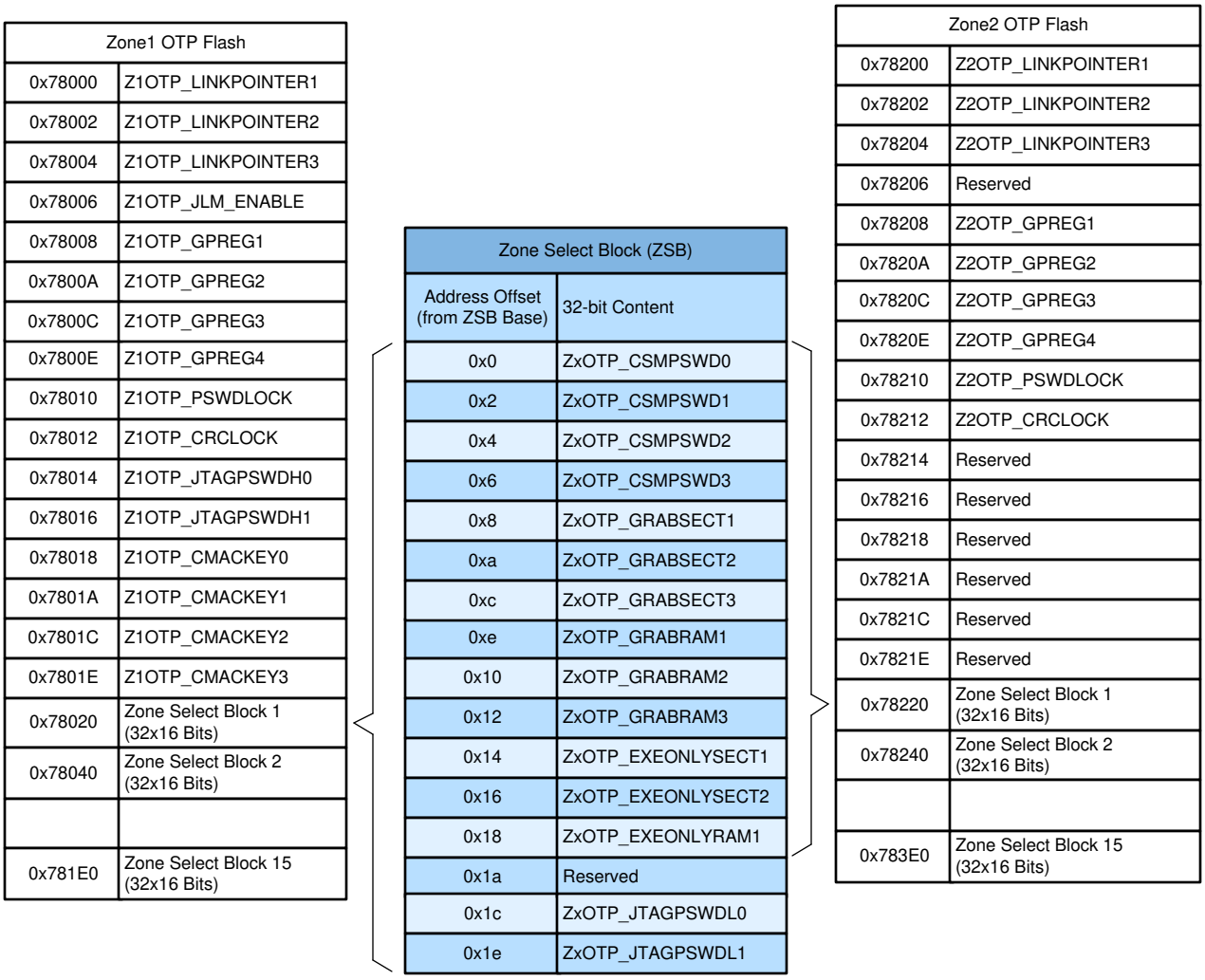
The location of the valid zone select block in OTP is decided based on the value of three 14-bit link pointers (Zx-LINKPOINTERx) programmed in the OTP of each zone. All OTP locations except link pointers and Z1OTP_JLM_ENABLE locations are protected with ECC. Since the link pointer locations are not protected with ECC, three link pointers are provided that need to be programmed with the same value. The final value of the link pointer is resolved in hardware, when a dummy read is done to all the link pointers, by comparing all the three values (bit-wise voting logic). Since in OTP, a '1' can be flipped by the user to '0' but '0' can not be flipped to '1' (no erase operation for OTP), the most significant bit position in the resolved link pointer which is '0', defines the valid base address for the zone select block. While generating the final link pointer value, if the bit pattern is not one of those listed in Figure 6-1, the final link pointer value becomes All_1 (0xFFFF_FFFF), which selects the Zone-Select-Block1 (also known as the default zone select block).

Zx-LINKPOINTER	Selected ZSB	Zone1 ZSB Address	Zone2 ZSB Address
32xxxxxxxxxxxxxxxx11111111111111	ZSB1	0x78020	0x78220
32xxxxxxxxxxxxxxxx11111111111110	ZSB2	0x78040	0x78240
32xxxxxxxxxxxxxxxx11111111111100	ZSB3	0x78060	0x78260
32xxxxxxxxxxxxxxxx11111111111000	ZSB4	0x78080	0x78280
32xxxxxxxxxxxxxxxx11111111110000	ZSB5	0x780a0	0x782a0
32xxxxxxxxxxxxxxxx11111111100000	ZSB6	0x780c0	0x782c0
32xxxxxxxxxxxxxxxx11111111000000	ZSB7	0x780e0	0x782e0
32xxxxxxxxxxxxxxxx11111110000000	ZSB8	0x78100	0x78300
32xxxxxxxxxxxxxxxx11111100000000	ZSB9	0x78120	0x78320
32xxxxxxxxxxxxxxxx11111000000000	ZSB10	0x78140	0x78340
32xxxxxxxxxxxxxxxx11110000000000	ZSB11	0x78160	0x78360
32xxxxxxxxxxxxxxxx11100000000000	ZSB12	0x78180	0x78380
32xxxxxxxxxxxxxxxx11000000000000	ZSB13	0x781a0	0x783a0
32xxxxxxxxxxxxxxxx10000000000000	ZSB14	0x781c0	0x783c0
32xxxxxxxxxxxxxxxx00000000000000	ZSB15	0x781e0	0x783e0

Figure 6-1. Storage of Zone-Select Bits in OTP

Note

Address locations for other security settings that are not part of Zone Select blocks can be programmed only once; therefore, you can program the blocks towards the end of the development cycle.



Zone1 OTP Flash	
0x78000	Z1OTP_LINKPOINTER1
0x78002	Z1OTP_LINKPOINTER2
0x78004	Z1OTP_LINKPOINTER3
0x78006	Z1OTP_JLM_ENABLE
0x78008	Z1OTP_GPREG1
0x7800A	Z1OTP_GPREG2
0x7800C	Z1OTP_GPREG3
0x7800E	Z1OTP_GPREG4
0x78010	Z1OTP_PSWDLOCK
0x78012	Z1OTP_CRCLOCK
0x78014	Z1OTP_JTAGPSWDH0
0x78016	Z1OTP_JTAGPSWDH1
0x78018	Z1OTP_CMACKEY0
0x7801A	Z1OTP_CMACKEY1
0x7801C	Z1OTP_CMACKEY2
0x7801E	Z1OTP_CMACKEY3
0x78020	Zone Select Block 1 (32x16 Bits)
0x78040	Zone Select Block 2 (32x16 Bits)
0x781E0	Zone Select Block 15 (32x16 Bits)

Zone Select Block (ZSB)	
Address Offset (from ZSB Base)	32-bit Content
0x0	ZxOTP_CSMPSWD0
0x2	ZxOTP_CSMPSWD1
0x4	ZxOTP_CSMPSWD2
0x6	ZxOTP_CSMPSWD3
0x8	ZxOTP_GRABSECT1
0xa	ZxOTP_GRABSECT2
0xc	ZxOTP_GRABSECT3
0xe	ZxOTP_GRABRAM1
0x10	ZxOTP_GRABRAM2
0x12	ZxOTP_GRABRAM3
0x14	ZxOTP_EXEONLYSECT1
0x16	ZxOTP_EXEONLYSECT2
0x18	ZxOTP_EXEONLYRAM1
0x1a	Reserved
0x1c	ZxOTP_JTAGPSWDL0
0x1e	ZxOTP_JTAGPSWDL1

Zone2 OTP Flash	
0x78200	Z2OTP_LINKPOINTER1
0x78202	Z2OTP_LINKPOINTER2
0x78204	Z2OTP_LINKPOINTER3
0x78206	Reserved
0x78208	Z2OTP_GPREG1
0x7820A	Z2OTP_GPREG2
0x7820C	Z2OTP_GPREG3
0x7820E	Z2OTP_GPREG4
0x78210	Z2OTP_PSWDLOCK
0x78212	Z2OTP_CRCLOCK
0x78214	Reserved
0x78216	Reserved
0x78218	Reserved
0x7821A	Reserved
0x7821C	Reserved
0x7821E	Reserved
0x78220	Zone Select Block 1 (32x16 Bits)
0x78240	Zone Select Block 2 (32x16 Bits)
0x783E0	Zone Select Block 15 (32x16 Bits)

Figure 6-2. Location of Zone-Select Block Based on Link-Pointer

CAUTION

USER OTP is ECC protected. You must program the ECC value while programming the security setting in USER OTP. Failing to program the correct ECC value causes the device to be blocked permanently and you have to replace the device.

6.2.8 C Code Example to Get Zone Select Block Addr for Zone1

```
unsigned long LinkPointer;
unsigned long *Zone1SelBlockPtr;
int Bitpos = 13;
int ZeroFound = 0;
// Read Z1-Linkpointer register of DCSM module.
LinkPointer = *(unsigned long *)0x5F000;
// Bits 31 to 15 as most-sigificant 0 are reserved LinkPointer options
LinkPointer = LinkPointer << 18;
while ((ZeroFound == 0) && (bitpos > -1))
{
if ((LinkPointer & 0x80000000) == 0)
{
ZeroFound = 1;
Zone1SelBlockPtr = (unsigned long *)(0x78000 + ((bitpos + 2)*32));
}
else
{
bitpos--;
LinkPointer = LinkPointer << 1;
}
}
if (ZeroFound == 0)
{
//Default in case there is no zero found.
Zone1SelBlockPtr = (unsigned long *)0x78020;
}
```

6.3 Flash and OTP Erase/Program

On this device, OTP as well as normal Flash, are secure resources. Each zone has its own dedicated CPU1 OTP, whereas normal Flash sectors can be allocated to any zone based on the value programmed in the GRABSECTx location in OTP. Each zone has its own 128bit CSM passwords. Read and write accesses are not allowed to resources assigned to Z1 by code running from memory allocated to Z2 and vice versa. Before programming any secure Flash sector the user must either unlock the zone to which that particular sector belongs using PMF or execute the Flash programming code from secure memory which belongs to the same zone. The same is the case for erasing any secure Flash sector. To program the security settings in CPU1 OTP Flash, the user must unlock the CSM of the respective zone. Unless the zone is unlocked, security settings in OTP Flash can not be updated. The OTP content cannot be erased.

A semaphore mechanism is provided to avoid the program/erase conflict between Z1 and Z2. A zone needs to grab this semaphore to successfully complete the erase/program operation on the secure Flash sectors allocated to that zone. A semaphore can be grabbed by a zone by writing the appropriate value in the SEM field of the FLSEM register. For further details of this field, see the register description.

6.4 Secure Copy Code

In some applications, the user may want to copy the code from secure Flash to secure RAM for better performance. The user cannot do this for EXEONLY Flash sectors because EXEONLY secure memories cannot be read from anywhere. TI provides specific “Secure Copy Code” library functions for each zone to enable the user to copy content from EXEONLY secure Flash sectors to EXEONLY RAM blocks. These functions do the copy-code operation in a highly secure environment and allow a copy to be performed only when the following conditions are met:

- The secure RAM block and the secure Flash sector belong to the same zone.
- Both the secure RAM block and the secure Flash sector have EXEONLY protection enabled.

For further usage of these library functions, see the device-specific Boot ROM documentation.

6.5 SecureCRC

Since reads from EXEONLY memories are not allowed, you cannot calculate the CRC on content in EXEONLY memories using the CRC engine available on this device (for example, VCRC, GCRC) or software. In some safety-critical applications, the user may have to calculate the CRC even on these memories. To enable this without compromising on security, TI provides specific “SecureCRC” library functions for each zone. These functions do the CRC calculation in highly secure environment and allow a CRC calculation to be performed only when the following conditions are met:

- The source address should be modulo the number of words (based on length_id) for which the CRC needs to be calculated.
- The destination address should belong to the same zone as the source address.

For further usage of these library functions, see the device-specific Boot ROM documentation.

Note

You must disable all the interrupts before calling the secure functions in ROM. If there is a vector fetch during secure function execution, the CPU gets reset immediately.

Disclaimer: The Code Security Module (CSM) included on this device was designed to password protect the data stored in the associated memory and is warranted by Texas Instruments (TI), in accordance with its standard terms and conditions, to conform to TI's published specifications for the warranty period applicable for this device. TI DOES NOT, HOWEVER, WARRANT OR REPRESENT THAT THE CSM CANNOT BE COMPROMISED OR BREACHED OR THAT THE DATA STORED IN THE ASSOCIATED MEMORY CANNOT BE ACCESSED THROUGH OTHER MEANS. MOREOVER, EXCEPT AS SET FORTH ABOVE, TI MAKES NO WARRANTIES OR REPRESENTATIONS CONCERNING THE CSM OR OPERATION OF THIS DEVICE, INCLUDING ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL TI BE LIABLE FOR ANY CONSEQUENTIAL, SPECIAL, INDIRECT, INCIDENTAL, OR PUNITIVE DAMAGES, HOWEVER CAUSED, ARISING IN ANY WAY OUT OF YOUR USE OF THE CSM OR THIS DEVICE, WHETHER OR NOT TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. EXCLUDED DAMAGES INCLUDE, BUT ARE NOT LIMITED TO LOSS OF DATA, LOSS OF GOODWILL, LOSS OF USE OR INTERRUPTION OF BUSINESS OR OTHER ECONOMIC LOSS.

6.6 CSM Impact on Other On-Chip Resources

On this device, some of the memories are not secure. To avoid any potential hacking when the device is in the default state (post reset), accesses (all types) to all memories (secure as well as non-secure, except BOOT-ROM and OTP) are disabled until proper security initialization is done. This means that after reset none of the memory resources except BOOT_ROM and OTP is accessible to the user.

The following steps are required by CPU1 after reset (any type of reset) to initialize the security on device.

Security Initialization

- Dummy Read to address location of SECDC (0x703F0, TI-reserved register) in TI OTP
- Dummy Read to address location of Z1OTP_LINKPOINTER1 in Z1 OTP
- Dummy Read to address location of Z1OTP_LINKPOINTER2 in Z1 OTP
- Dummy Read to address location of Z1OTP_LINKPOINTER3 in Z1 OTP
- Dummy Read to address location of Z2OTP_LINKPOINTER1 in Z2 OTP
- Dummy Read to address location of Z2OTP_LINKPOINTER2 in Z2 OTP
- Dummy Read to address location of Z2OTP_LINKPOINTER3 in Z2 OTP
- Dummy Read to address location Z1OTP_JLM_ENABLE in Z1 OTP
- Dummy Read to address location of Z1OTP_GPREG1, Z1OTP_GPREG2, Z1OTP_GPREG3, Z1OTP_GPREG4 in Z1 OTP
- Dummy Read to address location of Z1OTP_PSWDLOCK in Z1 OTP
- Dummy Read to address location of Z1OTP_CRCLOCK in Z1 OTP

- Dummy Read to address location of Z1OTP_JTAGPSWDH0, Z1OTP_JTAGPSWDH1 in Z1 OTP
- Dummy Read to address location of Z2OTP_GPREG1, Z2OTP_GPREG2, Z2OTP_GPREG3, Z2OTP_GPREG4 in Z2 OTP
- Dummy Read to address location of Z2OTP_PSWDLOCK in Z2 OTP
- Dummy Read to address location of Z2OTP_CRCLOCK in Z2 OTP
- Read to memory map register of Z1_LINKPOINTER in DCSM module to calculate the address of zone select block for Z1
- Dummy read to address location of Z1OTP_GRABSECT1, Z1OTP_GRABSECT2, Z1OTP_GRABSECT3 in Z1 OTP
- Dummy read to address location of Z1OTP_GRABRAM1, Z1OTP_GRABRAM2, Z1OTP_GRABRAM3 in Z1 OTP
- Dummy read to address location of Z1OTP_EXEONLYSECT1, Z1OTP_EXEONLYSECT2 in Z1 OTP
- Dummy read to address location of Z1OTP_EXEONLYRAM1 in Z1 OTP
- Dummy Read to address location of Z1OTP_JTAGPSWDL0, Z1OTP_JTAGPSWDL1 in Z1 OTP
- Read to memory map register of Z2_LINKPOINTER in DCSM module to calculate the address of zone select block for Z2
- Dummy read to address location of Z2OTP_GRABSECT1, Z2OTP_GRABSECT2, Z2OTP_GRABSECT3 in Z2 OTP
- Dummy read to address location of Z2OTP_GRABRAM1, Z2OTP_GRABRAM2, Z2OTP_GRABRAM3 in Z2 OTP
- Dummy read to address location of Z2OTP_EXEONLYSECT1, Z2OTP_EXEONLYSECT2 in Z2 OTP
- Dummy read to address location of Z2OTP_EXEONLYRAM1 in Z2 OTP

Note

Security Initialization is done by CPU1 BOOTROM code on all the resets (as part of device initialization) which assert CPU1 SYSRSn. This will not be part of user application code

The order of initialization matters hence if a memory watch window with the USER OTP address is opened in the debugger (CCS) the security initialization could occur in an incorrect order, locking the device down. To avoid this, user should not keep a memory window with USER OTP address opened in the debugger(CCS) when performing a reset.

6.7 Incorporating Code Security in User Applications

Code security is typically not required in the development phase of a project. However, security is needed once a robust code is developed for a zone. Before such a code is programmed in the Flash memory, a CSM password should be chosen to secure the zone. Once a CSM password is in place for a zone, the zone is secured (programming a password at the appropriate locations and either performing a device reset or setting the FORCESEC bit (Zx_CR.31) is the action that secures the device). From that time on, access to debug the contents of secure memory by any means (via JTAG, code running off external/on-chip memory, and so forth) requires a valid password. A password is not needed to run the code out of secure memory (such as in a typical end-user usage); however, access to secure memory contents for debug purposes, requires a password.

6.7.1 Environments That Require Security Unlocking

The following are the typical situations under which unsecuring the zone can be required:

- Code development using debuggers (such as Code Composer Studio). This is the most common environment during the design phase of a product.
- Flash programming using TI's Flash utilities such as Code Composer Studio On-Chip Flash Programmer plug-in or the Uniflash tool. Flash programming is common during code development and testing. Once the user supplies the necessary password, the Flash utilities disable the security logic before attempting to program the Flash. In custom programming solutions that use the Flash API supplied by TI, unlocking the CSM can be avoided by executing the Flash programming algorithms from secure memory.
- Custom environment defined by the application
  In addition to the above, access to secure memory contents can be required in situations such as:
  - Using the on-chip bootloader to load code or data into secure SARAM or to erase and program the Flash.
  - Executing code from on-chip unsecure memory and requiring access to secure memory for the lookup table. This is not a suggested operating condition as supplying the password from external code could compromise code security.

The unsecuring sequence is identical in all the above situations. This sequence is referred to as the password match flow (PMF) for simplicity. Figure 6-3 explains the sequence of operation that is required every time the user attempts to unsecure a particular zone. A code example is listed for clarity.

6.7.2 CSM Password Match Flow

Password match flow (PMF) is essentially a sequence of four dummy reads from password locations (PWL) followed by four writes (32-bit writes) to CSMKEY(0/1/2/3) registers. Figure 6-3 shows how PMF helps to initialize the security logic registers and disable security logic.

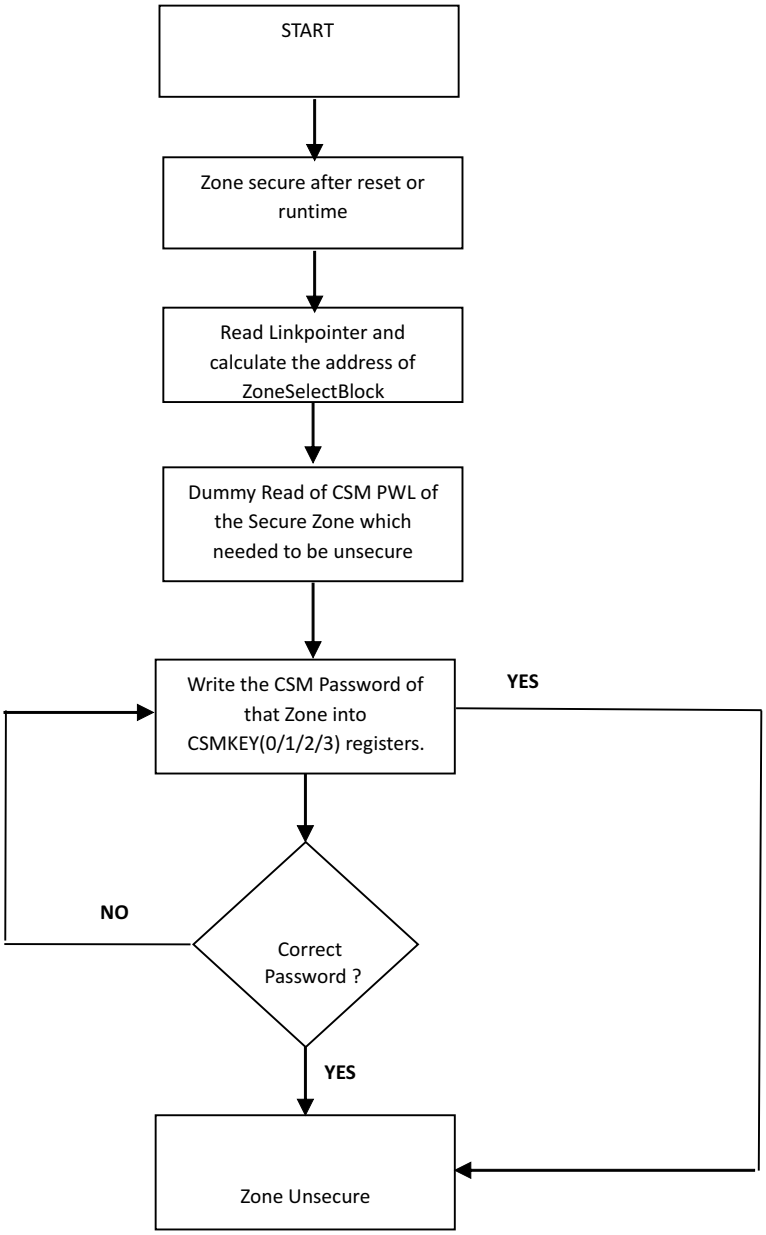


Figure 6-3. CSM Password Match Flow (PMF)

6.7.3 C Code Example to Unsecure C28x Zone1

```
volatile long int *CSM = (volatile long int *)5F010; //CSM register file volatile
long int *CSMPWL = (volatile long int *)0x78020; //CSM Password location (assuming default Zone
select block)
volatile int tmp;
int I;
// Read the 128-bits of the CSM password locations (PWL)
//
for (I=0;I<4; I++) tmp = *CSMPWL++;
// Write the 128-bit password to the CSMKEY registers
// If this password matches that stored in the
// CSLPWL then the CSM will become unsecure. If it does not
// match, then the zone will remain secure.
// An example password of: // 0x1111222233334444555566667777888 is used.
*CSM++ = 0x22221111; // Register Z1_CSMKEY0 at 0x5F010
*CSM++ = 0x44443333; // Register Z1_CSMKEY1 at 0x5F012
*CSM++ = 0x66665555; // Register Z1_CSMKEY2 at 0x5F014
*CSM++ = 0x88887777; // Register Z1_CSMKEY3 at 0x5F016
```

6.7.4 C Code Example to Resecure C28x Zone1

```
volatile int *Z1_CR = 0x5F019; //CSMSCR register
//Set FORCESEC bit
*Z1_CR = 0x8000;
```

Note

User must use the FORCESEC feature to resecure the zone from same subsystem that has unlocked the zone. For example, if CM subsystem has unlocked the Zone1 by entering the CSM password in CSMKEYx ,then only CM subsystem should resecure the Zone1 using FORCESEC feature.

6.7.5 Environments That Require ECSL Unlocking

The following are the typical situations under which unsecuring can be required:

- The user develops some main IP, and then outsources peripheral functions to a subcontractor who must be able to run the user code during debug and may halt while main IP code is running. If ECSL is not unlocked, then Code Composer Studio connections will get disconnected, which can be inconvenient for the user. Note that unlocking ECSL does not enable access to secure code but only avoids disconnection of CCS (JTAG).

6.7.6 ECSL Password Match Flow

A password match flow (PMF) is essentially a sequence of eight dummy reads from password locations (PWL) followed by two writes to KEY registers. Figure 6-4 shows how the PMF helps to initialize the security logic registers and disable security logic.

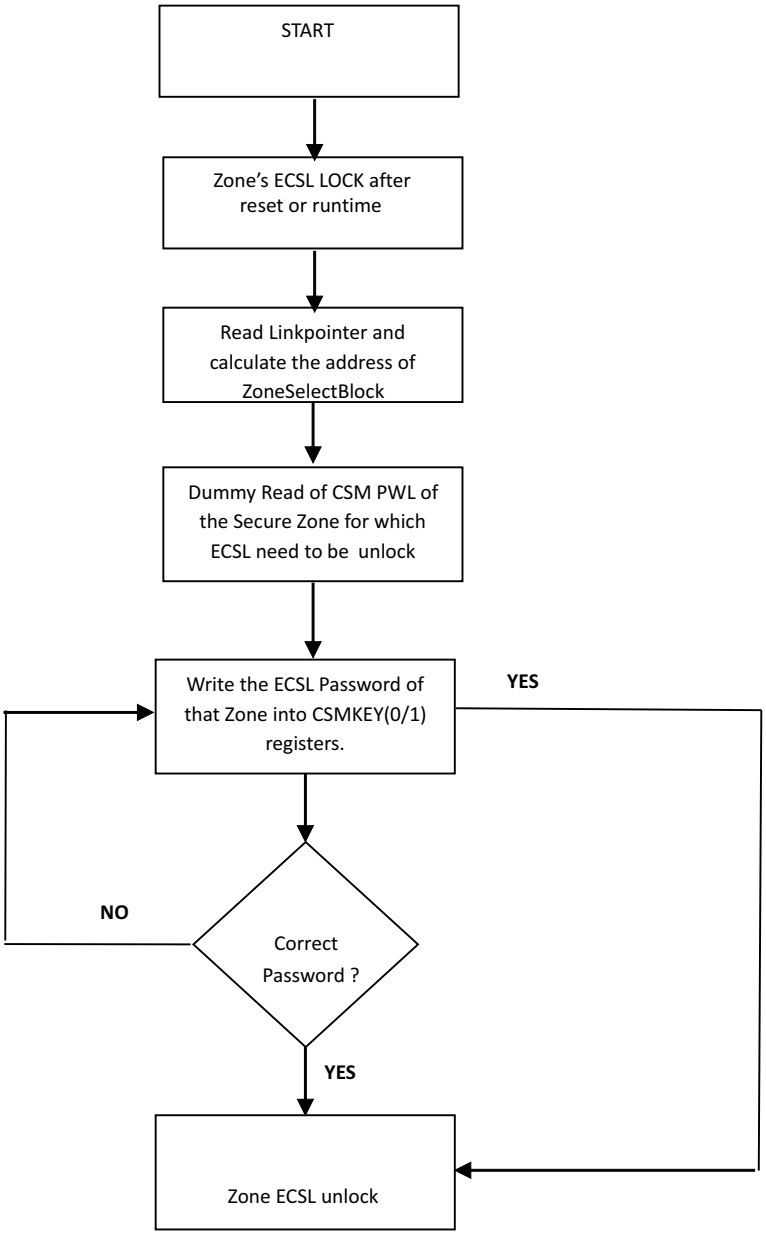


Figure 6-4. ECSL Password Match Flow (PMF)

6.7.7 ECSL Disable Considerations for any Zone

A zone with ECSL enabled should have a predetermined ECSL password stored in the ECSL password locations in Flash (same as lower 64 bits of CSM passwords). The following are steps to disable the ECSL for any particular zone:

- Perform a dummy read of CSM password locations of that Zone.
- Write the password into the CSMKEY0/1 registers, corresponding to that Zone.
- If the password is correct, the ECSL gets disabled; otherwise, it stays enabled.

6.7.7.1 C Code Example to Disable ECSL for C28x-Zone1

```
volatile long int *ECSL = (volatile int *)0x5F010; //ECSL register file
volatile long int *ECSLPWL = (volatile int *)0x78028; //ECSL Password location (assuming default
Zone sel block)
volatile int tmp;
int I;
// Read the 64-bits of the password locations (PWL).
for (I=0;I<2; I++) tmp = *ECSLPWL++;
// Write the 64-bit password to the CSMKEYx registers
// If this password matches that stored in the
// CSMPWL then ECSL will get disable. If it does not
// match, then the zone will remain secure.
// An example password of: // 0x1111222233334444 is used.
*ECSL++ = 0x22221111; // Register Z1_CSMKEY0 at 0x5F010
*ECSL++ = 0x44443333; // Register Z1_CSMKEY1 at 0x5F012
```

Note

If the CM subsystem is out of reset when ECSL is unlocked by any of the subsystem, one must reset the CM before trying to lock the ECSL again. Unless CM is reset, ECSL can not be locked again by entering the incorrect KEY or using the FORCESEC.

6.7.8 Device Unique ID

CPU1 TI OTP contains a 256-bit value that is made up of both random and sequential parts. This value can be used as a seed for code encryption. The starting address of the value is 0x7020C. The first 192 bits are random, the next 32 bits are sequential, and the last 32 bits are a checksum value.

6.8 Software

6.8.1 DCSM Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/dcsm

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

6.8.1.1 Empty DCSM Tool Example

FILE: dcsm_security_tool.c This example is an empty project setup for DCSM Tool and Driverlib development. For guidance refer to: *C2000 DCSM Security Tool*

6.8.1.2 DCSM Memory Access control by master CPU1 - C28X_CM

FILE: dcsm_ex1_cpu_access_control.c

This example demonstrates how to configure the 1st Zone Select Block in the OTP to allocate CM's C0RAM to zone 1 & CM's C1RAM to zone 2, later accessed by CM. Zone1 | Zone2 | CM's C0RAM | CM's C1RAM | In this example, zoning of memories is done by the OTP programming whose values are configured in dcsm_ex1_f2838x_dcsm_zxotp.asm while the securing functionalities are done through this file. It demonstrates how to control the access of the memories which would later be accessed by CM. This would even do a dummy read of the password needed by CM to unsecure the memory. The communication between the 2 CPUs are done using IPC (Inter process communication) through a synch function. This enables the CPU Core to wait until the expected task is completed on the other core.

External Connections

- None.

Watch Variables

- *result* - Status of Memory Access control by CPU1
- *set_error* - Count of errors occurring during the execution of the example.

Before running the example, the below configuration is expected to be done through the dcsm_ex1_f2838x_dcsm_zxotp.asm :

- Allocate CM's C0RAM to zone 1 , C1RAM to zone 2 ZSBx_Z1_GRABRAM2R 0x0AAAAA09 ZSBx_Z2_GRABRAM2R 0x0AAAAA06
- Password of zone 1 is 0xFFFFFFFF4D7FFFFFFFFFFFFFFFFFFFFF
- Password of zone 2 is 0xFFFFFFFF1F7FFFFFFFFFFFFFFFFFFFFF

6.8.1.3 DCSM Memory Access by CPU2 - C28X_DUAL

FILE: dcsm_ex1_cpu2_memory_access.c

This example demonstrates how the access of the memory is affected when the memories are secured by CPU1. CPU1 allocate CPU2's LS4-LS5 to zone 1 & CPU2's LS6-LS7 to zone 2 using the 1st Zone Select Block. Zone1 | Zone2 | CPU2's LS4-LS5 | CPU2's LS6-LS7 | It writes some data in the zones and checks after the CPU1 does a memory locking and matches with the data set . Further, once the CPU2 unlocks the memories, it matches with the data set written before CPU1 lock. Ideally after locking, zone1 should not be readable(or reads a 0 value) and zone2 that is not secured matches the written data set. It demonstrates how to lock and and unlock zone by showing where to put the password and how to check if it is secured or unsecured.

The communication between the 2 CPUs are handled using IPC (Inter process communication) through a synch function. This enables the CPU Core to wait until the expected task is completed on the other core.

External Connections

- None.

Watch Variables

- *result* - Status of CPU2's secure memory access

- *set_error*, error_not_locked ,error_not_unlocked ,error1 - Count of errors occurring during the execution of the example.
- *Zone1_Locked_Array* - Array demonstrating secured memory
- *Unsecure_mem_Array* - Array demonstrating Unsecured memory

6.8.1.4 DCSM Memory Access control by CPU1 - C28X_DUAL

FILE: dcsm_ex1_cpu1_access_control.c

This example demonstrates how to configure the 1st Zone Select Block in the OTP needed to allocate CPU2's LS4-LS5 to zone 1 & CPU2's LS6-LS7 to zone 2, later accessed by CPU2. Zone1 | Zone2 | CPU2's LS4-LS5 | CPU2's LS6-LS7 | In this example, zoning of memories is done by the OTP programming whose values are configured in dcsm_ex1_f2838x_dcsm_zxotp.asm while the securing functionalities are done through this file. It demonstrates how to control the access of the memories which would later be accessed by CPU2. This would even do a dummy read of the password needed by CPU2 to unsecure the memory. The communication between the 2 CPUs are done using IPC (Inter process communication) through a sync function. This enables the CPU Core to wait until the expected task is completed on the other core.

External Connections

- None.

Watch Variables

- *result* - Status of Memory Access control by CPU1
- *set_error* - Count of errors occurring during the execution of the example.

Before running the example, the below configuration is expected to be done through the dcsm_ex1_f2838x_dcsm_zxotp.asm :

- Allocate CPU2's LS4-LS5 to zone 1 , LS6-LS7 to zone 2 ZSBx_Z1_GRABRAM3R 0x0000A500 ZSBx_Z2_GRABRAM3R 0x00005A00
- Password of zone 1 is 0xFFFFFFFF4D7FFFFFFFFFFFFFFFFFFFF
- Password of zone 2 is 0xFFFFFFFF1F7FFFFFFFFFFFFFFFFFFFF

6.8.1.5 DCSM Memory partitioning Example

FILE: dcsm_ex1_secure_memory_partition.c

This example demonstrates how to configure and use DCSM. It configures the 1st Zone Select Block in the OTP to change the zone passwords and allocates LS0-LS3 to zone 1 & LS4-LS7 to zone 2. Zone1 | Zone2 | LS0-LS3 | LS4-LS7 | In this example, zoning of memories is done by the OTP programming whose values are configured in dcsm_ex1_f2838x_dcsm_zxotp.asm while the securing functionalities are done through this file. It writes some data in the zones and checks before locking and after locking and matches with the data set . Ideally after locking zone1, the data set stored in zone1 should not be readable(or reads a 0 value) and zone2 that is not secured matches the written data set. It demonstrates how to lock and and unlock zones by showing where to put the password and how to check if it is secured or unsecured.

External Connections

- None.

Watch Variables

- *result* - Status of Secure memory partitioning done through OTP programming.
- *set_error*, error_not_locked ,error_not_unlocked ,error1 - Count of errors occurring during the execution of the example.
- *Zone1_Locked_Array* - Array demonstrating secured memory
- *Unsecure_mem_Array* - Array demonstrating Unsecured memory

Before running the example, the below configuration is expected to be done through the dcsm_ex1_f2838x_dcsm_zxotp.asm :

- Allocate LS0-LS3 to zone 1 , LS4-LS7 to zone 2 ZSBx_Z1_GRABRAM1R 0x000AAA55 ZSBx_Z2_GRABRAM1R 0x000A55AA
- Password of zone 1 is 0xFFFFFFFF4D7FFFFFFFFFFFFFFFFFFFF

- Password of zone 2 is 0xFFFFFFFF1F7FFFFFFFFFFFFFFFFFFFFF

DCSM_unlockZone*CSM function should not be called in an actual application, should only be used for once to program the OTP memory. Ensure flash data cache is disabled before calling this function.

6.8.1.6 DCSM Memory Access by CM - C28X_CM

FILE: dcsm_ex1_cm_memory_access.c

This example demonstrates how the access of the memory is affected when the memories are secured by CPU1. CPU1 allocate CM's C0RAM to zone 1 & CM's C1RAM to zone 2 using the 1st Zone Select Block. Zone1 | Zone2 | CM's C0RAM | CM's C1RAM | It writes some data in the zones and checks after the CPU1 does a memory locking and matches with the data set . Further, once the CM unlocks the memories, it matches with the data set written before CPU1 lock. Ideally after locking, zone1 should not be readable(or reads a 0 value) and zone2 that is not secured matches the written data set. It demonstrates how to lock and and unlock zone by showing where to put the password and how to check if it is secured or unsecured.

The communication between the 2 CPUs are handled using IPC (Inter process communication) through a sync function. This enables the CPU Core to wait until the expected task is completed on the other core.

External Connections

- None.

Watch Variables

- *result* - Status of CM's secure memory access
- *set_error*, error_not_locked ,error_not_unlocked ,error1 - Count of errors occurring during the execution of the example.
- *Zone1_Locked_Array* - Array demonstrating secured memory
- *Unsecure_mem_Array* - Array demonstrating Unsecured memory

6.9 DCSM Registers

This section describes the various DCSM Registers.

Note

Except for the SECERRSTAT, SECERRCLR, and SECERRFRC registers, all other registers (non-OTP space) are mapped on all three subsystems. For the CM subsystem, a x8 offset needs to be used.

6.9.1 DCSM Base Address Table (C28)

Table 6-4. DCSM Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
DcsmZ1Regs	DCSM_Z1_REGS	DCSM_Z1_BASE	0x0005_F000	YES	YES	-	-	YES
DcsmZ2Regs	DCSM_Z2_REGS	DCSM_Z2_BASE	0x0005_F080	YES	YES	-	-	YES
DcsmCommonRegs	DCSM_COMMON_REGS	DCSMCOMMON_BASE	0x0005_F0C0	YES	YES	-	-	YES
DcsmZ1OtpRegs	DCSM_Z1_OTP	DCSM_Z1OTP_BASE	0x0007_8000	YES	-	-	-	-
DcsmZ2OtpRegs	DCSM_Z2_OTP	DCSM_Z2OTP_BASE	0x0007_8200	YES	-	-	-	-

6.9.2 CM DCSM Base Address Table (CM)

Table 6-5. CM DCSM Base Address Table (CM)

DriverLib Name	Base Address	µDMA Access	Ethernet DMA Access
DCSM_Z1_BASE	0x4008_5000	-	-
DCSM_Z2_BASE	0x4008_5100	-	-
DCSMCOMMON_BASE	0x4008_5180	-	-

6.9.3 DCSM_Z1_REGS Registers

Table 6-6 lists the memory-mapped registers for the DCSM_Z1_REGS registers. All register offset addresses not listed in Table 6-6 should be considered as reserved locations and the register contents should not be modified.

Table 6-6. DCSM_Z1_REGS Registers

Offset (x8)	Offset (x16)	Acronym	Register Name	Write Protection	Section
0h	0h	Z1_LINKPOINTER	Zone 1 Link Pointer		Go
4h	2h	Z1_OTPSECLOCK	Zone 1 OTP Secure Lock		Go
8h	4h	Z1_JLM_ENABLE	Zone 1 JTAGLOCK Enable Register		Go
Ch	6h	Z1_LINKPOINTERERR	Link Pointer Error		Go
10h	8h	Z1_GPREG1	Zone 1 General Purpose Register-1		Go
14h	Ah	Z1_GPREG2	Zone 1 General Purpose Register-2		Go
18h	Ch	Z1_GPREG3	Zone 1 General Purpose Register-3		Go
1Ch	Eh	Z1_GPREG4	Zone 1 General Purpose Register-4		Go
20h	10h	Z1_CSMKEY0	Zone 1 CSM Key 0		Go
24h	12h	Z1_CSMKEY1	Zone 1 CSM Key 1		Go
28h	14h	Z1_CSMKEY2	Zone 1 CSM Key 2		Go
2Ch	16h	Z1_CSMKEY3	Zone 1 CSM Key 3		Go
30h	18h	Z1_CR	Zone 1 CSM Control Register		Go
34h	1Ah	Z1_GRABSECT1R	Zone 1 Grab Flash Status Register 1		Go
38h	1Ch	Z1_GRABSECT2R	Zone 1 Grab Flash Status Register 2		Go
3Ch	1Eh	Z1_GRABSECT3R	Zone 1 Grab Flash Status Register 3		Go
40h	20h	Z1_GRABRAM1R	Zone 1 Grab RAM Status Register 1		Go
44h	22h	Z1_GRABRAM2R	Zone 1 Grab RAM Status Register 2		Go
48h	24h	Z1_GRABRAM3R	Zone 1 Grab RAM Status Register 3		Go
4Ch	26h	Z1_EXEONLYSECT1R	Zone 1 Execute Only Flash Status Register 1		Go
50h	28h	Z1_EXEONLYSECT2R	Zone 1 Execute Only Flash Status Register 2		Go
54h	2Ah	Z1_EXEONLYRAM1R	Zone 1 Execute Only RAM Status Register 1		Go
5Ch	2Eh	Z1_JTAGKEY0	JTAG Unlock Key Register 0		Go
60h	30h	Z1_JTAGKEY1	JTAG Unlock Key Register 1		Go
64h	32h	Z1_JTAGKEY2	JTAG Unlock Key Register 2		Go
68h	34h	Z1_JTAGKEY3	JTAG Unlock Key Register 3		Go
6Ch	36h	Z1_CMACKEY0	Secure Boot CMAC Key Status Register 0		Go
70h	38h	Z1_CMACKEY1	Secure Boot CMAC Key Status Register 1		Go
74h	3Ah	Z1_CMACKEY2	Secure Boot CMAC Key Status Register 2		Go
78h	3Ch	Z1_CMACKEY3	Secure Boot CMAC Key Status Register 3		Go

Complex bit access types are encoded to fit into small table cells. Table 6-7 shows the codes that are used for access types in this section.

Table 6-7. DCSM_Z1_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write

Table 6-7. DCSM_Z1_REGS Access Type Codes (continued)

Access Type	Code	Description
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

6.9.3.1 Z1_LINKPOINTER Register (Offset (x8) = 0h, Offset (x16) = 0h) [Reset = FFFFC000h]

Z1_LINKPOINTER is shown in Figure 6-5 and described in Table 6-8.

Return to the Summary Table.

Zone 1 Link Pointer

Figure 6-5. Z1_LINKPOINTER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																		LINKPOINTER													
R-0h																		R-0h													

Table 6-8. Z1_LINKPOINTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13-0	LINKPOINTER	R	0h	This is resolved Link-Pointer value which is generated by looking at the three physical Link-Pointer values loaded from OTP Reset type: SYSRSn

6.9.3.2 Z1_OTPSECLOCK Register (Offset (x8) = 4h, Offset (x16) = 2h) [Reset = 1h]

Z1_OTPSECLOCK is shown in Figure 6-6 and described in Table 6-9.

Return to the Summary Table.

Zone 1 OTP Secure Lock

Figure 6-6. Z1_OTPSECLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				CRCLOCK			
R-0h				R-0h			
7	6	5	4	3	2	1	0
PSWDLOCK				RESERVED			JTAGLOCK
R-0h				R-0h			R-1h

Table 6-9. Z1_OTPSECLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-12	RESERVED	R	0h	Reserved
11-8	CRCLOCK	R	0h	Value in this field gets loaded from Z1_CRCLOCK[3:0] when a read is issued to address location of Z1_CRCLOCK in OTP. 1111 : VCU has ability to calculate CRC on secure memories. Other Value : VCU doesn't have ability to calculate CRC on secure memories. Reset type: XRSn
7-4	PSWDLOCK	R	0h	Value in this field gets loaded from Z1_PSWDLOCK[3:0] when a read is issued to address location of Z1_PSWDLOCK in OTP. 1111 : CSM password locations in OTP are not protected and can be read from debugger as well as code running from anywhere. Other Value : CSM password locations in OTP are protected and can't be read without unlocking CSM of that zone. Reset type: XRSn
3-1	RESERVED	R	0h	Reserved
0	JTAGLOCK	R	1h	Reflects the state of the JTAGLOCK feature. 0 : JTAG is not locked 1 : JTAG is locked Reset type: PORESETn

6.9.3.3 Z1_JLM_ENABLE Register (Offset (x8) = 8h, Offset (x16) = 4h) [Reset = Fh]

Z1_JLM_ENABLE is shown in Figure 6-7 and described in Table 6-10.

Return to the Summary Table.

Zone 1 JTAGLOCK Enable Register

Figure 6-7. Z1_JLM_ENABLE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				Z1_JLM_ENABLE			
R-0h				R-Fh			

Table 6-10. Z1_JLM_ENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-0	Z1_JLM_ENABLE	R	Fh	Zone1 JLM_ENABLE register. The value in this field gets loaded from Z1OTP_JLM_ENABLE[3:0] when a read is issued to address location of Z1OTP_JLM_ENABLE in OTP. If Z1OTP_JLM_ENABLE[31:0] is equal to all ones during the load, the JTAGLOCK is not bypassed (is enabled). If the value of Z1OTP_JLM_ENABLE[31:0] is not all ones during the load, the JTAGLOCK is governed as follows by the Z1_JLM_ENABLE bits: 1111 : JTAG/Emulation access is allowed (JTAGLOCK is not enabled) Other values: JTAGLOCK is governed by the JTAGKEY==JTAGPSWD match condition Reset type: PORESETn

6.9.3.4 Z1_LINKPOINTERERR Register (Offset (x8) = Ch, Offset (x16) = 6h) [Reset = 0h]

Z1_LINKPOINTERERR is shown in Figure 6-8 and described in Table 6-11.

Return to the Summary Table.

Link Pointer Error

Figure 6-8. Z1_LINKPOINTERERR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		Z1_LINKPOINTERERR													
R-0h		R-0h													

Table 6-11. Z1_LINKPOINTERERR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13-0	Z1_LINKPOINTERERR	R	0h	These bits indicate errors during formation of the resolved Link-Pointer value after the three physical Link-Pointer values loaded of OTP in flash. 0 : No Error. Other : Error on bit positions which is set to 1. Reset type: SYSRSn

6.9.3.5 Z1_GPREG1 Register (Offset (x8) = 10h, Offset (x16) = 8h) [Reset = 0h]

Z1_GPREG1 is shown in Figure 6-9 and described in Table 6-12.

Return to the Summary Table.

Zone 1 General Purpose Register-1

Figure 6-9. Z1_GPREG1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPREG1																															
R-0h																															

Table 6-12. Z1_GPREG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPREG1	R	0h	This field gets loaded with the contents of Z1OTP_GPREG1 locations in Zone-1-USER-OTP when a dummy read is issued to that address. Users can use this register to load any general purpose non-voliate content from USER-OTP of Zone-1 Reset type: SYSRSn

6.9.3.6 Z1_GPREG2 Register (Offset (x8) = 14h, Offset (x16) = Ah) [Reset = 0h]

Z1_GPREG2 is shown in Figure 6-10 and described in Table 6-13.

Return to the Summary Table.

Zone 1 General Purpose Register-2

Figure 6-10. Z1_GPREG2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPREG2																															
R-0h																															

Table 6-13. Z1_GPREG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPREG2	R	0h	This field gets loaded with the contents of Z1OTP_GPREG2 locations in Zone-1-USER-OTP when a dummy read is issued to that address. Users can use this register to load any general purpose non-voliate content from USER-OTP of Zone-1 Reset type: SYSRSn

6.9.3.7 Z1_GPREG3 Register (Offset (x8) = 18h, Offset (x16) = Ch) [Reset = 0h]

Z1_GPREG3 is shown in Figure 6-11 and described in Table 6-14.

Return to the Summary Table.

Zone 1 General Purpose Register-3

Figure 6-11. Z1_GPREG3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPREG3																															
R-0h																															

Table 6-14. Z1_GPREG3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPREG3	R	0h	This field gets loaded with the contents of Z1OTP_GPREG3 locations in Zone-1-USER-OTP when a dummy read is issued to that address. Users can use this register to load any general purpose non-voliate content from USER-OTP of Zone-1 Reset type: SYSRSn

6.9.3.8 Z1_GPREG4 Register (Offset (x8) = 1Ch, Offset (x16) = Eh) [Reset = 0h]

Z1_GPREG4 is shown in Figure 6-12 and described in Table 6-15.

Return to the Summary Table.

Zone 1 General Purpose Register-4

Figure 6-12. Z1_GPREG4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPREG4																															
R-0h																															

Table 6-15. Z1_GPREG4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPREG4	R	0h	This field gets loaded with the contents of Z1OTP_GPREG4 locations in Zone-1-USER-OTP when a dummy read is issued to that address. Users can use this register to load any general purpose non-voliate content from USER-OTP of Zone-1 Reset type: SYSRSn

6.9.3.9 Z1_CSMKEY0 Register (Offset (x8) = 20h, Offset (x16) = 10h) [Reset = 0h]

Z1_CSMKEY0 is shown in Figure 6-13 and described in Table 6-16.

Return to the Summary Table.

Zone 1 CSM Key 0

Figure 6-13. Z1_CSMKEY0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1_CSMKEY0																															
R/W-0h																															

Table 6-16. Z1_CSMKEY0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1_CSMKEY0	R/W	0h	To unlock Zone1, user needs to write this regsiter with exact value as Z1_CSMPSWD0, programmed in OTP (zone gets unlock only if 128 bit password in OTP match with value written in four CSMKEY registers.) Reset type: SYSRSn

6.9.3.10 Z1_CSMKEY1 Register (Offset (x8) = 24h, Offset (x16) = 12h) [Reset = 0h]

Z1_CSMKEY1 is shown in Figure 6-14 and described in Table 6-17.

Return to the Summary Table.

Zone 1 CSM Key 1

Figure 6-14. Z1_CSMKEY1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1_CSMKEY1																															
R/W-0h																															

Table 6-17. Z1_CSMKEY1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1_CSMKEY1	R/W	0h	To unlock Zone1, user needs to write this regsiter with exact value as Z1_CSMPSWD1, programmed in OTP (zone gets unlock only if 128 bit password in OTP match with value written in four CSMKEY registers.) Reset type: SYSRSn

6.9.3.11 Z1_CSMKEY2 Register (Offset (x8) = 28h, Offset (x16) = 14h) [Reset = 0h]

Z1_CSMKEY2 is shown in Figure 6-15 and described in Table 6-18.

Return to the Summary Table.

Zone 1 CSM Key 2

Figure 6-15. Z1_CSMKEY2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1_CSMKEY2																															
R/W-0h																															

Table 6-18. Z1_CSMKEY2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1_CSMKEY2	R/W	0h	To unlock Zone1, user needs to write this regsiter with exact value as Z1_CSMPSWD2, programmed in OTP (zone gets unlock only if 128 bit password in OTP match with value written in four CSMKEY registers.) Reset type: SYSRSn

6.9.3.12 Z1_CSMKEY3 Register (Offset (x8) = 2Ch, Offset (x16) = 16h) [Reset = 0h]

Z1_CSMKEY3 is shown in Figure 6-16 and described in Table 6-19.

Return to the Summary Table.

Zone 1 CSM Key 3

Figure 6-16. Z1_CSMKEY3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1_CSMKEY3																															
R/W-0h																															

Table 6-19. Z1_CSMKEY3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1_CSMKEY3	R/W	0h	To unlock Zone1, user needs to write this regsiter with exact value as Z1_CSMPSWD3, programmed in OTP (zone gets unlock only if 128 bit password in OTP match with value written in four CSMKEY registers.) Reset type: SYSRSn

6.9.3.13 Z1_CR Register (Offset (x8) = 30h, Offset (x16) = 18h) [Reset = 00080000h]

Z1_CR is shown in Figure 6-17 and described in Table 6-20.

Return to the Summary Table.

Zone 1 CSM Control Register

Figure 6-17. Z1_CR Register

31	30	29	28	27	26	25	24
FORCESEC	RESERVED						
R-0/W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED	ARMED	UNSECURE	ALLONE	ALLZERO	RESERVED		
R-0h	R-0h	R-0h	R-0h	R-1h	R-0h		
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	RESERVED	RESERVED
R-0h				R-0h	R-0h	R-0h	R-0h

Table 6-20. Z1_CR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	FORCESEC	R-0/W	0h	A write '1' to this fields resets the state of zone. If zone is unlocked, it'll lock(secure) the zone and also resets all the bits in this register. Reset type: SYSRSn
30-24	RESERVED	R	0h	Reserved
23	RESERVED	R	0h	Reserved
22	ARMED	R	0h	0 : Dummy read to CSM Password locations in OTP hasn't been performed. 1 : Dummy read to CSM Password locations in OTP has been performed. Reset type: SYSRSn
21	UNSECURE	R	0h	Indiacates the state of Zone. 0 : Zone is in lock(secure) state. 1 : Zone is in unlock(unsecure) state. Reset type: SYSRSn
20	ALLONE	R	0h	Indicates the state of CSM passwords. 0 : Zone CSM Passwords are not all ones. 1 : Zone CSM Passwords are all ones. Reset type: SYSRSn
19	ALLZERO	R	1h	Indicates the state of CSM passowrds. 0 : CSM Passwords are not all zeros. 1 : CSM Passwords are all zero and device is permanently locked. Reset type: SYSRSn
18-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

6.9.3.14 Z1_GRABSECT1R Register (Offset (x8) = 34h, Offset (x16) = 1Ah) [Reset = 0h]

Z1_GRABSECT1R is shown in Figure 6-18 and described in Table 6-21.

Return to the Summary Table.

Zone 1 Grab Flash Status Register 1

Figure 6-18. Z1_GRABSECT1R Register

31	30	29	28	27	26	25	24
RESERVED				GRAB_SECT13		GRAB_SECT12	
R-0h				R-0h		R-0h	
23	22	21	20	19	18	17	16
GRAB_SECT11		GRAB_SECT10		GRAB_SECT9		GRAB_SECT8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
GRAB_SECT7		GRAB_SECT6		GRAB_SECT5		GRAB_SECT4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
GRAB_SECT3		GRAB_SECT2		GRAB_SECT1		GRAB_SECT0	
R-0h		R-0h		R-0h		R-0h	

Table 6-21. Z1_GRABSECT1R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-26	GRAB_SECT13	R	0h	Value in this field gets loaded from Z1_GRABSECT1[27:26] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 13 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 13 to Zone1. 10 : No request for CPU1 Flash Sector 13 11 : No request for CPU1 Flash Sector 13 when this zone is UNLOCKED. Else CPU1 Flash Sector 13 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
25-24	GRAB_SECT12	R	0h	Value in this field gets loaded from Z1_GRABSECT1[25:24] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 12 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 12 to Zone1. 10 : No request for CPU1 Flash Sector 12 11 : No request for CPU1 Flash Sector 12 when this zone is UNLOCKED. Else CPU1 Flash Sector 12 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
23-22	GRAB_SECT11	R	0h	Value in this field gets loaded from Z1_GRABSECT1[23:22] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 11 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 11 to Zone1. 10 : No request for CPU1 Flash Sector 11 11 : No request for CPU1 Flash Sector 11 when this zone is UNLOCKED. Else CPU1 Flash Sector 11 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-21. Z1_GRABSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	GRAB_SECT10	R	0h	Value in this field gets loaded from Z1_GRABSECT1[21:20] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 10 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 10 to Zone1. 10 : No request for CPU1 Flash Sector 10 11 : No request for CPU1 Flash Sector 10 when this zone is UNLOCKED. Else CPU1 Flash Sector 10 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
19-18	GRAB_SECT9	R	0h	Value in this field gets loaded from Z1_GRABSECT1[19:18] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 9 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 9 to Zone1. 10 : No request for CPU1 Flash Sector 9 11 : No request for CPU1 Flash Sector 9 when this zone is UNLOCKED. Else CPU1 Flash Sector 9 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
17-16	GRAB_SECT8	R	0h	Value in this field gets loaded from Z1_GRABSECT1[17:16] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 8 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 8 to Zone1. 10 : No request for CPU1 Flash Sector 8 11 : No request for CPU1 Flash Sector 8 when this zone is UNLOCKED. Else CPU1 Flash Sector 8 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
15-14	GRAB_SECT7	R	0h	Value in this field gets loaded from Z1_GRABSECT1[15:14] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 7 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 7 to Zone1. 10 : No request for CPU1 Flash Sector 7 11 : No request for CPU1 Flash Sector 7 when this zone is UNLOCKED. Else CPU1 Flash Sector 7 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
13-12	GRAB_SECT6	R	0h	Value in this field gets loaded from Z1_GRABSECT1[13:12] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 6 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 6 to Zone1. 10 : No request for CPU1 Flash Sector 6 11 : No request for CPU1 Flash Sector 6 when this zone is UNLOCKED. Else CPU1 Flash Sector 6 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
11-10	GRAB_SECT5	R	0h	Value in this field gets loaded from Z1_GRABSECT1[11:10] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 5 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 5 to Zone1. 10 : No request for CPU1 Flash Sector 5 11 : No request for CPU1 Flash Sector 5 when this zone is UNLOCKED. Else CPU1 Flash Sector 5 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_SECT4	R	0h	Value in this field gets loaded from Z1_GRABSECT1[9:8] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 4 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 4 to Zone1. 10 : No request for CPU1 Flash Sector 4 11 : No request for CPU1 Flash Sector 4 when this zone is UNLOCKED. Else CPU1 Flash Sector 4 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-21. Z1_GRABSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	GRAB_SECT3	R	0h	Value in this field gets loaded from Z1_GRABSECT1[7:6] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 3 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 3 to Zone1. 10 : No request for CPU1 Flash Sector 3 11 : No request for CPU1 Flash Sector 3 when this zone is UNLOCKED. Else CPU1 Flash Sector 3 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
5-4	GRAB_SECT2	R	0h	Value in this field gets loaded from Z1_GRABSECT1[5:4] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 2 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 2 to Zone1. 10 : No request for CPU1 Flash Sector 2 11 : No request for CPU1 Flash Sector 2 when this zone is UNLOCKED. Else CPU1 Flash Sector 2 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
3-2	GRAB_SECT1	R	0h	Value in this field gets loaded from Z1_GRABSECT1[3:2] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 1 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 1 to Zone1. 10 : No request for CPU1 Flash Sector 1 11 : No request for CPU1 Flash Sector 1 when this zone is UNLOCKED. Else CPU1 Flash Sector 1 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_SECT0	R	0h	Value in this field gets loaded from Z1_GRABSECT1[1:0] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 0 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 0 to Zone1. 10 : No request for CPU1 Flash Sector 0 11 : No request for CPU1 Flash Sector 0 when this zone is UNLOCKED. Else CPU1 Flash Sector 0 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

6.9.3.15 Z1_GRABSECT2R Register (Offset (x8) = 38h, Offset (x16) = 1Ch) [Reset = 0h]

Z1_GRABSECT2R is shown in Figure 6-19 and described in Table 6-22.

Return to the Summary Table.

Zone 1 Grab Flash Status Register 2

Figure 6-19. Z1_GRABSECT2R Register

31	30	29	28	27	26	25	24
RESERVED				GRAB_SECT13		GRAB_SECT12	
R-0h				R-0h		R-0h	
23	22	21	20	19	18	17	16
GRAB_SECT11		GRAB_SECT10		GRAB_SECT9		GRAB_SECT8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
GRAB_SECT7		GRAB_SECT6		GRAB_SECT5		GRAB_SECT4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
GRAB_SECT3		GRAB_SECT2		GRAB_SECT1		GRAB_SECT0	
R-0h		R-0h		R-0h		R-0h	

Table 6-22. Z1_GRABSECT2R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-26	GRAB_SECT13	R	0h	Value in this field gets loaded from Z1_GRABSECT2[27:26] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 13 is inaccessible. 01 : Request to allocate CM Flash Sector 13 to Zone1. 10 : No request for CM Flash Sector 13 11 : No request for CM Flash Sector 13 when this zone is UNLOCKED. Else CM Flash Sector 13 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
25-24	GRAB_SECT12	R	0h	Value in this field gets loaded from Z1_GRABSECT2[25:24] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 12 is inaccessible. 01 : Request to allocate CM Flash Sector 12 to Zone1. 10 : No request for CM Flash Sector 12 11 : No request for CM Flash Sector 12 when this zone is UNLOCKED. Else CM Flash Sector 12 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
23-22	GRAB_SECT11	R	0h	Value in this field gets loaded from Z1_GRABSECT2[23:22] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 11 is inaccessible. 01 : Request to allocate CM Flash Sector 11 to Zone1. 10 : No request for CM Flash Sector 11 11 : No request for CM Flash Sector 11 when this zone is UNLOCKED. Else CM Flash Sector 11 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-22. Z1_GRABSECT2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	GRAB_SECT10	R	0h	Value in this field gets loaded from Z1_GRABSECT2[21:20] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 10 is inaccessible. 01 : Request to allocate CM Flash Sector 10 to Zone1. 10 : No request for CM Flash Sector 10 11 : No request for CM Flash Sector 10 when this zone is UNLOCKED. Else CM Flash Sector 10 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
19-18	GRAB_SECT9	R	0h	Value in this field gets loaded from Z1_GRABSECT2[19:18] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 9 is inaccessible. 01 : Request to allocate CM Flash Sector 9 to Zone1. 10 : No request for CM Flash Sector 9 11 : No request for CM Flash Sector 9 when this zone is UNLOCKED. Else CM Flash Sector 9 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
17-16	GRAB_SECT8	R	0h	Value in this field gets loaded from Z1_GRABSECT2[17:16] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 8 is inaccessible. 01 : Request to allocate CM Flash Sector 8 to Zone1. 10 : No request for CM Flash Sector 8 11 : No request for CM Flash Sector 8 when this zone is UNLOCKED. Else CM Flash Sector 8 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
15-14	GRAB_SECT7	R	0h	Value in this field gets loaded from Z1_GRABSECT2[15:14] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 7 is inaccessible. 01 : Request to allocate CM Flash Sector 7 to Zone1. 10 : No request for CM Flash Sector 7 11 : No request for CM Flash Sector 7 when this zone is UNLOCKED. Else CM Flash Sector 7 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
13-12	GRAB_SECT6	R	0h	Value in this field gets loaded from Z1_GRABSECT2[13:12] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 6 is inaccessible. 01 : Request to allocate CM Flash Sector 6 to Zone1. 10 : No request for CM Flash Sector 6 11 : No request for CM Flash Sector 6 when this zone is UNLOCKED. Else CM Flash Sector 6 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
11-10	GRAB_SECT5	R	0h	Value in this field gets loaded from Z1_GRABSECT2[11:10] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 5 is inaccessible. 01 : Request to allocate CM Flash Sector 5 to Zone1. 10 : No request for CM Flash Sector 5 11 : No request for CM Flash Sector 5 when this zone is UNLOCKED. Else CM Flash Sector 5 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_SECT4	R	0h	Value in this field gets loaded from Z1_GRABSECT2[9:8] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 4 is inaccessible. 01 : Request to allocate CM Flash Sector 4 to Zone1. 10 : No request for CM Flash Sector 4 11 : No request for CM Flash Sector 4 when this zone is UNLOCKED. Else CM Flash Sector 4 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-22. Z1_GRABSECT2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	GRAB_SECT3	R	0h	Value in this field gets loaded from Z1_GRABSECT2[7:6] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 3 is inaccessible. 01 : Request to allocate CM Flash Sector 3 to Zone1. 10 : No request for CM Flash Sector 3 11 : No request for CM Flash Sector 3 when this zone is UNLOCKED. Else CM Flash Sector 3 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
5-4	GRAB_SECT2	R	0h	Value in this field gets loaded from Z1_GRABSECT2[5:4] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 2 is inaccessible. 01 : Request to allocate CM Flash Sector 2 to Zone1. 10 : No request for CM Flash Sector 2 11 : No request for CM Flash Sector 2 when this zone is UNLOCKED. Else CM Flash Sector 2 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
3-2	GRAB_SECT1	R	0h	Value in this field gets loaded from Z1_GRABSECT2[3:2] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 1 is inaccessible. 01 : Request to allocate CM Flash Sector 1 to Zone1. 10 : No request for CM Flash Sector 1 11 : No request for CM Flash Sector 1 when this zone is UNLOCKED. Else CM Flash Sector 1 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_SECT0	R	0h	Value in this field gets loaded from Z1_GRABSECT2[1:0] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 0 is inaccessible. 01 : Request to allocate CM Flash Sector 0 to Zone1. 10 : No request for CM Flash Sector 0 11 : No request for CM Flash Sector 0 when this zone is UNLOCKED. Else CM Flash Sector 0 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

6.9.3.16 Z1_GRABSECT3R Register (Offset (x8) = 3Ch, Offset (x16) = 1Eh) [Reset = 0h]

Z1_GRABSECT3R is shown in Figure 6-20 and described in Table 6-23.

Return to the Summary Table.

Zone 1 Grab Flash Status Register 3

Figure 6-20. Z1_GRABSECT3R Register

31	30	29	28	27	26	25	24
RESERVED				GRAB_SECT13		GRAB_SECT12	
R-0h				R-0h		R-0h	
23	22	21	20	19	18	17	16
GRAB_SECT11		GRAB_SECT10		GRAB_SECT9		GRAB_SECT8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
GRAB_SECT7		GRAB_SECT6		GRAB_SECT5		GRAB_SECT4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
GRAB_SECT3		GRAB_SECT2		GRAB_SECT1		GRAB_SECT0	
R-0h		R-0h		R-0h		R-0h	

Table 6-23. Z1_GRABSECT3R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-26	GRAB_SECT13	R	0h	Value in this field gets loaded from Z1_GRABSECT3[27:26] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 13 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 13 to Zone1. 10 : No request for CPU2 Flash Sector 13 11 : No request for CPU2 Flash Sector 13 when this zone is UNLOCKED. Else CPU2 Flash Sector 13 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
25-24	GRAB_SECT12	R	0h	Value in this field gets loaded from Z1_GRABSECT3[25:24] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 12 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 12 to Zone1. 10 : No request for CPU2 Flash Sector 12 11 : No request for CPU2 Flash Sector 12 when this zone is UNLOCKED. Else CPU2 Flash Sector 12 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
23-22	GRAB_SECT11	R	0h	Value in this field gets loaded from Z1_GRABSECT3[23:22] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 11 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 11 to Zone1. 10 : No request for CPU2 Flash Sector 11 11 : No request for CPU2 Flash Sector 11 when this zone is UNLOCKED. Else CPU2 Flash Sector 11 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-23. Z1_GRABSECT3R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	GRAB_SECT10	R	0h	Value in this field gets loaded from Z1_GRABSECT3[21:20] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 10 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 10 to Zone1. 10 : No request for CPU2 Flash Sector 10 11 : No request for CPU2 Flash Sector 10 when this zone is UNLOCKED. Else CPU2 Flash Sector 10 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
19-18	GRAB_SECT9	R	0h	Value in this field gets loaded from Z1_GRABSECT3[19:18] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 9 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 9 to Zone1. 10 : No request for CPU2 Flash Sector 9 11 : No request for CPU2 Flash Sector 9 when this zone is UNLOCKED. Else CPU2 Flash Sector 9 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
17-16	GRAB_SECT8	R	0h	Value in this field gets loaded from Z1_GRABSECT3[17:16] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 8 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 8 to Zone1. 10 : No request for CPU2 Flash Sector 8 11 : No request for CPU2 Flash Sector 8 when this zone is UNLOCKED. Else CPU2 Flash Sector 8 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
15-14	GRAB_SECT7	R	0h	Value in this field gets loaded from Z1_GRABSECT3[15:14] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 7 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 7 to Zone1. 10 : No request for CPU2 Flash Sector 7 11 : No request for CPU2 Flash Sector 7 when this zone is UNLOCKED. Else CPU2 Flash Sector 7 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
13-12	GRAB_SECT6	R	0h	Value in this field gets loaded from Z1_GRABSECT3[13:12] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 6 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 6 to Zone1. 10 : No request for CPU2 Flash Sector 6 11 : No request for CPU2 Flash Sector 6 when this zone is UNLOCKED. Else CPU2 Flash Sector 6 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
11-10	GRAB_SECT5	R	0h	Value in this field gets loaded from Z1_GRABSECT3[11:10] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 5 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 5 to Zone1. 10 : No request for CPU2 Flash Sector 5 11 : No request for CPU2 Flash Sector 5 when this zone is UNLOCKED. Else CPU2 Flash Sector 5 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_SECT4	R	0h	Value in this field gets loaded from Z1_GRABSECT3[9:8] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 4 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 4 to Zone1. 10 : No request for CPU2 Flash Sector 4 11 : No request for CPU2 Flash Sector 4 when this zone is UNLOCKED. Else CPU2 Flash Sector 4 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-23. Z1_GRABSECT3R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	GRAB_SECT3	R	0h	Value in this field gets loaded from Z1_GRABSECT3[7:6] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 3 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 3 to Zone1. 10 : No request for CPU2 Flash Sector 3 11 : No request for CPU2 Flash Sector 3 when this zone is UNLOCKED. Else CPU2 Flash Sector 3 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
5-4	GRAB_SECT2	R	0h	Value in this field gets loaded from Z1_GRABSECT3[5:4] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 2 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 2 to Zone1. 10 : No request for CPU2 Flash Sector 2 11 : No request for CPU2 Flash Sector 2 when this zone is UNLOCKED. Else CPU2 Flash Sector 2 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
3-2	GRAB_SECT1	R	0h	Value in this field gets loaded from Z1_GRABSECT3[3:2] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 1 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 1 to Zone1. 10 : No request for CPU2 Flash Sector 1 11 : No request for CPU2 Flash Sector 1 when this zone is UNLOCKED. Else CPU2 Flash Sector 1 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_SECT0	R	0h	Value in this field gets loaded from Z1_GRABSECT3[1:0] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 0 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 0 to Zone1. 10 : No request for CPU2 Flash Sector 0 11 : No request for CPU2 Flash Sector 0 when this zone is UNLOCKED. Else CPU2 Flash Sector 0 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

6.9.3.17 Z1_GRABRAM1R Register (Offset (x8) = 40h, Offset (x16) = 20h) [Reset = 0h]

Z1_GRABRAM1R is shown in Figure 6-21 and described in Table 6-24.

Return to the Summary Table.

Zone 1 Grab RAM Status Register 1

Figure 6-21. Z1_GRABRAM1R Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				GRAB_RAM9		GRAB_RAM8	
R-0h				R-0h		R-0h	
15	14	13	12	11	10	9	8
GRAB_RAM7		GRAB_RAM6		GRAB_RAM5		GRAB_RAM4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
GRAB_RAM3		GRAB_RAM2		GRAB_RAM1		GRAB_RAM0	
R-0h		R-0h		R-0h		R-0h	

Table 6-24. Z1_GRABRAM1R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-18	GRAB_RAM9	R	0h	Value in this field gets loaded from Z1_GRABRAM1[19:18] when a read is issued to address location of Z1_GRABRAM1 in OTP. 00 : Invalid. CPU1 D1 RAM is inaccessible. 01 : Request to allocate CPU1 D1 RAM to Zone1. 10 : No request for CPU1 D1 RAM 11 : No request for CPU1 D1 RAM when this zone is UNLOCKED. Else CPU1 D1 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
17-16	GRAB_RAM8	R	0h	Value in this field gets loaded from Z1_GRABRAM1[17:16] when a read is issued to address location of Z1_GRABRAM1 in OTP. 00 : Invalid. CPU1 D0 RAM is inaccessible. 01 : Request to allocate CPU1 D0 RAM to Zone1. 10 : No request for CPU1 D0 RAM 11 : No request for CPU1 D0 RAM when this zone is UNLOCKED. Else CPU1 D0 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
15-14	GRAB_RAM7	R	0h	Value in this field gets loaded from Z1_GRABRAM1[15:14] when a read is issued to address location of Z1_GRABRAM1 in OTP. 00 : Invalid. CPU1 LS7 RAM is inaccessible. 01 : Request to allocate CPU1 LS7 RAM to Zone1. 10 : No request for CPU1 LS7 RAM 11 : No request for CPU1 LS7 RAM when this zone is UNLOCKED. Else CPU1 LS7 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
13-12	GRAB_RAM6	R	0h	Value in this field gets loaded from Z1_GRABRAM1[13:12] when a read is issued to address location of Z1_GRABRAM1 in OTP. 00 : Invalid. CPU1 LS6 RAM is inaccessible. 01 : Request to allocate CPU1 LS6 RAM to Zone1. 10 : No request for CPU1 LS6 RAM 11 : No request for CPU1 LS6 RAM when this zone is UNLOCKED. Else CPU1 LS6 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-24. Z1_GRABRAM1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11-10	GRAB_RAM5	R	0h	Value in this field gets loaded from Z1_GRABRAM1[11:10] when a read is issued to address location of Z1_GRABRAM1 in OTP. 00 : Invalid. CPU1 LS5 RAM is inaccessible. 01 : Request to allocate CPU1 LS5 RAM to Zone1. 10 : No request for CPU1 LS5 RAM 11 : No request for CPU1 LS5 RAM when this zone is UNLOCKED. Else CPU1 LS5 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_RAM4	R	0h	Value in this field gets loaded from Z1_GRABRAM1[9:8] when a read is issued to address location of Z1_GRABRAM1 in OTP. 00 : Invalid. CPU1 LS4 RAM is inaccessible. 01 : Request to allocate CPU1 LS4 RAM to Zone1. 10 : No request for CPU1 LS4 RAM 11 : No request for CPU1 LS4 RAM when this zone is UNLOCKED. Else CPU1 LS4 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
7-6	GRAB_RAM3	R	0h	Value in this field gets loaded from Z1_GRABRAM1[7:6] when a read is issued to address location of Z1_GRABRAM1 in OTP. 00 : Invalid. CPU1 LS3 RAM is inaccessible. 01 : Request to allocate CPU1 LS3 RAM to Zone1. 10 : No request for CPU1 LS3 RAM 11 : No request for CPU1 LS3 RAM when this zone is UNLOCKED. Else CPU1 LS3 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
5-4	GRAB_RAM2	R	0h	Value in this field gets loaded from Z1_GRABRAM1[5:4] when a read is issued to address location of Z1_GRABRAM1 in OTP. 00 : Invalid. CPU1 LS2 RAM is inaccessible. 01 : Request to allocate CPU1 LS2 RAM to Zone1. 10 : No request for CPU1 LS2 RAM 11 : No request for CPU1 LS2 RAM when this zone is UNLOCKED. Else CPU1 LS2 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
3-2	GRAB_RAM1	R	0h	Value in this field gets loaded from Z1_GRABRAM1[3:2] when a read is issued to address location of Z1_GRABRAM1 in OTP. 00 : Invalid. CPU1 LS1 RAM is inaccessible. 01 : Request to allocate CPU1 LS1 RAM to Zone1. 10 : No request for CPU1 LS1 RAM 11 : No request for CPU1 LS1 RAM when this zone is UNLOCKED. Else CPU1 LS1 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_RAM0	R	0h	Value in this field gets loaded from Z1_GRABRAM1[1:0] when a read is issued to address location of Z1_GRABRAM1 in OTP. 00 : Invalid. CPU1 LS0 RAM is inaccessible. 01 : Request to allocate CPU1 LS0 RAM to Zone1. 10 : No request for CPU1 LS0 RAM 11 : No request for CPU1 LS0 RAM when this zone is UNLOCKED. Else CPU1 LS0 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn

6.9.3.18 Z1_GRABRAM2R Register (Offset (x8) = 44h, Offset (x16) = 22h) [Reset = 0h]

Z1_GRABRAM2R is shown in Figure 6-22 and described in Table 6-25.

Return to the Summary Table.

Zone 1 Grab RAM Status Register 2

Figure 6-22. Z1_GRABRAM2R Register

31	30	29	28	27	26	25	24
GRAB_RAM15		GRAB_RAM14		GRAB_RAM13		GRAB_RAM12	
R-0h		R-0h		R-0h		R-0h	
23	22	21	20	19	18	17	16
GRAB_RAM11		GRAB_RAM10		GRAB_RAM9		GRAB_RAM8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
GRAB_RAM7		GRAB_RAM6		GRAB_RAM5		GRAB_RAM4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
RESERVED				GRAB_RAM1		GRAB_RAM0	
R-0h				R-0h		R-0h	

Table 6-25. Z1_GRABRAM2R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GRAB_RAM15	R	0h	Value in this field gets loaded from Z1_GRABRAM2[31:30] when a read is issued to address location of Z1_GRABRAM2 in OTP. Defines Zone 1's grab request of the higher addressed half of CPU2TOCPU1MSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone1. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
29-28	GRAB_RAM14	R	0h	Value in this field gets loaded from Z1_GRABRAM2[29:28] when a read is issued to address location of Z1_GRABRAM2 in OTP. Defines Zone 1's grab request of the lower addressed half of CPU2TOCPU1MSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone1. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
27-26	GRAB_RAM13	R	0h	Value in this field gets loaded from Z1_GRABRAM2[27:26] when a read is issued to address location of Z1_GRABRAM2 in OTP. Defines Zone 1's grab request of the higher addressed half of CPU1TOCPU2MSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone1. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-25. Z1_GRABRAM2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-24	GRAB_RAM12	R	0h	Value in this field gets loaded from Z1_GRABRAM2[25:24] when a read is issued to address location of Z1_GRABRAM2 in OTP. Defines Zone 1's grab request of the lower addressed half of CPU1TOCPU2MSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone1. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
23-22	GRAB_RAM11	R	0h	Value in this field gets loaded from Z1_GRABRAM2[23:22] when a read is issued to address location of Z1_GRABRAM2 in OTP. Defines Zone 1's grab request of the higher addressed half of CMTOCPU2MSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone1. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
21-20	GRAB_RAM10	R	0h	Value in this field gets loaded from Z1_GRABRAM2[21:20] when a read is issued to address location of Z1_GRABRAM2 in OTP. Defines Zone 1's grab request of the lower addressed half of CMTOCPU2MSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone1. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
19-18	GRAB_RAM9	R	0h	Value in this field gets loaded from Z1_GRABRAM2[19:18] when a read is issued to address location of Z1_GRABRAM2 in OTP. Defines Zone 1's grab request of the higher addressed half of CPU2TOCMMSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone1. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
17-16	GRAB_RAM8	R	0h	Value in this field gets loaded from Z1_GRABRAM2[17:16] when a read is issued to address location of Z1_GRABRAM2 in OTP. Defines Zone 1's grab request of the lower addressed half of CPU2TOCMMSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone1. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
15-14	GRAB_RAM7	R	0h	Value in this field gets loaded from Z1_GRABRAM2[15:14] when a read is issued to address location of Z1_GRABRAM2 in OTP. Defines Zone 1's grab request of the higher addressed half of CMTOCPU1MSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone1. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-25. Z1_GRABRAM2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13-12	GRAB_RAM6	R	0h	Value in this field gets loaded from Z1_GRABRAM2[13:12] when a read is issued to address location of Z1_GRABRAM2 in OTP. Defines Zone 1's grab request of the lower addressed half of CMTOCPU1MSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone1. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
11-10	GRAB_RAM5	R	0h	Value in this field gets loaded from Z1_GRABRAM2[11:10] when a read is issued to address location of Z1_GRABRAM2 in OTP. Defines Zone 1's grab request of the higher addressed half of CPU1TOCMMSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone1. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_RAM4	R	0h	Value in this field gets loaded from Z1_GRABRAM2[9:8] when a read is issued to address location of Z1_GRABRAM2 in OTP. Defines Zone 1's grab request of the lower addressed half of CPU1TOCMMSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone1. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
7-4	RESERVED	R	0h	Reserved
3-2	GRAB_RAM1	R	0h	Value in this field gets loaded from Z1_GRABRAM2[3:2] when a read is issued to address location of Z1_GRABRAM2 in OTP. 00 : Invalid. CM C1 RAM is inaccessible. 01 : Request to allocate CM C1 RAM to Zone1. 10 : No request for CM C1 RAM 11 : No request for CM C1 RAM when this zone is UNLOCKED. Else CM C1 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_RAM0	R	0h	Value in this field gets loaded from Z1_GRABRAM2[1:0] when a read is issued to address location of Z1_GRABRAM2 in OTP. 00 : Invalid. CM C0 RAM is inaccessible. 01 : Request to allocate CM C0 RAM to Zone1. 10 : No request for CM C0 RAM 11 : No request for CM C0 RAM when this zone is UNLOCKED. Else CM C0 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn

6.9.3.19 Z1_GRABRAM3R Register (Offset (x8) = 48h, Offset (x16) = 24h) [Reset = 0h]

Z1_GRABRAM3R is shown in Figure 6-23 and described in Table 6-26.

Return to the Summary Table.

Zone 1 Grab RAM Status Register 3

Figure 6-23. Z1_GRABRAM3R Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				GRAB_RAM9		GRAB_RAM8	
R-0h				R-0h		R-0h	
15	14	13	12	11	10	9	8
GRAB_RAM7		GRAB_RAM6		GRAB_RAM5		GRAB_RAM4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
GRAB_RAM3		GRAB_RAM2		GRAB_RAM1		GRAB_RAM0	
R-0h		R-0h		R-0h		R-0h	

Table 6-26. Z1_GRABRAM3R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-18	GRAB_RAM9	R	0h	Value in this field gets loaded from Z1_GRABRAM3[19:18] when a read is issued to address location of Z1_GRABRAM3 in OTP. 00 : Invalid. CPU2 D1 RAM is inaccessible. 01 : Request to allocate CPU2 D1 RAM to Zone1. 10 : No request for CPU2 D1 RAM 11 : No request for CPU2 D1 RAM when this zone is UNLOCKED. Else CPU2 D1 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
17-16	GRAB_RAM8	R	0h	Value in this field gets loaded from Z1_GRABRAM3[17:16] when a read is issued to address location of Z1_GRABRAM3 in OTP. 00 : Invalid. CPU2 D0 RAM is inaccessible. 01 : Request to allocate CPU2 D0 RAM to Zone1. 10 : No request for CPU2 D0 RAM 11 : No request for CPU2 D0 RAM when this zone is UNLOCKED. Else CPU2 D0 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
15-14	GRAB_RAM7	R	0h	Value in this field gets loaded from Z1_GRABRAM3[15:14] when a read is issued to address location of Z1_GRABRAM3 in OTP. 00 : Invalid. CPU2 LS7 RAM is inaccessible. 01 : Request to allocate CPU2 LS7 RAM to Zone1. 10 : No request for CPU2 LS7 RAM 11 : No request for CPU2 LS7 RAM when this zone is UNLOCKED. Else CPU2 LS7 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
13-12	GRAB_RAM6	R	0h	Value in this field gets loaded from Z1_GRABRAM3[13:12] when a read is issued to address location of Z1_GRABRAM3 in OTP. 00 : Invalid. CPU2 LS6 RAM is inaccessible. 01 : Request to allocate CPU2 LS6 RAM to Zone1. 10 : No request for CPU2 LS6 RAM 11 : No request for CPU2 LS6 RAM when this zone is UNLOCKED. Else CPU2 LS6 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-26. Z1_GRABRAM3R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11-10	GRAB_RAM5	R	0h	Value in this field gets loaded from Z1_GRABRAM3[11:10] when a read is issued to address location of Z1_GRABRAM3 in OTP. 00 : Invalid. CPU2 LS5 RAM is inaccessible. 01 : Request to allocate CPU2 LS5 RAM to Zone1. 10 : No request for CPU2 LS5 RAM 11 : No request for CPU2 LS5 RAM when this zone is UNLOCKED. Else CPU2 LS5 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_RAM4	R	0h	Value in this field gets loaded from Z1_GRABRAM3[9:8] when a read is issued to address location of Z1_GRABRAM3 in OTP. 00 : Invalid. CPU2 LS4 RAM is inaccessible. 01 : Request to allocate CPU2 LS4 RAM to Zone1. 10 : No request for CPU2 LS4 RAM 11 : No request for CPU2 LS4 RAM when this zone is UNLOCKED. Else CPU2 LS4 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
7-6	GRAB_RAM3	R	0h	Value in this field gets loaded from Z1_GRABRAM3[7:6] when a read is issued to address location of Z1_GRABRAM3 in OTP. 00 : Invalid. CPU2 LS3 RAM is inaccessible. 01 : Request to allocate CPU2 LS3 RAM to Zone1. 10 : No request for CPU2 LS3 RAM 11 : No request for CPU2 LS3 RAM when this zone is UNLOCKED. Else CPU2 LS3 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
5-4	GRAB_RAM2	R	0h	Value in this field gets loaded from Z1_GRABRAM3[5:4] when a read is issued to address location of Z1_GRABRAM3 in OTP. 00 : Invalid. CPU2 LS2 RAM is inaccessible. 01 : Request to allocate CPU2 LS2 RAM to Zone1. 10 : No request for CPU2 LS2 RAM 11 : No request for CPU2 LS2 RAM when this zone is UNLOCKED. Else CPU2 LS2 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
3-2	GRAB_RAM1	R	0h	Value in this field gets loaded from Z1_GRABRAM3[3:2] when a read is issued to address location of Z1_GRABRAM3 in OTP. 00 : Invalid. CPU2 LS1 RAM is inaccessible. 01 : Request to allocate CPU2 LS1 RAM to Zone1. 10 : No request for CPU2 LS1 RAM 11 : No request for CPU2 LS1 RAM when this zone is UNLOCKED. Else CPU2 LS1 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_RAM0	R	0h	Value in this field gets loaded from Z1_GRABRAM3[1:0] when a read is issued to address location of Z1_GRABRAM3 in OTP. 00 : Invalid. CPU2 LS0 RAM is inaccessible. 01 : Request to allocate CPU2 LS0 RAM to Zone1. 10 : No request for CPU2 LS0 RAM 11 : No request for CPU2 LS0 RAM when this zone is UNLOCKED. Else CPU2 LS0 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn

6.9.3.20 Z1_EXEONLYSECT1R Register (Offset (x8) = 4Ch, Offset (x16) = 26h) [Reset = 0h]

Z1_EXEONLYSECT1R is shown in Figure 6-24 and described in Table 6-27.

Return to the Summary Table.

Zone 1 Execute Only Flash Status Register 1

Figure 6-24. Z1_EXEONLYSECT1R Register

31	30	29	28	27	26	25	24
RESERVED		EXEONLY_CM_SECT13	EXEONLY_CM_SECT12	EXEONLY_CM_SECT11	EXEONLY_CM_SECT10	EXEONLY_CM_SECT9	EXEONLY_CM_SECT8
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
EXEONLY_CM_SECT7	EXEONLY_CM_SECT6	EXEONLY_CM_SECT5	EXEONLY_CM_SECT4	EXEONLY_CM_SECT3	EXEONLY_CM_SECT2	EXEONLY_CM_SECT1	EXEONLY_CM_SECT0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED		EXEONLY_CPU1_SECT13	EXEONLY_CPU1_SECT12	EXEONLY_CPU1_SECT11	EXEONLY_CPU1_SECT10	EXEONLY_CPU1_SECT9	EXEONLY_CPU1_SECT8
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
EXEONLY_CPU1_SECT7	EXEONLY_CPU1_SECT6	EXEONLY_CPU1_SECT5	EXEONLY_CPU1_SECT4	EXEONLY_CPU1_SECT3	EXEONLY_CPU1_SECT2	EXEONLY_CPU1_SECT1	EXEONLY_CPU1_SECT0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 6-27. Z1_EXEONLYSECT1R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29	EXEONLY_CM_SECT13	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[29] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 13 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CM Flash Sector 13 (only if it's allocated to Zone1) Reset type: SYSRSn
28	EXEONLY_CM_SECT12	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[28] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 12 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CM Flash Sector 12 (only if it's allocated to Zone1) Reset type: SYSRSn
27	EXEONLY_CM_SECT11	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[27] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 11 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CM Flash Sector 11 (only if it's allocated to Zone1) Reset type: SYSRSn
26	EXEONLY_CM_SECT10	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[26] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 10 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CM Flash Sector 10 (only if it's allocated to Zone1) Reset type: SYSRSn

Table 6-27. Z1_EXEONLYSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	EXEONLY_CM_SECT9	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[25] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 9 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CM Flash Sector 9 (only if it's allocated to Zone1) Reset type: SYSRSn
24	EXEONLY_CM_SECT8	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[24] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 8 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CM Flash Sector 8 (only if it's allocated to Zone1) Reset type: SYSRSn
23	EXEONLY_CM_SECT7	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[23] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 7 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CM Flash Sector 7 (only if it's allocated to Zone1) Reset type: SYSRSn
22	EXEONLY_CM_SECT6	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[22] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 6 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CM Flash Sector 6 (only if it's allocated to Zone1) Reset type: SYSRSn
21	EXEONLY_CM_SECT5	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[21] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 5 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CM Flash Sector 5 (only if it's allocated to Zone1) Reset type: SYSRSn
20	EXEONLY_CM_SECT4	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[20] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 4 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CM Flash Sector 4 (only if it's allocated to Zone1) Reset type: SYSRSn
19	EXEONLY_CM_SECT3	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[19] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 3 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CM Flash Sector 3 (only if it's allocated to Zone1) Reset type: SYSRSn
18	EXEONLY_CM_SECT2	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[18] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 2 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CM Flash Sector 2 (only if it's allocated to Zone1) Reset type: SYSRSn

Table 6-27. Z1_EXEONLYSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	EXEONLY_CM_SECT1	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[17] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 1 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CM Flash Sector 1 (only if it's allocated to Zone1) Reset type: SYSRSn
16	EXEONLY_CM_SECT0	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[16] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 0 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CM Flash Sector 0 (only if it's allocated to Zone1) Reset type: SYSRSn
15-14	RESERVED	R	0h	Reserved
13	EXEONLY_CPU1_SECT13	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[13] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 13 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 13 (only if it's allocated to Zone1) Reset type: SYSRSn
12	EXEONLY_CPU1_SECT12	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[12] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 12 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 12 (only if it's allocated to Zone1) Reset type: SYSRSn
11	EXEONLY_CPU1_SECT11	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[11] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 11 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 11 (only if it's allocated to Zone1) Reset type: SYSRSn
10	EXEONLY_CPU1_SECT10	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[10] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 10 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 10 (only if it's allocated to Zone1) Reset type: SYSRSn
9	EXEONLY_CPU1_SECT9	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[9] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 9 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 9 (only if it's allocated to Zone1) Reset type: SYSRSn
8	EXEONLY_CPU1_SECT8	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[8] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 8 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 8 (only if it's allocated to Zone1) Reset type: SYSRSn

Table 6-27. Z1_EXEONLYSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	EXEONLY_CPU1_SECT7	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[7] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 7 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 7 (only if it's allocated to Zone1) Reset type: SYSRSn
6	EXEONLY_CPU1_SECT6	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[6] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 6 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 6 (only if it's allocated to Zone1) Reset type: SYSRSn
5	EXEONLY_CPU1_SECT5	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[5] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 5 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 5 (only if it's allocated to Zone1) Reset type: SYSRSn
4	EXEONLY_CPU1_SECT4	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[4] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 4 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 4 (only if it's allocated to Zone1) Reset type: SYSRSn
3	EXEONLY_CPU1_SECT3	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[3] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 3 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 3 (only if it's allocated to Zone1) Reset type: SYSRSn
2	EXEONLY_CPU1_SECT2	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[2] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 2 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 2 (only if it's allocated to Zone1) Reset type: SYSRSn
1	EXEONLY_CPU1_SECT1	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[1] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 1 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 1 (only if it's allocated to Zone1) Reset type: SYSRSn
0	EXEONLY_CPU1_SECT0	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[0] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 0 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 0 (only if it's allocated to Zone1) Reset type: SYSRSn

6.9.3.21 Z1_EXEONLYSECT2R Register (Offset (x8) = 50h, Offset (x16) = 28h) [Reset = 0h]

Z1_EXEONLYSECT2R is shown in Figure 6-25 and described in Table 6-28.

Return to the Summary Table.

Zone 1 Execute Only Flash Status Register 2

Figure 6-25. Z1_EXEONLYSECT2R Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		EXEONLY_CPU2_SECT13	EXEONLY_CPU2_SECT12	EXEONLY_CPU2_SECT11	EXEONLY_CPU2_SECT10	EXEONLY_CPU2_SECT9	EXEONLY_CPU2_SECT8
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
EXEONLY_CPU2_SECT7	EXEONLY_CPU2_SECT6	EXEONLY_CPU2_SECT5	EXEONLY_CPU2_SECT4	EXEONLY_CPU2_SECT3	EXEONLY_CPU2_SECT2	EXEONLY_CPU2_SECT1	EXEONLY_CPU2_SECT0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 6-28. Z1_EXEONLYSECT2R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13	EXEONLY_CPU2_SECT13	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[13] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 13 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 13 (only if it's allocated to Zone1) Reset type: SYSRSn
12	EXEONLY_CPU2_SECT12	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[12] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 12 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 12 (only if it's allocated to Zone1) Reset type: SYSRSn
11	EXEONLY_CPU2_SECT11	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[11] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 11 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 11 (only if it's allocated to Zone1) Reset type: SYSRSn
10	EXEONLY_CPU2_SECT10	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[10] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 10 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 10 (only if it's allocated to Zone1) Reset type: SYSRSn

Table 6-28. Z1_EXEONLYSECT2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	EXEONLY_CPU2_SECT9	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[9] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 9 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 9 (only if it's allocated to Zone1) Reset type: SYSRSn
8	EXEONLY_CPU2_SECT8	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[8] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 8 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 8 (only if it's allocated to Zone1) Reset type: SYSRSn
7	EXEONLY_CPU2_SECT7	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[7] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 7 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 7 (only if it's allocated to Zone1) Reset type: SYSRSn
6	EXEONLY_CPU2_SECT6	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[6] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 6 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 6 (only if it's allocated to Zone1) Reset type: SYSRSn
5	EXEONLY_CPU2_SECT5	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[5] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 5 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 5 (only if it's allocated to Zone1) Reset type: SYSRSn
4	EXEONLY_CPU2_SECT4	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[4] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 4 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 4 (only if it's allocated to Zone1) Reset type: SYSRSn
3	EXEONLY_CPU2_SECT3	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[3] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 3 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 3 (only if it's allocated to Zone1) Reset type: SYSRSn
2	EXEONLY_CPU2_SECT2	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[2] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 2 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 2 (only if it's allocated to Zone1) Reset type: SYSRSn

Table 6-28. Z1_EXEONLYSECT2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	EXEONLY_CPU2_SECT1	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[1] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 1 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 1 (only if it's allocated to Zone1) Reset type: SYSRSn
0	EXEONLY_CPU2_SECT0	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[0] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 0 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 0 (only if it's allocated to Zone1) Reset type: SYSRSn

6.9.3.22 Z1_EXEONLYRAM1R Register (Offset (x8) = 54h, Offset (x16) = 2Ah) [Reset = 0h]

Z1_EXEONLYRAM1R is shown in Figure 6-26 and described in Table 6-29.

Return to the Summary Table.

Zone 1 Execute Only RAM Status Register 1

Figure 6-26. Z1_EXEONLYRAM1R Register

31	30	29	28	27	26	25	24
EXEONLY_RAM31	EXEONLY_RAM30	EXEONLY_RAM29	EXEONLY_RAM28	EXEONLY_RAM27	EXEONLY_RAM26	EXEONLY_RAM25	EXEONLY_RAM24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
EXEONLY_RAM23	EXEONLY_RAM22	RESERVED				EXEONLY_RAM17	EXEONLY_RAM16
R-0h	R-0h	R-0h				R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED						EXEONLY_RAM9	EXEONLY_RAM8
R-0h						R-0h	R-0h
7	6	5	4	3	2	1	0
EXEONLY_RAM7	EXEONLY_RAM6	EXEONLY_RAM5	EXEONLY_RAM4	EXEONLY_RAM3	EXEONLY_RAM2	EXEONLY_RAM1	EXEONLY_RAM0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 6-29. Z1_EXEONLYRAM1R Register Field Descriptions

Bit	Field	Type	Reset	Description
31	EXEONLY_RAM31	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[31] when a read is issued to Z1_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CPU2 LS0 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 LS0 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
30	EXEONLY_RAM30	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[30] when a read is issued to Z1_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CPU2 LS1 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 LS1 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
29	EXEONLY_RAM29	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[29] when a read is issued to Z1_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CPU2 LS2 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 LS2 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
28	EXEONLY_RAM28	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[28] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 LS3 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 LS3 RAM (only if it's allocated to Zone1) Reset type: SYSRSn

Table 6-29. Z1_EXEONLYRAM1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	EXEONLY_RAM27	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[27] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 LS4 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 LS4 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
26	EXEONLY_RAM26	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[26] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 LS5 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 LS5 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
25	EXEONLY_RAM25	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[25] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 LS6 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 LS6 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
24	EXEONLY_RAM24	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[24] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 LS7 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 LS7 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
23	EXEONLY_RAM23	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[23] when a read is issued to Z1_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CPU2 D0 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 D0 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
22	EXEONLY_RAM22	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[22] when a read is issued to Z1_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CPU2 D1 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU2 D1 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
21-18	RESERVED	R	0h	Reserved
17	EXEONLY_RAM17	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[17] when a read is issued to Z1_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CM C1 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CM C1 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
16	EXEONLY_RAM16	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[16] when a read is issued to Z1_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CM C0 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CM C0 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
15-10	RESERVED	R	0h	Reserved

Table 6-29. Z1_EXEONLYRAM1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	EXEONLY_RAM9	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[9] when a read is issued to Z1_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CPU1 D1 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 D1 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
8	EXEONLY_RAM8	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[8] when a read is issued to Z1_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CPU1 D0 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 D0 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
7	EXEONLY_RAM7	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[7] when a read is issued to Z1_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CPU1 LS7 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 LS7 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
6	EXEONLY_RAM6	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[6] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 LS6 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 LS6 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
5	EXEONLY_RAM5	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[5] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 LS5 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 LS5 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
4	EXEONLY_RAM4	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[4] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 LS4 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 LS4 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
3	EXEONLY_RAM3	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[3] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 LS3 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 LS3 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
2	EXEONLY_RAM2	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[2] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 LS2 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 LS2 RAM (only if it's allocated to Zone1) Reset type: SYSRSn

Table 6-29. Z1_EXEONLYRAM1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	EXEONLY_RAM1	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[1] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 LS1 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 LS1 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
0	EXEONLY_RAM0	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[0] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 LS0 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for CPU1 LS0 RAM (only if it's allocated to Zone1) Reset type: SYSRSn

6.9.3.23 Z1_JTAGKEY0 Register (Offset (x8) = 5Ch, Offset (x16) = 2Eh) [Reset = 0h]

Z1_JTAGKEY0 is shown in Figure 6-27 and described in Table 6-30.

Return to the Summary Table.

JTAG Unlock Key Register 0

Figure 6-27. Z1_JTAGKEY0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY0																															
R-0h																															

Table 6-30. Z1_JTAGKEY0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY0	R	0h	Value in this field is scanned in via the JLM scan chain. JTAGKEY is compared to a hidden register that gets dummy loaded when a read is issued to the JTAGPSWD locations in OTP. Reset type: PORESETn

6.9.3.24 Z1_JTAGKEY1 Register (Offset (x8) = 60h, Offset (x16) = 30h) [Reset = 0h]

Z1_JTAGKEY1 is shown in Figure 6-28 and described in Table 6-31.

Return to the Summary Table.

JTAG Unlock Key Register 1

Figure 6-28. Z1_JTAGKEY1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY1																															
R-0h																															

Table 6-31. Z1_JTAGKEY1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY1	R	0h	Value in this field is scanned in via the JLM scan chain. JTAGKEY is compared to a hidden register that gets dummy loaded when a read is issued to the JTAGPSWD locations in OTP. Reset type: PORESETn

6.9.3.25 Z1_JTAGKEY2 Register (Offset (x8) = 64h, Offset (x16) = 32h) [Reset = 0h]

Z1_JTAGKEY2 is shown in Figure 6-29 and described in Table 6-32.

Return to the Summary Table.

JTAG Unlock Key Register 2

Figure 6-29. Z1_JTAGKEY2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY2																															
R-0h																															

Table 6-32. Z1_JTAGKEY2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY2	R	0h	Value in this field is scanned in via the JLM scan chain. JTAGKEY is compared to a hidden register that gets dummy loaded when a read is issued to the JTAGPSWD locations in OTP. Reset type: PORESETn

6.9.3.26 Z1_JTAGKEY3 Register (Offset (x8) = 68h, Offset (x16) = 34h) [Reset = 0h]

Z1_JTAGKEY3 is shown in Figure 6-30 and described in Table 6-33.

Return to the Summary Table.

JTAG Unlock Key Register 3

Figure 6-30. Z1_JTAGKEY3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY3																															
R-0h																															

Table 6-33. Z1_JTAGKEY3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY3	R	0h	Value in this field is scanned in via the JLM scan chain. JTAGKEY is compared to a hidden register that gets dummy loaded when a read is issued to the JTAGPSWD locations in OTP. Reset type: PORESETn

6.9.3.27 Z1_CMACKEY0 Register (Offset (x8) = 6Ch, Offset (x16) = 36h) [Reset = 0h]

Z1_CMACKEY0 is shown in Figure 6-31 and described in Table 6-34.

Return to the Summary Table.

Secure Boot CMAC Key Status Register 0

Figure 6-31. Z1_CMACKEY0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY0																															
R-0h																															

Table 6-34. Z1_CMACKEY0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY0	R	0h	Value in this field gets loaded from CMACKEY0 when a read is issued to its address in OTP. Reset type: SYSRSn

6.9.3.28 Z1_CMACKEY1 Register (Offset (x8) = 70h, Offset (x16) = 38h) [Reset = 0h]

Z1_CMACKEY1 is shown in Figure 6-32 and described in Table 6-35.

Return to the Summary Table.

Secure Boot CMAC Key Status Register 1

Figure 6-32. Z1_CMACKEY1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY1																															
R-0h																															

Table 6-35. Z1_CMACKEY1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY1	R	0h	Value in this field gets loaded from CMACKEY1 when a read is issued to its address in OTP. Reset type: SYSRSn

6.9.3.29 Z1_CMACKEY2 Register (Offset (x8) = 74h, Offset (x16) = 3Ah) [Reset = 0h]

Z1_CMACKEY2 is shown in Figure 6-33 and described in Table 6-36.

Return to the Summary Table.

Secure Boot CMAC Key Status Register 2

Figure 6-33. Z1_CMACKEY2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY2																															
R-0h																															

Table 6-36. Z1_CMACKEY2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY2	R	0h	Value in this field gets loaded from CMACKEY2 when a read is issued to its address in OTP. Reset type: SYSRSn

6.9.3.30 Z1_CMACKEY3 Register (Offset (x8) = 78h, Offset (x16) = 3Ch) [Reset = 0h]

Z1_CMACKEY3 is shown in Figure 6-34 and described in Table 6-37.

Return to the Summary Table.

Secure Boot CMAC Key Status Register 3

Figure 6-34. Z1_CMACKEY3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY3																															
R-0h																															

Table 6-37. Z1_CMACKEY3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY3	R	0h	Value in this field gets loaded from CMACKEY3 when a read is issued to its address in OTP. Reset type: SYSRSn

6.9.4 DCSM_Z2_REGS Registers

Table 6-38 lists the memory-mapped registers for the DCSM_Z2_REGS registers. All register offset addresses not listed in Table 6-38 should be considered as reserved locations and the register contents should not be modified.

Table 6-38. DCSM_Z2_REGS Registers

Offset (x8)	Offset (x16)	Acronym	Register Name	Write Protection	Section
0h	0h	Z2_LINKPOINTER	Zone 2 Link Pointer		Go
4h	2h	Z2_OTPSECLOCK	Zone 2 OTP Secure Lock		Go
Ch	6h	Z2_LINKPOINTERERR	Link Pointer Error		Go
10h	8h	Z2_GPREG1	Zone 2 General Purpose Register-1		Go
14h	Ah	Z2_GPREG2	Zone 2 General Purpose Register-2		Go
18h	Ch	Z2_GPREG3	Zone 2 General Purpose Register-3		Go
1Ch	Eh	Z2_GPREG4	Zone 2 General Purpose Register-4		Go
20h	10h	Z2_CSMKEY0	Zone 2 CSM Key 0		Go
24h	12h	Z2_CSMKEY1	Zone 2 CSM Key 1		Go
28h	14h	Z2_CSMKEY2	Zone 2 CSM Key 2		Go
2Ch	16h	Z2_CSMKEY3	Zone 2 CSM Key 3		Go
30h	18h	Z2_CR	Zone 2 CSM Control Register		Go
34h	1Ah	Z2_GRABSECT1R	Zone 2 Grab Flash Status Register 1		Go
38h	1Ch	Z2_GRABSECT2R	Zone 2 Grab Flash Status Register 2		Go
3Ch	1Eh	Z2_GRABSECT3R	Zone 2 Grab Flash Status Register 3		Go
40h	20h	Z2_GRABRAM1R	Zone 2 Grab RAM Status Register 1		Go
44h	22h	Z2_GRABRAM2R	Zone 2 Grab RAM Status Register 2		Go
48h	24h	Z2_GRABRAM3R	Zone 2 Grab RAM Status Register 3		Go
4Ch	26h	Z2_EXEONLYSECT1R	Zone 2 Execute Only Flash Status Register 1		Go
50h	28h	Z2_EXEONLYSECT2R	Zone 2 Execute Only Flash Status Register 2		Go
54h	2Ah	Z2_EXEONLYRAM1R	Zone 2 Execute Only RAM Status Register 1		Go

Complex bit access types are encoded to fit into small table cells. Table 6-39 shows the codes that are used for access types in this section.

Table 6-39. DCSM_Z2_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 6-39. DCSM_Z2_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

6.9.4.1 Z2_LINKPOINTER Register (Offset (x8) = 0h, Offset (x16) = 0h) [Reset = FFFFC000h]

Z2_LINKPOINTER is shown in Figure 6-35 and described in Table 6-40.

Return to the Summary Table.

Zone 2 Link Pointer

Figure 6-35. Z2_LINKPOINTER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																		LINKPOINTER													
R-0h																		R-0h													

Table 6-40. Z2_LINKPOINTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13-0	LINKPOINTER	R	0h	This is resolved Link-Pointer value which is generated by looking at the three physical Link-Pointer values loaded from OTP Reset type: SYSRSn

6.9.4.2 Z2_OTPSECLOCK Register (Offset (x8) = 4h, Offset (x16) = 2h) [Reset = 1h]

Z2_OTPSECLOCK is shown in Figure 6-36 and described in Table 6-41.

Return to the Summary Table.

Zone 2 OTP Secure Lock

Figure 6-36. Z2_OTPSECLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				CRCLOCK			
R-0h				R-0h			
7	6	5	4	3	2	1	0
PSWDLOCK				RESERVED			JTAGLOCK
R-0h				R-0h			R-1h

Table 6-41. Z2_OTPSECLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-12	RESERVED	R	0h	Reserved
11-8	CRCLOCK	R	0h	Value in this field gets loaded from Z2_CRCLOCK[3:0] when a read is issued to address location of Z2_CRCLOCK in OTP. 1111 : VCU has ability to calculate CRC on secure memories. Other Value : VCU doesn't have ability to calculate CRC on secure memories. Reset type: XRSn
7-4	PSWDLOCK	R	0h	Value in this field gets loaded from Z2_PSWDLOCK[3:0] when a read is issued to address location of Z1_PSWDLOCK in OTP. 1111 : CSM password locations in OTP are not protected and can be read from debugger as well as code running from anywhere. Other Value : CSM password locations in OTP are protected and can't be read without unlocking CSM of that zone. Reset type: XRSn
3-1	RESERVED	R	0h	Reserved
0	JTAGLOCK	R	1h	Reflects the state of the JTAGLOCK feature. 0 : JTAG is not locked 1 : JTAG is locked This bit is a copy of the Z1_OTPSECLOCK.JTAGLOCK bit. Reset type: PORESETn

6.9.4.3 Z2_LINKPOINTERERR Register (Offset (x8) = Ch, Offset (x16) = 6h) [Reset = 0h]

Z2_LINKPOINTERERR is shown in Figure 6-37 and described in Table 6-42.

Return to the Summary Table.

Link Pointer Error

Figure 6-37. Z2_LINKPOINTERERR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		Z2_LINKPOINTERERR													
R-0h		R-0h													

Table 6-42. Z2_LINKPOINTERERR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13-0	Z2_LINKPOINTERERR	R	0h	These bits indicate errors during formation of the resolved Link-Pointer value after the three physical Link-Pointer values loaded of OTP in flash. 0 : No Error. Other : Error on bit positions which is set to 1. Reset type: SYSRSn

6.9.4.4 Z2_GPREG1 Register (Offset (x8) = 10h, Offset (x16) = 8h) [Reset = 0h]

Z2_GPREG1 is shown in Figure 6-38 and described in Table 6-43.

Return to the Summary Table.

Zone 2 General Purpose Register-1

Figure 6-38. Z2_GPREG1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPREG1																															
R-0h																															

Table 6-43. Z2_GPREG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPREG1	R	0h	This field gets loaded with the contents of Z2OTP_GPREG1 locations in Zone-2-USER-OTP when a dummy read is issued to that address. Users can use this register to load any general purpose non-voliate content from USER-OTP of Zone-2 Reset type: SYSRSn

6.9.4.5 Z2_GPREG2 Register (Offset (x8) = 14h, Offset (x16) = Ah) [Reset = 0h]

Z2_GPREG2 is shown in Figure 6-39 and described in Table 6-44.

Return to the Summary Table.

Zone 2 General Purpose Register-2

Figure 6-39. Z2_GPREG2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPREG2																															
R-0h																															

Table 6-44. Z2_GPREG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPREG2	R	0h	This field gets loaded with the contents of Z2OTP_GPREG2 locations in Zone-2-USER-OTP when a dummy read is issued to that address. Users can use this register to load any general purpose non-voliate content from USER-OTP of Zone-2 Reset type: SYSRSn

6.9.4.6 Z2_GPREG3 Register (Offset (x8) = 18h, Offset (x16) = Ch) [Reset = 0h]

Z2_GPREG3 is shown in Figure 6-40 and described in Table 6-45.

Return to the Summary Table.

Zone 2 General Purpose Register-3

Figure 6-40. Z2_GPREG3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPREG3																															
R-0h																															

Table 6-45. Z2_GPREG3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPREG3	R	0h	This field gets loaded with the contents of Z2OTP_GPREG3 locations in Zone-2-USER-OTP when a dummy read is issued to that address. Users can use this register to load any general purpose non-voliate content from USER-OTP of Zone-2 Reset type: SYSRSn

6.9.4.7 Z2_GPREG4 Register (Offset (x8) = 1Ch, Offset (x16) = Eh) [Reset = 0h]

Z2_GPREG4 is shown in Figure 6-41 and described in Table 6-46.

Return to the Summary Table.

Zone 2 General Purpose Register-4

Figure 6-41. Z2_GPREG4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPREG4																															
R-0h																															

Table 6-46. Z2_GPREG4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPREG4	R	0h	This field gets loaded with the contents of Z2OTP_GPREG4 locations in Zone-2-USER-OTP when a dummy read is issued to that address. Users can use this register to load any general purpose non-voliate content from USER-OTP of Zone-2 Reset type: SYSRSn

6.9.4.8 Z2_CSMKEY0 Register (Offset (x8) = 20h, Offset (x16) = 10h) [Reset = 0h]

Z2_CSMKEY0 is shown in Figure 6-42 and described in Table 6-47.

Return to the Summary Table.

Zone 2 CSM Key 0

Figure 6-42. Z2_CSMKEY0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2_CSMKEY0																															
R/W-0h																															

Table 6-47. Z2_CSMKEY0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2_CSMKEY0	R/W	0h	To unlock Zone2, user needs to write this regsiter with exact value as Z2_CSMPSWD0, programmed in OTP (zone gets unlock only if 128 bit password in OTP match with value written in four CSMKEY registers.) Reset type: SYSRSn

6.9.4.9 Z2_CSMKEY1 Register (Offset (x8) = 24h, Offset (x16) = 12h) [Reset = 0h]

Z2_CSMKEY1 is shown in Figure 6-43 and described in Table 6-48.

Return to the Summary Table.

Zone 2 CSM Key 1

Figure 6-43. Z2_CSMKEY1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2_CSMKEY1																															
R/W-0h																															

Table 6-48. Z2_CSMKEY1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2_CSMKEY1	R/W	0h	To unlock Zone2, user needs to write this regsiter with exact value as Z2_CSMPSWD1, programmed in OTP (zone gets unlock only if 128 bit password in OTP match with value written in four CSMKEY registers.) Reset type: SYSRSn

6.9.4.10 Z2_CSMKEY2 Register (Offset (x8) = 28h, Offset (x16) = 14h) [Reset = 0h]

Z2_CSMKEY2 is shown in Figure 6-44 and described in Table 6-49.

Return to the Summary Table.

Zone 2 CSM Key 2

Figure 6-44. Z2_CSMKEY2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2_CSMKEY2																															
R/W-0h																															

Table 6-49. Z2_CSMKEY2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2_CSMKEY2	R/W	0h	To unlock Zone2, user needs to write this regsiter with exact value as Z2_CSMPSWD2, programmed in OTP (zone gets unlock only if 128 bit password in OTP match with value written in four CSMKEY registers.) Reset type: SYSRSn

6.9.4.11 Z2_CSMKEY3 Register (Offset (x8) = 2Ch, Offset (x16) = 16h) [Reset = 0h]

Z2_CSMKEY3 is shown in Figure 6-45 and described in Table 6-50.

Return to the Summary Table.

Zone 2 CSM Key 3

Figure 6-45. Z2_CSMKEY3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2_CSMKEY3																															
R/W-0h																															

Table 6-50. Z2_CSMKEY3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2_CSMKEY3	R/W	0h	To unlock Zone2, user needs to write this regsiter with exact value as Z2_CSMPSWD3, programmed in OTP (zone gets unlock only if 128 bit password in OTP match with value written in four CSMKEY registers.) Reset type: SYSRSn

6.9.4.12 Z2_CR Register (Offset (x8) = 30h, Offset (x16) = 18h) [Reset = 00080000h]

Z2_CR is shown in Figure 6-46 and described in Table 6-51.

Return to the Summary Table.

Zone 2 CSM Control Register

Figure 6-46. Z2_CR Register

31	30	29	28	27	26	25	24
FORCESEC	RESERVED						
R-0/W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED	ARMED	UNSECURE	ALLONE	ALLZERO	RESERVED		
R-0h	R-0h	R-0h	R-0h	R-1h	R-0h		
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	RESERVED	RESERVED
R-0h				R-0h	R-0h	R-0h	R-0h

Table 6-51. Z2_CR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	FORCESEC	R-0/W	0h	A write '1' to this fields resets the state of zone. If zone is unlocked, it'll lock(secure) the zone and also resets all the bits in this register. Reset type: SYSRSn
30-24	RESERVED	R	0h	Reserved
23	RESERVED	R	0h	Reserved
22	ARMED	R	0h	0 : Dummy read to CSM Password locations in OTP hasn't been performed. 1 : Dummy read to CSM Password locations in OTP has been performed. Reset type: SYSRSn
21	UNSECURE	R	0h	Indiacates the state of Zone. 0 : Zone is in lock(secure) state. 1 : Zone is in unlock(unsecure) state. Reset type: SYSRSn
20	ALLONE	R	0h	Indicates the state of CSM passwords. 0 : Zone CSM Passwords are not all ones. 1 : Zone CSM Passwords are all ones. Reset type: SYSRSn
19	ALLZERO	R	1h	Indicates the state of CSM passowrds. 0 : CSM Passwords are not all zeros. 1 : CSM Passwords are all zero and device is permanently locked. Reset type: SYSRSn
18-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

6.9.4.13 Z2_GRABSECT1R Register (Offset (x8) = 34h, Offset (x16) = 1Ah) [Reset = 0h]

Z2_GRABSECT1R is shown in Figure 6-47 and described in Table 6-52.

Return to the Summary Table.

Zone 2 Grab Flash Status Register 1

Figure 6-47. Z2_GRABSECT1R Register

31	30	29	28	27	26	25	24
RESERVED				GRAB_SECT13		GRAB_SECT12	
R-0h				R-0h		R-0h	
23	22	21	20	19	18	17	16
GRAB_SECT11		GRAB_SECT10		GRAB_SECT9		GRAB_SECT8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
GRAB_SECT7		GRAB_SECT6		GRAB_SECT5		GRAB_SECT4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
GRAB_SECT3		GRAB_SECT2		GRAB_SECT1		GRAB_SECT0	
R-0h		R-0h		R-0h		R-0h	

Table 6-52. Z2_GRABSECT1R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-26	GRAB_SECT13	R	0h	Value in this field gets loaded from Z2_GRABSECT1[27:26] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 13 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 13 to Zone2. 10 : No request for CPU1 Flash Sector 13 11 : No request for CPU1 Flash Sector 13 when this zone is UNLOCKED. Else CPU1 Flash Sector 13 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
25-24	GRAB_SECT12	R	0h	Value in this field gets loaded from Z2_GRABSECT1[25:24] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 12 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 12 to Zone2. 10 : No request for CPU1 Flash Sector 12 11 : No request for CPU1 Flash Sector 12 when this zone is UNLOCKED. Else CPU1 Flash Sector 12 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
23-22	GRAB_SECT11	R	0h	Value in this field gets loaded from Z2_GRABSECT1[23:22] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 11 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 11 to Zone2. 10 : No request for CPU1 Flash Sector 11 11 : No request for CPU1 Flash Sector 11 when this zone is UNLOCKED. Else CPU1 Flash Sector 11 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-52. Z2_GRABSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	GRAB_SECT10	R	0h	Value in this field gets loaded from Z2_GRABSECT1[21:20] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 10 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 10 to Zone2. 10 : No request for CPU1 Flash Sector 10 11 : No request for CPU1 Flash Sector 10 when this zone is UNLOCKED. Else CPU1 Flash Sector 10 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
19-18	GRAB_SECT9	R	0h	Value in this field gets loaded from Z2_GRABSECT1[19:18] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 9 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 9 to Zone2. 10 : No request for CPU1 Flash Sector 9 11 : No request for CPU1 Flash Sector 9 when this zone is UNLOCKED. Else CPU1 Flash Sector 9 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
17-16	GRAB_SECT8	R	0h	Value in this field gets loaded from Z2_GRABSECT1[17:16] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 8 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 8 to Zone2. 10 : No request for CPU1 Flash Sector 8 11 : No request for CPU1 Flash Sector 8 when this zone is UNLOCKED. Else CPU1 Flash Sector 8 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
15-14	GRAB_SECT7	R	0h	Value in this field gets loaded from Z2_GRABSECT1[15:14] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 7 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 7 to Zone2. 10 : No request for CPU1 Flash Sector 7 11 : No request for CPU1 Flash Sector 7 when this zone is UNLOCKED. Else CPU1 Flash Sector 7 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
13-12	GRAB_SECT6	R	0h	Value in this field gets loaded from Z2_GRABSECT1[13:12] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 6 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 6 to Zone2. 10 : No request for CPU1 Flash Sector 6 11 : No request for CPU1 Flash Sector 6 when this zone is UNLOCKED. Else CPU1 Flash Sector 6 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
11-10	GRAB_SECT5	R	0h	Value in this field gets loaded from Z2_GRABSECT1[11:10] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 5 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 5 to Zone2. 10 : No request for CPU1 Flash Sector 5 11 : No request for CPU1 Flash Sector 5 when this zone is UNLOCKED. Else CPU1 Flash Sector 5 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_SECT4	R	0h	Value in this field gets loaded from Z2_GRABSECT1[9:8] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 4 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 4 to Zone2. 10 : No request for CPU1 Flash Sector 4 11 : No request for CPU1 Flash Sector 4 when this zone is UNLOCKED. Else CPU1 Flash Sector 4 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-52. Z2_GRABSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	GRAB_SECT3	R	0h	Value in this field gets loaded from Z2_GRABSECT1[7:6] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 3 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 3 to Zone2. 10 : No request for CPU1 Flash Sector 3 11 : No request for CPU1 Flash Sector 3 when this zone is UNLOCKED. Else CPU1 Flash Sector 3 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
5-4	GRAB_SECT2	R	0h	Value in this field gets loaded from Z2_GRABSECT1[5:4] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 2 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 2 to Zone2. 10 : No request for CPU1 Flash Sector 2 11 : No request for CPU1 Flash Sector 2 when this zone is UNLOCKED. Else CPU1 Flash Sector 2 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
3-2	GRAB_SECT1	R	0h	Value in this field gets loaded from Z2_GRABSECT1[3:2] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 1 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 1 to Zone2. 10 : No request for CPU1 Flash Sector 1 11 : No request for CPU1 Flash Sector 1 when this zone is UNLOCKED. Else CPU1 Flash Sector 1 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_SECT0	R	0h	Value in this field gets loaded from Z2_GRABSECT1[1:0] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. CPU1 Flash Sector 0 is inaccessible. 01 : Request to allocate CPU1 Flash Sector 0 to Zone2. 10 : No request for CPU1 Flash Sector 0 11 : No request for CPU1 Flash Sector 0 when this zone is UNLOCKED. Else CPU1 Flash Sector 0 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

6.9.4.14 Z2_GRABSECT2R Register (Offset (x8) = 38h, Offset (x16) = 1Ch) [Reset = 0h]

Z2_GRABSECT2R is shown in Figure 6-48 and described in Table 6-53.

Return to the Summary Table.

Zone 2 Grab Flash Status Register 2

Figure 6-48. Z2_GRABSECT2R Register

31	30	29	28	27	26	25	24
RESERVED				GRAB_SECT13		GRAB_SECT12	
R-0h				R-0h		R-0h	
23	22	21	20	19	18	17	16
GRAB_SECT11		GRAB_SECT10		GRAB_SECT9		GRAB_SECT8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
GRAB_SECT7		GRAB_SECT6		GRAB_SECT5		GRAB_SECT4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
GRAB_SECT3		GRAB_SECT2		GRAB_SECT1		GRAB_SECT0	
R-0h		R-0h		R-0h		R-0h	

Table 6-53. Z2_GRABSECT2R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-26	GRAB_SECT13	R	0h	Value in this field gets loaded from Z2_GRABSECT2[27:26] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 13 is inaccessible. 01 : Request to allocate CM Flash Sector 13 to Zone2. 10 : No request for CM Flash Sector 13 11 : No request for CM Flash Sector 13 when this zone is UNLOCKED. Else CM Flash Sector 13 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
25-24	GRAB_SECT12	R	0h	Value in this field gets loaded from Z2_GRABSECT2[25:24] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 12 is inaccessible. 01 : Request to allocate CM Flash Sector 12 to Zone2. 10 : No request for CM Flash Sector 12 11 : No request for CM Flash Sector 12 when this zone is UNLOCKED. Else CM Flash Sector 12 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
23-22	GRAB_SECT11	R	0h	Value in this field gets loaded from Z2_GRABSECT2[23:22] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 11 is inaccessible. 01 : Request to allocate CM Flash Sector 11 to Zone2. 10 : No request for CM Flash Sector 11 11 : No request for CM Flash Sector 11 when this zone is UNLOCKED. Else CM Flash Sector 11 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-53. Z2_GRABSECT2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	GRAB_SECT10	R	0h	Value in this field gets loaded from Z2_GRABSECT2[21:20] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 10 is inaccessible. 01 : Request to allocate CM Flash Sector 10 to Zone2. 10 : No request for CM Flash Sector 10 11 : No request for CM Flash Sector 10 when this zone is UNLOCKED. Else CM Flash Sector 10 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
19-18	GRAB_SECT9	R	0h	Value in this field gets loaded from Z2_GRABSECT2[19:18] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 9 is inaccessible. 01 : Request to allocate CM Flash Sector 9 to Zone2. 10 : No request for CM Flash Sector 9 11 : No request for CM Flash Sector 9 when this zone is UNLOCKED. Else CM Flash Sector 9 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
17-16	GRAB_SECT8	R	0h	Value in this field gets loaded from Z2_GRABSECT2[17:16] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 8 is inaccessible. 01 : Request to allocate CM Flash Sector 8 to Zone2. 10 : No request for CM Flash Sector 8 11 : No request for CM Flash Sector 8 when this zone is UNLOCKED. Else CM Flash Sector 8 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
15-14	GRAB_SECT7	R	0h	Value in this field gets loaded from Z2_GRABSECT2[15:14] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 7 is inaccessible. 01 : Request to allocate CM Flash Sector 7 to Zone2. 10 : No request for CM Flash Sector 7 11 : No request for CM Flash Sector 7 when this zone is UNLOCKED. Else CM Flash Sector 7 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
13-12	GRAB_SECT6	R	0h	Value in this field gets loaded from Z2_GRABSECT2[13:12] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 6 is inaccessible. 01 : Request to allocate CM Flash Sector 6 to Zone2. 10 : No request for CM Flash Sector 6 11 : No request for CM Flash Sector 6 when this zone is UNLOCKED. Else CM Flash Sector 6 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
11-10	GRAB_SECT5	R	0h	Value in this field gets loaded from Z2_GRABSECT2[11:10] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 5 is inaccessible. 01 : Request to allocate CM Flash Sector 5 to Zone2. 10 : No request for CM Flash Sector 5 11 : No request for CM Flash Sector 5 when this zone is UNLOCKED. Else CM Flash Sector 5 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_SECT4	R	0h	Value in this field gets loaded from Z2_GRABSECT2[9:8] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 4 is inaccessible. 01 : Request to allocate CM Flash Sector 4 to Zone2. 10 : No request for CM Flash Sector 4 11 : No request for CM Flash Sector 4 when this zone is UNLOCKED. Else CM Flash Sector 4 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-53. Z2_GRABSECT2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	GRAB_SECT3	R	0h	Value in this field gets loaded from Z2_GRABSECT2[7:6] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 3 is inaccessible. 01 : Request to allocate CM Flash Sector 3 to Zone2. 10 : No request for CM Flash Sector 3 11 : No request for CM Flash Sector 3 when this zone is UNLOCKED. Else CM Flash Sector 3 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
5-4	GRAB_SECT2	R	0h	Value in this field gets loaded from Z2_GRABSECT2[5:4] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 2 is inaccessible. 01 : Request to allocate CM Flash Sector 2 to Zone2. 10 : No request for CM Flash Sector 2 11 : No request for CM Flash Sector 2 when this zone is UNLOCKED. Else CM Flash Sector 2 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
3-2	GRAB_SECT1	R	0h	Value in this field gets loaded from Z2_GRABSECT2[3:2] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 1 is inaccessible. 01 : Request to allocate CM Flash Sector 1 to Zone2. 10 : No request for CM Flash Sector 1 11 : No request for CM Flash Sector 1 when this zone is UNLOCKED. Else CM Flash Sector 1 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_SECT0	R	0h	Value in this field gets loaded from Z2_GRABSECT2[1:0] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. CM Flash Sector 0 is inaccessible. 01 : Request to allocate CM Flash Sector 0 to Zone2. 10 : No request for CM Flash Sector 0 11 : No request for CM Flash Sector 0 when this zone is UNLOCKED. Else CM Flash Sector 0 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

6.9.4.15 Z2_GRABSECT3R Register (Offset (x8) = 3Ch, Offset (x16) = 1Eh) [Reset = 0h]

Z2_GRABSECT3R is shown in Figure 6-49 and described in Table 6-54.

Return to the Summary Table.

Zone 2 Grab Flash Status Register 3

Figure 6-49. Z2_GRABSECT3R Register

31	30	29	28	27	26	25	24
RESERVED				GRAB_SECT13		GRAB_SECT12	
R-0h				R-0h		R-0h	
23	22	21	20	19	18	17	16
GRAB_SECT11		GRAB_SECT10		GRAB_SECT9		GRAB_SECT8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
GRAB_SECT7		GRAB_SECT6		GRAB_SECT5		GRAB_SECT4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
GRAB_SECT3		GRAB_SECT2		GRAB_SECT1		GRAB_SECT0	
R-0h		R-0h		R-0h		R-0h	

Table 6-54. Z2_GRABSECT3R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-26	GRAB_SECT13	R	0h	Value in this field gets loaded from Z2_GRABSECT3[27:26] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 13 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 13 to Zone2. 10 : No request for CPU2 Flash Sector 13 11 : No request for CPU2 Flash Sector 13 when this zone is UNLOCKED. Else CPU2 Flash Sector 13 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
25-24	GRAB_SECT12	R	0h	Value in this field gets loaded from Z2_GRABSECT3[25:24] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 12 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 12 to Zone2. 10 : No request for CPU2 Flash Sector 12 11 : No request for CPU2 Flash Sector 12 when this zone is UNLOCKED. Else CPU2 Flash Sector 12 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
23-22	GRAB_SECT11	R	0h	Value in this field gets loaded from Z2_GRABSECT3[23:22] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 11 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 11 to Zone2. 10 : No request for CPU2 Flash Sector 11 11 : No request for CPU2 Flash Sector 11 when this zone is UNLOCKED. Else CPU2 Flash Sector 11 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-54. Z2_GRABSECT3R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	GRAB_SECT10	R	0h	Value in this field gets loaded from Z2_GRABSECT3[21:20] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 10 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 10 to Zone2. 10 : No request for CPU2 Flash Sector 10 11 : No request for CPU2 Flash Sector 10 when this zone is UNLOCKED. Else CPU2 Flash Sector 10 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
19-18	GRAB_SECT9	R	0h	Value in this field gets loaded from Z2_GRABSECT3[19:18] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 9 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 9 to Zone2. 10 : No request for CPU2 Flash Sector 9 11 : No request for CPU2 Flash Sector 9 when this zone is UNLOCKED. Else CPU2 Flash Sector 9 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
17-16	GRAB_SECT8	R	0h	Value in this field gets loaded from Z2_GRABSECT3[17:16] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 8 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 8 to Zone2. 10 : No request for CPU2 Flash Sector 8 11 : No request for CPU2 Flash Sector 8 when this zone is UNLOCKED. Else CPU2 Flash Sector 8 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
15-14	GRAB_SECT7	R	0h	Value in this field gets loaded from Z2_GRABSECT3[15:14] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 7 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 7 to Zone2. 10 : No request for CPU2 Flash Sector 7 11 : No request for CPU2 Flash Sector 7 when this zone is UNLOCKED. Else CPU2 Flash Sector 7 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
13-12	GRAB_SECT6	R	0h	Value in this field gets loaded from Z2_GRABSECT3[13:12] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 6 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 6 to Zone2. 10 : No request for CPU2 Flash Sector 6 11 : No request for CPU2 Flash Sector 6 when this zone is UNLOCKED. Else CPU2 Flash Sector 6 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
11-10	GRAB_SECT5	R	0h	Value in this field gets loaded from Z2_GRABSECT3[11:10] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 5 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 5 to Zone2. 10 : No request for CPU2 Flash Sector 5 11 : No request for CPU2 Flash Sector 5 when this zone is UNLOCKED. Else CPU2 Flash Sector 5 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_SECT4	R	0h	Value in this field gets loaded from Z2_GRABSECT3[9:8] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 4 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 4 to Zone2. 10 : No request for CPU2 Flash Sector 4 11 : No request for CPU2 Flash Sector 4 when this zone is UNLOCKED. Else CPU2 Flash Sector 4 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-54. Z2_GRABSECT3R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	GRAB_SECT3	R	0h	Value in this field gets loaded from Z2_GRABSECT3[7:6] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 3 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 3 to Zone2. 10 : No request for CPU2 Flash Sector 3 11 : No request for CPU2 Flash Sector 3 when this zone is UNLOCKED. Else CPU2 Flash Sector 3 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
5-4	GRAB_SECT2	R	0h	Value in this field gets loaded from Z2_GRABSECT3[5:4] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 2 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 2 to Zone2. 10 : No request for CPU2 Flash Sector 2 11 : No request for CPU2 Flash Sector 2 when this zone is UNLOCKED. Else CPU2 Flash Sector 2 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
3-2	GRAB_SECT1	R	0h	Value in this field gets loaded from Z2_GRABSECT3[3:2] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 1 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 1 to Zone2. 10 : No request for CPU2 Flash Sector 1 11 : No request for CPU2 Flash Sector 1 when this zone is UNLOCKED. Else CPU2 Flash Sector 1 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_SECT0	R	0h	Value in this field gets loaded from Z2_GRABSECT3[1:0] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. CPU2 Flash Sector 0 is inaccessible. 01 : Request to allocate CPU2 Flash Sector 0 to Zone2. 10 : No request for CPU2 Flash Sector 0 11 : No request for CPU2 Flash Sector 0 when this zone is UNLOCKED. Else CPU2 Flash Sector 0 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

6.9.4.16 Z2_GRABRAM1R Register (Offset (x8) = 40h, Offset (x16) = 20h) [Reset = 0h]

Z2_GRABRAM1R is shown in Figure 6-50 and described in Table 6-55.

Return to the Summary Table.

Zone 2 Grab RAM Status Register 1

Figure 6-50. Z2_GRABRAM1R Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				GRAB_RAM9		GRAB_RAM8	
R-0h				R-0h		R-0h	
15	14	13	12	11	10	9	8
GRAB_RAM7		GRAB_RAM6		GRAB_RAM5		GRAB_RAM4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
GRAB_RAM3		GRAB_RAM2		GRAB_RAM1		GRAB_RAM0	
R-0h		R-0h		R-0h		R-0h	

Table 6-55. Z2_GRABRAM1R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-18	GRAB_RAM9	R	0h	Value in this field gets loaded from Z2_GRABRAM1[19:18] when a read is issued to address location of Z2_GRABRAM1 in OTP. 00 : Invalid. CPU1 D1 RAM is inaccessible. 01 : Request to allocate CPU1 D1 RAM to Zone2. 10 : No request for CPU1 D1 RAM 11 : No request for CPU1 D1 RAM when this zone is UNLOCKED. Else CPU1 D1 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
17-16	GRAB_RAM8	R	0h	Value in this field gets loaded from Z2_GRABRAM1[17:16] when a read is issued to address location of Z2_GRABRAM1 in OTP. 00 : Invalid. CPU1 D0 RAM is inaccessible. 01 : Request to allocate CPU1 D0 RAM to Zone2. 10 : No request for CPU1 D0 RAM 11 : No request for CPU1 D0 RAM when this zone is UNLOCKED. Else CPU1 D0 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
15-14	GRAB_RAM7	R	0h	Value in this field gets loaded from Z2_GRABRAM1[15:14] when a read is issued to address location of Z2_GRABRAM1 in OTP. 00 : Invalid. CPU1 LS7 RAM is inaccessible. 01 : Request to allocate CPU1 LS7 RAM to Zone2. 10 : No request for CPU1 LS7 RAM 11 : No request for CPU1 LS7 RAM when this zone is UNLOCKED. Else CPU1 LS7 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
13-12	GRAB_RAM6	R	0h	Value in this field gets loaded from Z2_GRABRAM1[13:12] when a read is issued to address location of Z2_GRABRAM1 in OTP. 00 : Invalid. CPU1 LS6 RAM is inaccessible. 01 : Request to allocate CPU1 LS6 RAM to Zone2. 10 : No request for CPU1 LS6 RAM 11 : No request for CPU1 LS6 RAM when this zone is UNLOCKED. Else CPU1 LS6 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-55. Z2_GRABRAM1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11-10	GRAB_RAM5	R	0h	Value in this field gets loaded from Z2_GRABRAM1[11:10] when a read is issued to address location of Z2_GRABRAM1 in OTP. 00 : Invalid. CPU1 LS5 RAM is inaccessible. 01 : Request to allocate CPU1 LS5 RAM to Zone2. 10 : No request for CPU1 LS5 RAM 11 : No request for CPU1 LS5 RAM when this zone is UNLOCKED. Else CPU1 LS5 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_RAM4	R	0h	Value in this field gets loaded from Z2_GRABRAM1[9:8] when a read is issued to address location of Z2_GRABRAM1 in OTP. 00 : Invalid. CPU1 LS4 RAM is inaccessible. 01 : Request to allocate CPU1 LS4 RAM to Zone2. 10 : No request for CPU1 LS4 RAM 11 : No request for CPU1 LS4 RAM when this zone is UNLOCKED. Else CPU1 LS4 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
7-6	GRAB_RAM3	R	0h	Value in this field gets loaded from Z2_GRABRAM1[7:6] when a read is issued to address location of Z2_GRABRAM1 in OTP. 00 : Invalid. CPU1 LS3 RAM is inaccessible. 01 : Request to allocate CPU1 LS3 RAM to Zone2. 10 : No request for CPU1 LS3 RAM 11 : No request for CPU1 LS3 RAM when this zone is UNLOCKED. Else CPU1 LS3 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
5-4	GRAB_RAM2	R	0h	Value in this field gets loaded from Z2_GRABRAM1[5:4] when a read is issued to address location of Z2_GRABRAM1 in OTP. 00 : Invalid. CPU1 LS2 RAM is inaccessible. 01 : Request to allocate CPU1 LS2 RAM to Zone2. 10 : No request for CPU1 LS2 RAM 11 : No request for CPU1 LS2 RAM when this zone is UNLOCKED. Else CPU1 LS2 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
3-2	GRAB_RAM1	R	0h	Value in this field gets loaded from Z2_GRABRAM1[3:2] when a read is issued to address location of Z2_GRABRAM1 in OTP. 00 : Invalid. CPU1 LS1 RAM is inaccessible. 01 : Request to allocate CPU1 LS1 RAM to Zone2. 10 : No request for CPU1 LS1 RAM 11 : No request for CPU1 LS1 RAM when this zone is UNLOCKED. Else CPU1 LS1 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_RAM0	R	0h	Value in this field gets loaded from Z2_GRABRAM1[1:0] when a read is issued to address location of Z2_GRABRAM1 in OTP. 00 : Invalid. CPU1 LS0 RAM is inaccessible. 01 : Request to allocate CPU1 LS0 RAM to Zone2. 10 : No request for CPU1 LS0 RAM 11 : No request for CPU1 LS0 RAM when this zone is UNLOCKED. Else CPU1 LS0 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn

6.9.4.17 Z2_GRABRAM2R Register (Offset (x8) = 44h, Offset (x16) = 22h) [Reset = 0h]

Z2_GRABRAM2R is shown in Figure 6-51 and described in Table 6-56.

Return to the Summary Table.

Zone 2 Grab RAM Status Register 2

Figure 6-51. Z2_GRABRAM2R Register

31	30	29	28	27	26	25	24
GRAB_RAM15		GRAB_RAM14		GRAB_RAM13		GRAB_RAM12	
R-0h		R-0h		R-0h		R-0h	
23	22	21	20	19	18	17	16
GRAB_RAM11		GRAB_RAM10		GRAB_RAM9		GRAB_RAM8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
GRAB_RAM7		GRAB_RAM6		GRAB_RAM5		GRAB_RAM4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
RESERVED				GRAB_RAM1		GRAB_RAM0	
R-0h				R-0h		R-0h	

Table 6-56. Z2_GRABRAM2R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GRAB_RAM15	R	0h	Value in this field gets loaded from Z2_GRABRAM2[31:30] when a read is issued to address location of Z2_GRABRAM2 in OTP. Defines Zone 2's grab request of the higher addressed half of CPU2TOCPU1MSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone2. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
29-28	GRAB_RAM14	R	0h	Value in this field gets loaded from Z2_GRABRAM2[29:28] when a read is issued to address location of Z2_GRABRAM2 in OTP. Defines Zone 2's grab request of the lower addressed half of CPU2TOCPU1MSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone2. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
27-26	GRAB_RAM13	R	0h	Value in this field gets loaded from Z2_GRABRAM2[27:26] when a read is issued to address location of Z2_GRABRAM2 in OTP. Defines Zone 2's grab request of the higher addressed half of CPU1TOCPU2MSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone2. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-56. Z2_GRABRAM2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-24	GRAB_RAM12	R	0h	Value in this field gets loaded from Z2_GRABRAM2[25:24] when a read is issued to address location of Z2_GRABRAM2 in OTP. Defines Zone 2's grab request of the lower addressed half of CPU1TOCPU2MSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone2. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
23-22	GRAB_RAM11	R	0h	Value in this field gets loaded from Z2_GRABRAM2[23:22] when a read is issued to address location of Z2_GRABRAM2 in OTP. Defines Zone 2's grab request of the higher addressed half of CMTOCPU2MSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone2. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
21-20	GRAB_RAM10	R	0h	Value in this field gets loaded from Z2_GRABRAM2[21:20] when a read is issued to address location of Z2_GRABRAM2 in OTP. Defines Zone 2's grab request of the lower addressed half of CMTOCPU2MSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone2. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
19-18	GRAB_RAM9	R	0h	Value in this field gets loaded from Z2_GRABRAM2[19:18] when a read is issued to address location of Z2_GRABRAM2 in OTP. Defines Zone 2's grab request of the higher addressed half of CPU2TOCMMSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone2. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
17-16	GRAB_RAM8	R	0h	Value in this field gets loaded from Z2_GRABRAM2[17:16] when a read is issued to address location of Z2_GRABRAM2 in OTP. Defines Zone 2's grab request of the lower addressed half of CPU2TOCMMSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone2. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
15-14	GRAB_RAM7	R	0h	Value in this field gets loaded from Z2_GRABRAM2[15:14] when a read is issued to address location of Z2_GRABRAM2 in OTP. Defines Zone 2's grab request of the higher addressed half of CMTOCPU1MSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone2. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-56. Z2_GRABRAM2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13-12	GRAB_RAM6	R	0h	Value in this field gets loaded from Z2_GRABRAM2[13:12] when a read is issued to address location of Z2_GRABRAM2 in OTP. Defines Zone 2's grab request of the lower addressed half of CMTOCPU1MSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone2. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
11-10	GRAB_RAM5	R	0h	Value in this field gets loaded from Z2_GRABRAM2[11:10] when a read is issued to address location of Z2_GRABRAM2 in OTP. Defines Zone 2's grab request of the higher addressed half of CPU1TOCMMSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone2. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_RAM4	R	0h	Value in this field gets loaded from Z2_GRABRAM2[9:8] when a read is issued to address location of Z2_GRABRAM2 in OTP. Defines Zone 2's grab request of the lower addressed half of CPU1TOCMMSGRAM0. 00 : Invalid. MSG RAM is inaccessible. 01 : Request to allocate MSG RAM to Zone2. 10 : No request for MSG RAM 11 : No request for MSG RAM when this zone is UNLOCKED. Else MSG RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
7-4	RESERVED	R	0h	Reserved
3-2	GRAB_RAM1	R	0h	Value in this field gets loaded from Z2_GRABRAM2[3:2] when a read is issued to address location of Z2_GRABRAM2 in OTP. 00 : Invalid. CM C1 RAM is inaccessible. 01 : Request to allocate CM C1 RAM to Zone2. 10 : No request for CM C1 RAM 11 : No request for CM C1 RAM when this zone is UNLOCKED. Else CM C1 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_RAM0	R	0h	Value in this field gets loaded from Z2_GRABRAM2[1:0] when a read is issued to address location of Z2_GRABRAM2 in OTP. 00 : Invalid. CM C0 RAM is inaccessible. 01 : Request to allocate CM C0 RAM to Zone2. 10 : No request for CM C0 RAM 11 : No request for CM C0 RAM when this zone is UNLOCKED. Else CM C0 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn

6.9.4.18 Z2_GRABRAM3R Register (Offset (x8) = 48h, Offset (x16) = 24h) [Reset = 0h]

Z2_GRABRAM3R is shown in Figure 6-52 and described in Table 6-57.

Return to the Summary Table.

Zone 2 Grab RAM Status Register 3

Figure 6-52. Z2_GRABRAM3R Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				GRAB_RAM9		GRAB_RAM8	
R-0h				R-0h		R-0h	
15	14	13	12	11	10	9	8
GRAB_RAM7		GRAB_RAM6		GRAB_RAM5		GRAB_RAM4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
GRAB_RAM3		GRAB_RAM2		GRAB_RAM1		GRAB_RAM0	
R-0h		R-0h		R-0h		R-0h	

Table 6-57. Z2_GRABRAM3R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-18	GRAB_RAM9	R	0h	Value in this field gets loaded from Z2_GRABRAM3[19:18] when a read is issued to address location of Z2_GRABRAM3 in OTP. 00 : Invalid. CPU2 D1 RAM is inaccessible. 01 : Request to allocate CPU2 D1 RAM to Zone2. 10 : No request for CPU2 D1 RAM 11 : No request for CPU2 D1 RAM when this zone is UNLOCKED. Else CPU2 D1 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
17-16	GRAB_RAM8	R	0h	Value in this field gets loaded from Z2_GRABRAM3[17:16] when a read is issued to address location of Z2_GRABRAM3 in OTP. 00 : Invalid. CPU2 D0 RAM is inaccessible. 01 : Request to allocate CPU2 D0 RAM to Zone2. 10 : No request for CPU2 D0 RAM 11 : No request for CPU2 D0 RAM when this zone is UNLOCKED. Else CPU2 D0 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
15-14	GRAB_RAM7	R	0h	Value in this field gets loaded from Z2_GRABRAM3[15:14] when a read is issued to address location of Z2_GRABRAM3 in OTP. 00 : Invalid. CPU2 LS7 RAM is inaccessible. 01 : Request to allocate CPU2 LS7 RAM to Zone2. 10 : No request for CPU2 LS7 RAM 11 : No request for CPU2 LS7 RAM when this zone is UNLOCKED. Else CPU2 LS7 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
13-12	GRAB_RAM6	R	0h	Value in this field gets loaded from Z2_GRABRAM3[13:12] when a read is issued to address location of Z2_GRABRAM3 in OTP. 00 : Invalid. CPU2 LS6 RAM is inaccessible. 01 : Request to allocate CPU2 LS6 RAM to Zone2. 10 : No request for CPU2 LS6 RAM 11 : No request for CPU2 LS6 RAM when this zone is UNLOCKED. Else CPU2 LS6 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 6-57. Z2_GRABRAM3R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11-10	GRAB_RAM5	R	0h	Value in this field gets loaded from Z2_GRABRAM3[11:10] when a read is issued to address location of Z2_GRABRAM3 in OTP. 00 : Invalid. CPU2 LS5 RAM is inaccessible. 01 : Request to allocate CPU2 LS5 RAM to Zone2. 10 : No request for CPU2 LS5 RAM 11 : No request for CPU2 LS5 RAM when this zone is UNLOCKED. Else CPU2 LS5 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_RAM4	R	0h	Value in this field gets loaded from Z2_GRABRAM3[9:8] when a read is issued to address location of Z2_GRABRAM3 in OTP. 00 : Invalid. CPU2 LS4 RAM is inaccessible. 01 : Request to allocate CPU2 LS4 RAM to Zone2. 10 : No request for CPU2 LS4 RAM 11 : No request for CPU2 LS4 RAM when this zone is UNLOCKED. Else CPU2 LS4 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
7-6	GRAB_RAM3	R	0h	Value in this field gets loaded from Z2_GRABRAM3[7:6] when a read is issued to address location of Z2_GRABRAM3 in OTP. 00 : Invalid. CPU2 LS3 RAM is inaccessible. 01 : Request to allocate CPU2 LS3 RAM to Zone2. 10 : No request for CPU2 LS3 RAM 11 : No request for CPU2 LS3 RAM when this zone is UNLOCKED. Else CPU2 LS3 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
5-4	GRAB_RAM2	R	0h	Value in this field gets loaded from Z2_GRABRAM3[5:4] when a read is issued to address location of Z2_GRABRAM3 in OTP. 00 : Invalid. CPU2 LS2 RAM is inaccessible. 01 : Request to allocate CPU2 LS2 RAM to Zone2. 10 : No request for CPU2 LS2 RAM 11 : No request for CPU2 LS2 RAM when this zone is UNLOCKED. Else CPU2 LS2 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
3-2	GRAB_RAM1	R	0h	Value in this field gets loaded from Z2_GRABRAM3[3:2] when a read is issued to address location of Z2_GRABRAM3 in OTP. 00 : Invalid. CPU2 LS1 RAM is inaccessible. 01 : Request to allocate CPU2 LS1 RAM to Zone2. 10 : No request for CPU2 LS1 RAM 11 : No request for CPU2 LS1 RAM when this zone is UNLOCKED. Else CPU2 LS1 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_RAM0	R	0h	Value in this field gets loaded from Z2_GRABRAM3[1:0] when a read is issued to address location of Z2_GRABRAM3 in OTP. 00 : Invalid. CPU2 LS0 RAM is inaccessible. 01 : Request to allocate CPU2 LS0 RAM to Zone2. 10 : No request for CPU2 LS0 RAM 11 : No request for CPU2 LS0 RAM when this zone is UNLOCKED. Else CPU2 LS0 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn

6.9.4.19 Z2_EXEONLYSECT1R Register (Offset (x8) = 4Ch, Offset (x16) = 26h) [Reset = 0h]

Z2_EXEONLYSECT1R is shown in Figure 6-53 and described in Table 6-58.

Return to the Summary Table.

Zone 2 Execute Only Flash Status Register 1

Figure 6-53. Z2_EXEONLYSECT1R Register

31	30	29	28	27	26	25	24
RESERVED		EXEONLY_CM_SECT13	EXEONLY_CM_SECT12	EXEONLY_CM_SECT11	EXEONLY_CM_SECT10	EXEONLY_CM_SECT9	EXEONLY_CM_SECT8
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
EXEONLY_CM_SECT7	EXEONLY_CM_SECT6	EXEONLY_CM_SECT5	EXEONLY_CM_SECT4	EXEONLY_CM_SECT3	EXEONLY_CM_SECT2	EXEONLY_CM_SECT1	EXEONLY_CM_SECT0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED		EXEONLY_CPU1_SECT13	EXEONLY_CPU1_SECT12	EXEONLY_CPU1_SECT11	EXEONLY_CPU1_SECT10	EXEONLY_CPU1_SECT9	EXEONLY_CPU1_SECT8
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
EXEONLY_CPU1_SECT7	EXEONLY_CPU1_SECT6	EXEONLY_CPU1_SECT5	EXEONLY_CPU1_SECT4	EXEONLY_CPU1_SECT3	EXEONLY_CPU1_SECT2	EXEONLY_CPU1_SECT1	EXEONLY_CPU1_SECT0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 6-58. Z2_EXEONLYSECT1R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29	EXEONLY_CM_SECT13	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[29] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 13 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CM Flash Sector 13 (only if it's allocated to Zone2) Reset type: SYSRSn
28	EXEONLY_CM_SECT12	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[28] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 12 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CM Flash Sector 12 (only if it's allocated to Zone2) Reset type: SYSRSn
27	EXEONLY_CM_SECT11	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[27] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 11 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CM Flash Sector 11 (only if it's allocated to Zone2) Reset type: SYSRSn
26	EXEONLY_CM_SECT10	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[26] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 10 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CM Flash Sector 10 (only if it's allocated to Zone2) Reset type: SYSRSn

Table 6-58. Z2_EXEONLYSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	EXEONLY_CM_SECT9	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[25] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 9 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CM Flash Sector 9 (only if it's allocated to Zone2) Reset type: SYSRSn
24	EXEONLY_CM_SECT8	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[24] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 8 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CM Flash Sector 8 (only if it's allocated to Zone2) Reset type: SYSRSn
23	EXEONLY_CM_SECT7	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[23] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 7 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CM Flash Sector 7 (only if it's allocated to Zone2) Reset type: SYSRSn
22	EXEONLY_CM_SECT6	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[22] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 6 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CM Flash Sector 6 (only if it's allocated to Zone2) Reset type: SYSRSn
21	EXEONLY_CM_SECT5	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[21] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 5 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CM Flash Sector 5 (only if it's allocated to Zone2) Reset type: SYSRSn
20	EXEONLY_CM_SECT4	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[20] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 4 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CM Flash Sector 4 (only if it's allocated to Zone2) Reset type: SYSRSn
19	EXEONLY_CM_SECT3	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[19] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 3 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CM Flash Sector 3 (only if it's allocated to Zone2) Reset type: SYSRSn
18	EXEONLY_CM_SECT2	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[18] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 2 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CM Flash Sector 2 (only if it's allocated to Zone2) Reset type: SYSRSn

Table 6-58. Z2_EXEONLYSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	EXEONLY_CM_SECT1	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[17] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 1 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CM Flash Sector 1 (only if it's allocated to Zone2) Reset type: SYSRSn
16	EXEONLY_CM_SECT0	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[16] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CM Flash Sector 0 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CM Flash Sector 0 (only if it's allocated to Zone2) Reset type: SYSRSn
15-14	RESERVED	R	0h	Reserved
13	EXEONLY_CPU1_SECT13	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[13] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 13 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 13 (only if it's allocated to Zone2) Reset type: SYSRSn
12	EXEONLY_CPU1_SECT12	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[12] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 12 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 12 (only if it's allocated to Zone2) Reset type: SYSRSn
11	EXEONLY_CPU1_SECT11	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[11] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 11 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 11 (only if it's allocated to Zone2) Reset type: SYSRSn
10	EXEONLY_CPU1_SECT10	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[10] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 10 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 10 (only if it's allocated to Zone2) Reset type: SYSRSn
9	EXEONLY_CPU1_SECT9	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[9] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 9 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 9 (only if it's allocated to Zone2) Reset type: SYSRSn
8	EXEONLY_CPU1_SECT8	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[8] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 8 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 8 (only if it's allocated to Zone2) Reset type: SYSRSn

Table 6-58. Z2_EXEONLYSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	EXEONLY_CPU1_SECT7	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[7] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 7 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 7 (only if it's allocated to Zone2) Reset type: SYSRSn
6	EXEONLY_CPU1_SECT6	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[6] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 6 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 6 (only if it's allocated to Zone2) Reset type: SYSRSn
5	EXEONLY_CPU1_SECT5	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[5] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 5 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 5 (only if it's allocated to Zone2) Reset type: SYSRSn
4	EXEONLY_CPU1_SECT4	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[4] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 4 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 4 (only if it's allocated to Zone2) Reset type: SYSRSn
3	EXEONLY_CPU1_SECT3	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[3] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 3 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 3 (only if it's allocated to Zone2) Reset type: SYSRSn
2	EXEONLY_CPU1_SECT2	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[2] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 2 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 2 (only if it's allocated to Zone2) Reset type: SYSRSn
1	EXEONLY_CPU1_SECT1	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[1] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 1 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 1 (only if it's allocated to Zone2) Reset type: SYSRSn
0	EXEONLY_CPU1_SECT0	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[0] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 Flash Sector 0 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 Flash Sector 0 (only if it's allocated to Zone2) Reset type: SYSRSn

6.9.4.20 Z2_EXEONLYSECT2R Register (Offset (x8) = 50h, Offset (x16) = 28h) [Reset = 0h]

Z2_EXEONLYSECT2R is shown in Figure 6-54 and described in Table 6-59.

Return to the Summary Table.

Zone 2 Execute Only Flash Status Register 2

Figure 6-54. Z2_EXEONLYSECT2R Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		EXEONLY_CPU2_SECT13	EXEONLY_CPU2_SECT12	EXEONLY_CPU2_SECT11	EXEONLY_CPU2_SECT10	EXEONLY_CPU2_SECT9	EXEONLY_CPU2_SECT8
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
EXEONLY_CPU2_SECT7	EXEONLY_CPU2_SECT6	EXEONLY_CPU2_SECT5	EXEONLY_CPU2_SECT4	EXEONLY_CPU2_SECT3	EXEONLY_CPU2_SECT2	EXEONLY_CPU2_SECT1	EXEONLY_CPU2_SECT0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 6-59. Z2_EXEONLYSECT2R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13	EXEONLY_CPU2_SECT13	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[13] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 13 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 13 (only if it's allocated to Zone2) Reset type: SYSRSn
12	EXEONLY_CPU2_SECT12	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[12] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 12 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 12 (only if it's allocated to Zone2) Reset type: SYSRSn
11	EXEONLY_CPU2_SECT11	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[11] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 11 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 11 (only if it's allocated to Zone2) Reset type: SYSRSn
10	EXEONLY_CPU2_SECT10	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[10] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 10 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 10 (only if it's allocated to Zone2) Reset type: SYSRSn

Table 6-59. Z2_EXEONLYSECT2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	EXEONLY_CPU2_SECT9	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[9] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 9 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 9 (only if it's allocated to Zone2) Reset type: SYSRSn
8	EXEONLY_CPU2_SECT8	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[8] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 8 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 8 (only if it's allocated to Zone2) Reset type: SYSRSn
7	EXEONLY_CPU2_SECT7	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[7] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 7 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 7 (only if it's allocated to Zone2) Reset type: SYSRSn
6	EXEONLY_CPU2_SECT6	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[6] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 6 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 6 (only if it's allocated to Zone2) Reset type: SYSRSn
5	EXEONLY_CPU2_SECT5	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[5] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 5 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 5 (only if it's allocated to Zone2) Reset type: SYSRSn
4	EXEONLY_CPU2_SECT4	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[4] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 4 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 4 (only if it's allocated to Zone2) Reset type: SYSRSn
3	EXEONLY_CPU2_SECT3	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[3] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 3 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 3 (only if it's allocated to Zone2) Reset type: SYSRSn
2	EXEONLY_CPU2_SECT2	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[2] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 2 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 2 (only if it's allocated to Zone2) Reset type: SYSRSn

Table 6-59. Z2_EXEONLYSECT2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	EXEONLY_CPU2_SECT1	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[1] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 1 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 1 (only if it's allocated to Zone2) Reset type: SYSRSn
0	EXEONLY_CPU2_SECT0	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[0] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 Flash Sector 0 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 Flash Sector 0 (only if it's allocated to Zone2) Reset type: SYSRSn

6.9.4.21 Z2_EXEONLYRAM1R Register (Offset (x8) = 54h, Offset (x16) = 2Ah) [Reset = 0h]

Z2_EXEONLYRAM1R is shown in Figure 6-55 and described in Table 6-60.

Return to the Summary Table.

Zone 2 Execute Only RAM Status Register 1

Figure 6-55. Z2_EXEONLYRAM1R Register

31	30	29	28	27	26	25	24
EXEONLY_RAM31	EXEONLY_RAM30	EXEONLY_RAM29	EXEONLY_RAM28	EXEONLY_RAM27	EXEONLY_RAM26	EXEONLY_RAM25	EXEONLY_RAM24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
EXEONLY_RAM23	EXEONLY_RAM22	RESERVED				EXEONLY_RAM17	EXEONLY_RAM16
R-0h	R-0h	R-0h				R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED						EXEONLY_RAM9	EXEONLY_RAM8
R-0h						R-0h	R-0h
7	6	5	4	3	2	1	0
EXEONLY_RAM7	EXEONLY_RAM6	EXEONLY_RAM5	EXEONLY_RAM4	EXEONLY_RAM3	EXEONLY_RAM2	EXEONLY_RAM1	EXEONLY_RAM0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 6-60. Z2_EXEONLYRAM1R Register Field Descriptions

Bit	Field	Type	Reset	Description
31	EXEONLY_RAM31	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[31] when a read is issued to Z2_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CPU2 LS0 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 LS0 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
30	EXEONLY_RAM30	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[30] when a read is issued to Z2_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CPU2 LS1 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 LS1 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
29	EXEONLY_RAM29	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[29] when a read is issued to Z2_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CPU2 LS2 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 LS2 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
28	EXEONLY_RAM28	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[28] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 LS3 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 LS3 RAM (only if it's allocated to Zone2) Reset type: SYSRSn

Table 6-60. Z2_EXEONLYRAM1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	EXEONLY_RAM27	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[27] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 LS4 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 LS4 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
26	EXEONLY_RAM26	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[26] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 LS5 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 LS5 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
25	EXEONLY_RAM25	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[25] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 LS6 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 LS6 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
24	EXEONLY_RAM24	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[24] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU2 LS7 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 LS7 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
23	EXEONLY_RAM23	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[23] when a read is issued to Z2_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CPU2 D0 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 D0 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
22	EXEONLY_RAM22	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[22] when a read is issued to Z2_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CPU2 D1 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU2 D1 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
21-18	RESERVED	R	0h	Reserved
17	EXEONLY_RAM17	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[17] when a read is issued to Z2_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CM C1 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CM C1 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
16	EXEONLY_RAM16	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[16] when a read is issued to Z2_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CM C0 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CM C0 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
15-10	RESERVED	R	0h	Reserved

Table 6-60. Z2_EXEONLYRAM1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	EXEONLY_RAM9	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[9] when a read is issued to Z2_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CPU1 D1 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 D1 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
8	EXEONLY_RAM8	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[8] when a read is issued to Z2_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CPU1 D0 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 D0 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
7	EXEONLY_RAM7	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[7] when a read is issued to Z2_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for CPU1 LS7 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 LS7 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
6	EXEONLY_RAM6	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[6] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 LS6 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 LS6 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
5	EXEONLY_RAM5	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[5] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 LS5 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 LS5 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
4	EXEONLY_RAM4	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[4] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 LS4 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 LS4 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
3	EXEONLY_RAM3	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[3] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 LS3 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 LS3 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
2	EXEONLY_RAM2	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[2] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 LS2 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 LS2 RAM (only if it's allocated to Zone2) Reset type: SYSRSn

Table 6-60. Z2_EXEONLYRAM1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	EXEONLY_RAM1	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[1] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 LS1 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 LS1 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
0	EXEONLY_RAM0	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[0] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for CPU1 LS0 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for CPU1 LS0 RAM (only if it's allocated to Zone2) Reset type: SYSRSn

6.9.5 DCSM_COMMON_REGS Registers

Table 6-61 lists the memory-mapped registers for the DCSM_COMMON_REGS registers. All register offset addresses not listed in Table 6-61 should be considered as reserved locations and the register contents should not be modified.

Table 6-61. DCSM_COMMON_REGS Registers

Offset (x8)	Offset (x16)	Acronym	Register Name	Write Protection	Section
0h	0h	FLSEM	Flash Wrapper Semaphore Register	EALLOW	Go
10h	8h	SECTSTAT1	Flash Sectors Status Register 1		Go
14h	Ah	SECTSTAT2	Flash Sectors Status Register 2		Go
18h	Ch	SECTSTAT3	Flash Sectors Status Register 3		Go
20h	10h	RAMSTAT1	RAM Status Register 1		Go
24h	12h	RAMSTAT2	RAM Status Register 2		Go
28h	14h	RAMSTAT3	RAM Status Register 3		Go
30h	18h	SECERRSTAT	Security Error Status Register		Go
34h	1Ah	SECERRCLR	Security Error Clear Register		Go
38h	1Ch	SECERRFRC	Security Error Force Register		Go

Complex bit access types are encoded to fit into small table cells. Table 6-62 shows the codes that are used for access types in this section.

Table 6-62. DCSM_COMMON_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

6.9.5.1 FLSEM Register (Offset (x8) = 0h, Offset (x16) = 0h) [Reset = 0h]

FLSEM is shown in Figure 6-56 and described in Table 6-63.

Return to the Summary Table.

Flash Wrapper Semaphore Register

Figure 6-56. FLSEM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY								RESERVED						SEM	
R-0/W-0h								R-0h						R/W-0h	

Table 6-63. FLSEM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	KEY	R-0/W	0h	Writing a value 0xA5 into this field will allow the writing of the SEM bits, else writes are ignored. Reads will return 0. Reset type: SYSRSn
7-2	RESERVED	R	0h	Reserved
1-0	SEM	R/W	0h	00 : Flash Wrapper registers can be written by code running from non-secure zone. 01 : Flash Wrapper registers can be written by code running from Zone1 security zone. 10 : Flash Wrapper registers can be written by code running from Zone2 security zone 11 : Flash Wrapper registers can be written by code running from non-secure zone. Allowed State Transitions in this field. 00 TO 11 : Not allowed. 11 TO 00 : Not allowed. 00/11 TO 01 : Code running from Zone1 only can perform this transition. 01 TO 00/11 : Code running from Zone1 only can perform this transition. 00/11 TO 10 : Code running from Zone2 only can perform this transition. 10 TO 00/11 : Code running from Zone2 can perform this transition 10 TO 01 : Not allowed. 01 TO 10 : Not allowed. Reset type: SYSRSn

6.9.5.2 SECTSTAT1 Register (Offset (x8) = 10h, Offset (x16) = 8h) [Reset = 0h]

SECTSTAT1 is shown in Figure 6-57 and described in Table 6-64.

Return to the Summary Table.

Flash Sectors Status Register 1

Figure 6-57. SECTSTAT1 Register

31	30	29	28	27	26	25	24
RESERVED				STATUS_SECT13		STATUS_SECT12	
R-0h				R-0h		R-0h	
23	22	21	20	19	18	17	16
STATUS_SECT11		STATUS_SECT10		STATUS_SECT9		STATUS_SECT8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
STATUS_SECT7		STATUS_SECT6		STATUS_SECT5		STATUS_SECT4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
STATUS_SECT3		STATUS_SECT2		STATUS_SECT1		STATUS_SECT0	
R-0h		R-0h		R-0h		R-0h	

Table 6-64. SECTSTAT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-26	STATUS_SECT13	R	0h	Reflects the status of flash CPU1 BANK Sector 13. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
25-24	STATUS_SECT12	R	0h	Reflects the status of flash CPU1 BANK Sector 12. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
23-22	STATUS_SECT11	R	0h	Reflects the status of flash CPU1 BANK Sector 11. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
21-20	STATUS_SECT10	R	0h	Reflects the status of flash CPU1 BANK Sector 10. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

Table 6-64. SECTSTAT1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	STATUS_SECT9	R	0h	Reflects the status of flash CPU1 BANK Sector 9. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
17-16	STATUS_SECT8	R	0h	Reflects the status of flash CPU1 BANK sector 8. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
15-14	STATUS_SECT7	R	0h	Reflects the status of flash CPU1 BANK Sector 7. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
13-12	STATUS_SECT6	R	0h	Reflects the status of flash CPU1 BANK Sector 6. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
11-10	STATUS_SECT5	R	0h	Reflects the status of flash CPU1 BANK Sector 5. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
9-8	STATUS_SECT4	R	0h	Reflects the status of flash CPU1 BANK Sector 4. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
7-6	STATUS_SECT3	R	0h	Reflects the status of flash CPU1 BANK Sector 3. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
5-4	STATUS_SECT2	R	0h	Reflects the status of flash CPU1 BANK Sector 2. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

Table 6-64. SECTSTAT1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	STATUS_SECT1	R	0h	Reflects the status of flash CPU1 BANK sector 1. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
1-0	STATUS_SECT0	R	0h	Reflects the status of flash CPU1 BANK Sector 0. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

6.9.5.3 SECTSTAT2 Register (Offset (x8) = 14h, Offset (x16) = Ah) [Reset = 0h]

SECTSTAT2 is shown in Figure 6-58 and described in Table 6-65.

Return to the Summary Table.

Flash Sectors Status Register 2

Figure 6-58. SECTSTAT2 Register

31	30	29	28	27	26	25	24
RESERVED				STATUS_SECT13		STATUS_SECT12	
R-0h				R-0h		R-0h	
23	22	21	20	19	18	17	16
STATUS_SECT11		STATUS_SECT10		STATUS_SECT9		STATUS_SECT8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
STATUS_SECT7		STATUS_SECT6		STATUS_SECT5		STATUS_SECT4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
STATUS_SECT3		STATUS_SECT2		STATUS_SECT1		STATUS_SECT0	
R-0h		R-0h		R-0h		R-0h	

Table 6-65. SECTSTAT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-26	STATUS_SECT13	R	0h	Reflects the status of flash CM BANK Sector 13. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
25-24	STATUS_SECT12	R	0h	Reflects the status of flash CM BANK Sector 12. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
23-22	STATUS_SECT11	R	0h	Reflects the status of flash CM BANK Sector 11. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
21-20	STATUS_SECT10	R	0h	Reflects the status of flash CM BANK Sector 10. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

Table 6-65. SECTSTAT2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	STATUS_SECT9	R	0h	Reflects the status of flash CM BANK Sector 9. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
17-16	STATUS_SECT8	R	0h	Reflects the status of flash CM BANK sector 8. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
15-14	STATUS_SECT7	R	0h	Reflects the status of flash CM BANK Sector 7. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
13-12	STATUS_SECT6	R	0h	Reflects the status of flash CM BANK Sector 6. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
11-10	STATUS_SECT5	R	0h	Reflects the status of flash CM BANK Sector 5. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
9-8	STATUS_SECT4	R	0h	Reflects the status of flash CM BANK Sector 4. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
7-6	STATUS_SECT3	R	0h	Reflects the status of flash CM BANK Sector 3. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
5-4	STATUS_SECT2	R	0h	Reflects the status of flash CM BANK Sector 2. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

Table 6-65. SECTSTAT2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	STATUS_SECT1	R	0h	Reflects the status of flash CM BANK sector 1. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
1-0	STATUS_SECT0	R	0h	Reflects the status of flash CM BANK Sector 0. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

6.9.5.4 SECTSTAT3 Register (Offset (x8) = 18h, Offset (x16) = Ch) [Reset = 0h]

SECTSTAT3 is shown in Figure 6-59 and described in Table 6-66.

Return to the Summary Table.

Flash Sectors Status Register 3

Figure 6-59. SECTSTAT3 Register

31	30	29	28	27	26	25	24
RESERVED				STATUS_SECT13		STATUS_SECT12	
R-0h				R-0h		R-0h	
23	22	21	20	19	18	17	16
STATUS_SECT11		STATUS_SECT10		STATUS_SECT9		STATUS_SECT8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
STATUS_SECT7		STATUS_SECT6		STATUS_SECT5		STATUS_SECT4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
STATUS_SECT3		STATUS_SECT2		STATUS_SECT1		STATUS_SECT0	
R-0h		R-0h		R-0h		R-0h	

Table 6-66. SECTSTAT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-26	STATUS_SECT13	R	0h	Reflects the status of flash CPU2 BANK Sector 13. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
25-24	STATUS_SECT12	R	0h	Reflects the status of flash CPU2 BANK Sector 12. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
23-22	STATUS_SECT11	R	0h	Reflects the status of flash CPU2 BANK Sector 11. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
21-20	STATUS_SECT10	R	0h	Reflects the status of flash CPU2 BANK Sector 10. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

Table 6-66. SECTSTAT3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	STATUS_SECT9	R	0h	Reflects the status of flash CPU2 BANK Sector 9. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
17-16	STATUS_SECT8	R	0h	Reflects the status of flash CPU2 BANK sector 8. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
15-14	STATUS_SECT7	R	0h	Reflects the status of flash CPU2 BANK Sector 7. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
13-12	STATUS_SECT6	R	0h	Reflects the status of flash CPU2 BANK Sector 6. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
11-10	STATUS_SECT5	R	0h	Reflects the status of flash CPU2 BANK Sector 5. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
9-8	STATUS_SECT4	R	0h	Reflects the status of flash CPU2 BANK Sector 4. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
7-6	STATUS_SECT3	R	0h	Reflects the status of flash CPU2 BANK Sector 3. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
5-4	STATUS_SECT2	R	0h	Reflects the status of flash CPU2 BANK Sector 2. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

Table 6-66. SECTSTAT3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	STATUS_SECT1	R	0h	Reflects the status of flash CPU2 BANK sector 1. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
1-0	STATUS_SECT0	R	0h	Reflects the status of flash CPU2 BANK Sector 0. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

6.9.5.5 RAMSTAT1 Register (Offset (x8) = 20h, Offset (x16) = 10h) [Reset = 0h]

RAMSTAT1 is shown in Figure 6-60 and described in Table 6-67.

Return to the Summary Table.

RAM Status Register 1

Figure 6-60. RAMSTAT1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				STATUS_RAM9		STATUS_RAM8	
R-0h				R-0h		R-0h	
15	14	13	12	11	10	9	8
STATUS_RAM7		STATUS_RAM6		STATUS_RAM5		STATUS_RAM4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
STATUS_RAM3		STATUS_RAM2		STATUS_RAM1		STATUS_RAM0	
R-0h		R-0h		R-0h		R-0h	

Table 6-67. RAMSTAT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-18	STATUS_RAM9	R	0h	Reflects the status of CPU1.D1 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
17-16	STATUS_RAM8	R	0h	Reflects the status of CPU1.D0 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
15-14	STATUS_RAM7	R	0h	Reflects the status of CPU1.LS7 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
13-12	STATUS_RAM6	R	0h	Reflects the status of CPU1.LS6 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

Table 6-67. RAMSTAT1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11-10	STATUS_RAM5	R	0h	Reflects the status of CPU1.LS5 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
9-8	STATUS_RAM4	R	0h	Reflects the status of CPU1.LS4 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
7-6	STATUS_RAM3	R	0h	Reflects the status of CPU1.LS3 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
5-4	STATUS_RAM2	R	0h	Reflects the status of CPU1.LS2 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
3-2	STATUS_RAM1	R	0h	Reflects the status of CPU1.LS1 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
1-0	STATUS_RAM0	R	0h	Reflects the status of CPU1.LS0 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

6.9.5.6 RAMSTAT2 Register (Offset (x8) = 24h, Offset (x16) = 12h) [Reset = 0h]

RAMSTAT2 is shown in Figure 6-61 and described in Table 6-68.

Return to the Summary Table.

RAM Status Register 2

Figure 6-61. RAMSTAT2 Register

31	30	29	28	27	26	25	24
STATUS_RAM15		STATUS_RAM14		STATUS_RAM13		STATUS_RAM12	
R-0h		R-0h		R-0h		R-0h	
23	22	21	20	19	18	17	16
STATUS_RAM11		STATUS_RAM10		STATUS_RAM9		STATUS_RAM8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
STATUS_RAM7		STATUS_RAM6		STATUS_RAM5		STATUS_RAM4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
RESERVED				STATUS_RAM1		STATUS_RAM0	
R-0h				R-0h		R-0h	

Table 6-68. RAMSTAT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	STATUS_RAM15	R	0h	Reflects the status of CM to CPU2 MSG RAM 1. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
29-28	STATUS_RAM14	R	0h	Reflects the status of MSG RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
27-26	STATUS_RAM13	R	0h	Reflects the status of MSG RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
25-24	STATUS_RAM12	R	0h	Reflects the status of MSG RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
23-22	STATUS_RAM11	R	0h	Reflects the status of MSG RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

Table 6-68. RAMSTAT2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	STATUS_RAM10	R	0h	Reflects the status of CM to CPU2 MSG RAM 0. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
19-18	STATUS_RAM9	R	0h	Reflects the status of CPU2 to CM MSG RAM 1. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
17-16	STATUS_RAM8	R	0h	Reflects the status of CPU2 to CM MSG RAM 0. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
15-14	STATUS_RAM7	R	0h	Reflects the status of CM to CPU1 MSG RAM 1. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
13-12	STATUS_RAM6	R	0h	Reflects the status of CM to CPU1 MSG RAM 0. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
11-10	STATUS_RAM5	R	0h	Reflects the status of CPU1 to CM MSG RAM 1. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
9-8	STATUS_RAM4	R	0h	Reflects the status of CPU1 to CM MSG RAM 0. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
7-4	RESERVED	R	0h	Reserved
3-2	STATUS_RAM1	R	0h	Reflects the status of C1 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

Table 6-68. RAMSTAT2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	STATUS_RAM0	R	0h	Reflects the status of C0 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

6.9.5.7 RAMSTAT3 Register (Offset (x8) = 28h, Offset (x16) = 14h) [Reset = 0h]

RAMSTAT3 is shown in Figure 6-62 and described in Table 6-69.

Return to the Summary Table.

RAM Status Register 3

Figure 6-62. RAMSTAT3 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				STATUS_RAM9		STATUS_RAM8	
R-0h				R-0h		R-0h	
15	14	13	12	11	10	9	8
STATUS_RAM7		STATUS_RAM6		STATUS_RAM5		STATUS_RAM4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
STATUS_RAM3		STATUS_RAM2		STATUS_RAM1		STATUS_RAM0	
R-0h		R-0h		R-0h		R-0h	

Table 6-69. RAMSTAT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-18	STATUS_RAM9	R	0h	Reflects the status of CPU2.D1 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
17-16	STATUS_RAM8	R	0h	Reflects the status of CPU2.D0 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
15-14	STATUS_RAM7	R	0h	Reflects the status of CPU2.LS7 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
13-12	STATUS_RAM6	R	0h	Reflects the status of CPU2.LS6 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

Table 6-69. RAMSTAT3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11-10	STATUS_RAM5	R	0h	Reflects the status of CPU2.LS5 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
9-8	STATUS_RAM4	R	0h	Reflects the status of CPU2.LS4 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
7-6	STATUS_RAM3	R	0h	Reflects the status of CPU2.LS3 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
5-4	STATUS_RAM2	R	0h	Reflects the status of CPU2.LS2 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
3-2	STATUS_RAM1	R	0h	Reflects the status of CPU2.LS1 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
1-0	STATUS_RAM0	R	0h	Reflects the status of CPU2.LS0 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

6.9.5.8 SECERRSTAT Register (Offset (x8) = 30h, Offset (x16) = 18h) [Reset = 0h]

SECERRSTAT is shown in Figure 6-63 and described in Table 6-70.

Return to the Summary Table.

Security Error Status Register

Figure 6-63. SECERRSTAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ERR
R-0h															R-0h

Table 6-70. SECERRSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	ERR	R	0h	This bit indicates if any error has occurred in the load of any security configuration from USER-OTP. 0: No error has occurred in the load of security information from USER-OTP 1: Error has occurred in the load of security information from USER-OTP Reset type: PORESETn

6.9.5.9 SECERRCLR Register (Offset (x8) = 34h, Offset (x16) = 1Ah) [Reset = 0h]

SECERRCLR is shown in Figure 6-64 and described in Table 6-71.

Return to the Summary Table.

Security Error Clear Register

Figure 6-64. SECERRCLR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ERR
R-0h															R-0/W1S-0h

Table 6-71. SECERRCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	ERR	R-0/W1S	0h	A write of '1' clears the SECERRSTST.ERR bit. Write of '0' is ignored. This bit always reads back '0'. Reset type: N/A

6.9.5.10 SECERRFRC Register (Offset (x8) = 38h, Offset (x16) = 1Ch) [Reset = 0h]

SECERRFRC is shown in Figure 6-65 and described in Table 6-72.

Return to the Summary Table.

Security Error Force Register

Figure 6-65. SECERRFRC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
R-0/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ERR
R-0h															R-0/W1S-0h

Table 6-72. SECERRFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	In order to write to the ERR bits, 0x5a5a must be written to these key bits at the same time. Otherwise, writes are ignored. The key is cleared immediately after writing, so it must be written again for every write to ERR. Reads will return 0. Reset type: N/A
15-1	RESERVED	R	0h	Reserved
0	ERR	R-0/W1S	0h	A write of '1', along with the proper KEY, sets the SECERRSTST.ERR bit. Write of '0' is ignored. This bit always reads back '0'. Reset type: N/A

6.9.6 DCSM_Z1_OTP Registers

Table 6-73 lists the memory-mapped registers for the DCSM_Z1_OTP registers. All register offset addresses not listed in Table 6-73 should be considered as reserved locations and the register contents should not be modified.

Table 6-73. DCSM_Z1_OTP Registers

Offset (x8)	Offset (x16)	Acronym	Register Name	Write Protection	Section
0h	0h	Z1OTP_LINKPOINTER1	Zone 1 Link Pointer1		Go
4h	2h	Z1OTP_LINKPOINTER2	Zone 1 Link Pointer2		Go
8h	4h	Z1OTP_LINKPOINTER3	Zone 1 Link Pointer3		Go
Ch	6h	Z1OTP_JLM_ENABLE	Zone 1 JTAGLOCK Enable Register		Go
10h	8h	Z1OTP_GPREG1	Zone 1 General Purpose Register 1		Go
14h	Ah	Z1OTP_GPREG2	Zone 1 General Purpose Register 2		Go
18h	Ch	Z1OTP_GPREG3	Zone 1 General Purpose Register 3		Go
1Ch	Eh	Z1OTP_GPREG4	Zone 1 General Purpose Register 4		Go
20h	10h	Z1OTP_PSWDLOCK	Secure Password Lock		Go
24h	12h	Z1OTP_CRCLOCK	Secure CRC Lock		Go
28h	14h	Z1OTP_JTAGPSWDH0	JTAG Lock Permanent Password 0		Go
2Ch	16h	Z1OTP_JTAGPSWDH1	JTAG Lock Permanent Password 1		Go
30h	18h	Z1OTP_CMACKEY0	Secure Boot CMAC Key 0		Go
34h	1Ah	Z1OTP_CMACKEY1	Secure Boot CMAC Key 1		Go
38h	1Ch	Z1OTP_CMACKEY2	Secure Boot CMAC Key 2		Go
3Ch	1Eh	Z1OTP_CMACKEY3	Secure Boot CMAC Key 3		Go

Complex bit access types are encoded to fit into small table cells. Table 6-74 shows the codes that are used for access types in this section.

Table 6-74. DCSM_Z1_OTP Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

6.9.6.1 Z1OTP_LINKPOINTER1 Register (Offset (x8) = 0h, Offset (x16) = 0h) [Reset = FFFFFFFFh]

Z1OTP_LINKPOINTER1 is shown in Figure 6-66 and described in Table 6-75.

Return to the Summary Table.

Zone 1 Link Pointer1

Figure 6-66. Z1OTP_LINKPOINTER1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_LINKPOINTER1																															
R-FFFFFFFFh																															

Table 6-75. Z1OTP_LINKPOINTER1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_LINKPOINTER1	R	FFFFFFFFh	Zone1 Link Pointer 1 location in USER OTP. Note: [1] ECC comparison is disabled for this location [2] When this value is loaded into DCSM, if the bits[31:14] !=0, device will remain in BLOCKED state. Before shipping parts to customers, TI would change the value of these bits to 0s. Reset type: N/A

6.9.6.2 Z1OTP_LINKPOINTER2 Register (Offset (x8) = 4h, Offset (x16) = 2h) [Reset = FFFFFFFFh]

Z1OTP_LINKPOINTER2 is shown in Figure 6-67 and described in Table 6-76.

Return to the Summary Table.

Zone 1 Link Pointer2

Figure 6-67. Z1OTP_LINKPOINTER2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_LINKPOINTER2																															
R-FFFFFFFFh																															

Table 6-76. Z1OTP_LINKPOINTER2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_LINKPOINTER2	R	FFFFFFFFh	Zone1 Link Pointer 2 location in USER OTP. Note: [1] ECC comparison is disabled for this location [2] When this value is loaded into DCSM, if the bits[31:14] !=0, device will remain in BLOCKED state. Before shipping parts to customers, TI would change the value of these bits to 0s. Reset type: N/A

6.9.6.3 Z1OTP_LINKPOINTER3 Register (Offset (x8) = 8h, Offset (x16) = 4h) [Reset = FFFFFFFFh]

Z1OTP_LINKPOINTER3 is shown in Figure 6-68 and described in Table 6-77.

Return to the Summary Table.

Zone 1 Link Pointer3

Figure 6-68. Z1OTP_LINKPOINTER3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_LINKPOINTER3																															
R-FFFFFFFFh																															

Table 6-77. Z1OTP_LINKPOINTER3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_LINKPOINTER3	R	FFFFFFFFh	Zone1 Link Pointer 3 location in USER OTP. Note: [1] ECC comparison is disabled for this location [2] When this value is loaded into DCSM, if the bits[31:14] !=0, device will remain in BLOCKED state. Before shipping parts to customers, TI would change the value of these bits to 0s. Reset type: N/A

6.9.6.4 Z1OTP_JLM_ENABLE Register (Offset (x8) = Ch, Offset (x16) = 6h) [Reset = FFFFFFFFh]

Z1OTP_JLM_ENABLE is shown in Figure 6-69 and described in Table 6-78.

Return to the Summary Table.

Zone 1 JTAGLOCK Enable Register

Figure 6-69. Z1OTP_JLM_ENABLE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_JLM_ENABLE																															
R-FFFFFFFFh																															

Table 6-78. Z1OTP_JLM_ENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_JLM_ENABLE	R	FFFFFFFFh	Zone1 JLM_ENABLE register location in USER OTP. Note: When this value is loaded into Z1_JLM_ENABLE, if the value is 32-bit all-1s, the JTAGLOCK will be enabled. Before shipping parts to customers, TI will program the default value to 0xFFFF_000F, which will disable the JTAGLOCK feature. Users should program 0xFFFF_0000 to enable the JTAGLOCK feature. Reset type: N/A

6.9.6.5 Z1OTP_GPREG1 Register (Offset (x8) = 10h, Offset (x16) = 8h) [Reset = FFFFFFFFh]

Z1OTP_GPREG1 is shown in Figure 6-70 and described in Table 6-79.

Return to the Summary Table.

Zone 1 General Purpose Register 1

Figure 6-70. Z1OTP_GPREG1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_GPREG1																															
R-FFFFFFFFh																															

Table 6-79. Z1OTP_GPREG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_GPREG1	R	FFFFFFFFh	Zone1 General Purpose register location in USER OTP. Reset type: N/A

6.9.6.6 Z1OTP_GPREG2 Register (Offset (x8) = 14h, Offset (x16) = Ah) [Reset = FFFFFFFFh]

Z1OTP_GPREG2 is shown in Figure 6-71 and described in Table 6-80.

Return to the Summary Table.

Zone 1 General Purpose Register 2

Figure 6-71. Z1OTP_GPREG2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_GPREG2																															
R-FFFFFFFFh																															

Table 6-80. Z1OTP_GPREG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_GPREG2	R	FFFFFFFFh	Zone1 General Purpose register location in USER OTP. Reset type: N/A

6.9.6.7 Z1OTP_GPREG3 Register (Offset (x8) = 18h, Offset (x16) = Ch) [Reset = FFFFFFFFh]

Z1OTP_GPREG3 is shown in Figure 6-72 and described in Table 6-81.

Return to the Summary Table.

Zone 1 General Purpose Register 3

Figure 6-72. Z1OTP_GPREG3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_GPREG3																															
R-FFFFFFFFh																															

Table 6-81. Z1OTP_GPREG3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_GPREG3	R	FFFFFFFFh	Zone1 General Purpose register location in USER OTP. Reset type: N/A

6.9.6.8 Z1OTP_GPREG4 Register (Offset (x8) = 1Ch, Offset (x16) = Eh) [Reset = FFFFFFFFh]

Z1OTP_GPREG4 is shown in Figure 6-73 and described in Table 6-82.

Return to the Summary Table.

Zone 1 General Purpose Register 4

Figure 6-73. Z1OTP_GPREG4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_GPREG4																															
R-FFFFFFFFh																															

Table 6-82. Z1OTP_GPREG4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_GPREG4	R	FFFFFFFFh	Zone1 General Purpose register location in USER OTP. Reset type: N/A

6.9.6.9 Z1OTP_PSWDLOCK Register (Offset (x8) = 20h, Offset (x16) = 10h) [Reset = FFFFFFFFh]

Z1OTP_PSWDLOCK is shown in Figure 6-74 and described in Table 6-83.

Return to the Summary Table.

Secure Password Lock

Figure 6-74. Z1OTP_PSWDLOCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_PSWDLOCK																															
R-FFFFFFFFh																															

Table 6-83. Z1OTP_PSWDLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_PSWDLOCK	R	FFFFFFFFh	Zone1 password lock location in USER OTP. Note: When this value is loaded into DCSM, if the value is 32-bit all-1s, CSMPSWD will remain locked. Before shipping parts to customers, TI would change the value of this location in such a way that the ECC field remains all-1s and also LSB 4-bits remain 4'b1111. Reset type: N/A

6.9.6.10 Z1OTP_CRCLOCK Register (Offset (x8) = 24h, Offset (x16) = 12h) [Reset = FFFFFFFFh]

Z1OTP_CRCLOCK is shown in Figure 6-75 and described in Table 6-84.

Return to the Summary Table.

Secure CRC Lock

Figure 6-75. Z1OTP_CRCLOCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_CRCLOCK																															
R-FFFFFFFFh																															

Table 6-84. Z1OTP_CRCLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_CRCLOCK	R	FFFFFFFFh	Zone1 CRC lock location in USER OTP. Note: When this value is loaded into DCSM, if the value is 32-bit all-1s, VCU will not have ability to calculate CRC on secured memory content.. Before shipping parts to customers, TI would change the value of this location in such a way that the ECC field remains all-1s and also LSB 4-bits remain 4'b1111. Reset type: N/A

6.9.6.11 Z1OTP_JTAGPSWDH0 Register (Offset (x8) = 28h, Offset (x16) = 14h) [Reset = FFFFFFFFh]

Z1OTP_JTAGPSWDH0 is shown in Figure 6-76 and described in Table 6-85.

Return to the Summary Table.

JTAG Lock Permanent Password 0

Figure 6-76. Z1OTP_JTAGPSWDH0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JTAGPSWDH0																															
R-FFFFFFFFh																															

Table 6-85. Z1OTP_JTAGPSWDH0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	JTAGPSWDH0	R	FFFFFFFFh	JTAG Lock Password High 0 (bits 95:64) location in USER Z1 OTP. This value is dummy loaded into the non-memory-mapped JTAGPSWD register, bits 95:64. TI must program a default value into this location, leaving the ECC bits all 1's. Reset type: N/A

6.9.6.12 Z1OTP_JTAGPSWDH1 Register (Offset (x8) = 2Ch, Offset (x16) = 16h) [Reset = FFFFFFFFh]

Z1OTP_JTAGPSWDH1 is shown in Figure 6-77 and described in Table 6-86.

Return to the Summary Table.

JTAG Lock Permanent Password 1

Figure 6-77. Z1OTP_JTAGPSWDH1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JTAGPSWDH1																															
R-FFFFFFFFh																															

Table 6-86. Z1OTP_JTAGPSWDH1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	JTAGPSWDH1	R	FFFFFFFFh	JTAG Lock Password High 1 (bits 127:96) location in USER Z1 OTP. This value is dummy loaded into the non-memory-mapped JTAGPSWD register, bits 127:96. Reset type: N/A

6.9.6.13 Z1OTP_CMACKEY0 Register (Offset (x8) = 30h, Offset (x16) = 18h) [Reset = FFFFFFFFh]

Z1OTP_CMACKEY0 is shown in Figure 6-78 and described in Table 6-87.

Return to the Summary Table.

Secure Boot CMAC Key 0

Figure 6-78. Z1OTP_CMACKEY0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMACKEY0																															
R-FFFFFFFFh																															

Table 6-87. Z1OTP_CMACKEY0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMACKEY0	R	FFFFFFFFh	Secure Boot CMAC Key 0 (bits 31:0) location in User Z1 OTP. This value is dummy loaded into the CMACKEY0 register. Reset type: N/A

6.9.6.14 Z1OTP_CMACKEY1 Register (Offset (x8) = 34h, Offset (x16) = 1Ah) [Reset = FFFFFFFFh]

Z1OTP_CMACKEY1 is shown in Figure 6-79 and described in Table 6-88.

Return to the Summary Table.

Secure Boot CMAC Key 1

Figure 6-79. Z1OTP_CMACKEY1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMACKEY1																															
R-FFFFFFFFh																															

Table 6-88. Z1OTP_CMACKEY1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMACKEY1	R	FFFFFFFFh	Secure Boot CMAC Key 1 (bits 63:32) location in User Z1 OTP. This value is dummy loaded into the CMACKEY1 register. Reset type: N/A

6.9.6.15 Z1OTP_CMACKEY2 Register (Offset (x8) = 38h, Offset (x16) = 1Ch) [Reset = FFFFFFFFh]

Z1OTP_CMACKEY2 is shown in Figure 6-80 and described in Table 6-89.

Return to the Summary Table.

Secure Boot CMAC Key 2

Figure 6-80. Z1OTP_CMACKEY2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMACKEY2																															
R-FFFFFFFFh																															

Table 6-89. Z1OTP_CMACKEY2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMACKEY2	R	FFFFFFFFh	Secure Boot CMAC Key 2 (bits 95:64) location in User Z1 OTP. This value is dummy loaded into the CMACKEY2 register. Reset type: N/A

6.9.6.16 Z1OTP_CMACKEY3 Register (Offset (x8) = 3Ch, Offset (x16) = 1Eh) [Reset = FFFFFFFFh]

Z1OTP_CMACKEY3 is shown in Figure 6-81 and described in Table 6-90.

Return to the Summary Table.

Secure Boot CMAC Key 3

Figure 6-81. Z1OTP_CMACKEY3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMACKEY3																															
R-FFFFFFFFh																															

Table 6-90. Z1OTP_CMACKEY3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMACKEY3	R	FFFFFFFFh	Secure Boot CMAC Key 3 (bits 127:96) location in User Z1 OTP. This value is dummy loaded into the CMACKEY3 register. Reset type: N/A

6.9.7 DCSM_Z2_OTP Registers

Table 6-91 lists the memory-mapped registers for the DCSM_Z2_OTP registers. All register offset addresses not listed in Table 6-91 should be considered as reserved locations and the register contents should not be modified.

Table 6-91. DCSM_Z2_OTP Registers

Offset (x8)	Offset (x16)	Acronym	Register Name	Write Protection	Section
0h	0h	Z2OTP_LINKPOINTER1	Zone 2 Link Pointer1		Go
4h	2h	Z2OTP_LINKPOINTER2	Zone 2 Link Pointer2		Go
8h	4h	Z2OTP_LINKPOINTER3	Zone 2 Link Pointer3		Go
10h	8h	Z2OTP_GPREG1	Zone 2 General Purpose Register 1		Go
14h	Ah	Z2OTP_GPREG2	Zone 2 General Purpose Register 2		Go
18h	Ch	Z2OTP_GPREG3	Zone 2 General Purpose Register 3		Go
1Ch	Eh	Z2OTP_GPREG4	Zone 2 General Purpose Register 4		Go
20h	10h	Z2OTP_PSWDLOCK	Secure Password Lock		Go
24h	12h	Z2OTP_CRCLOCK	Secure CRC Lock		Go

Complex bit access types are encoded to fit into small table cells. Table 6-92 shows the codes that are used for access types in this section.

Table 6-92. DCSM_Z2_OTP Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

6.9.7.1 Z2OTP_LINKPOINTER1 Register (Offset (x8) = 0h, Offset (x16) = 0h) [Reset = FFFFFFFFh]

Z2OTP_LINKPOINTER1 is shown in Figure 6-82 and described in Table 6-93.

Return to the Summary Table.

Zone 2 Link Pointer1

Figure 6-82. Z2OTP_LINKPOINTER1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2OTP_LINKPOINTER1																															
R-FFFFFFFFh																															

Table 6-93. Z2OTP_LINKPOINTER1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2OTP_LINKPOINTER1	R	FFFFFFFFh	Zone2 Link Pointer 1 location in USER OTP. Note: [1] ECC comparison is disabled for this location [2] When this value is loaded into DCSM, if the bits[31:14] !=0, device will remain in BLOCKED state. Before shipping parts to customers, TI would change the value of these bits to 0s. Reset type: N/A

6.9.7.2 Z2OTP_LINKPOINTER2 Register (Offset (x8) = 4h, Offset (x16) = 2h) [Reset = FFFFFFFFh]

Z2OTP_LINKPOINTER2 is shown in Figure 6-83 and described in Table 6-94.

Return to the Summary Table.

Zone 2 Link Pointer2

Figure 6-83. Z2OTP_LINKPOINTER2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2OTP_LINKPOINTER2																															
R-FFFFFFFFh																															

Table 6-94. Z2OTP_LINKPOINTER2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2OTP_LINKPOINTER2	R	FFFFFFFFh	Zone2 Link Pointer 2 location in USER OTP. Note: [1] ECC comparison is disabled for this location [2] When this value is loaded into DCSM, if the bits[31:14] !=0, device will remain in BLOCKED state. Before shipping parts to customers, TI would change the value of these bits to 0s. Reset type: N/A

6.9.7.3 Z2OTP_LINKPOINTER3 Register (Offset (x8) = 8h, Offset (x16) = 4h) [Reset = FFFFFFFFh]

Z2OTP_LINKPOINTER3 is shown in Figure 6-84 and described in Table 6-95.

Return to the Summary Table.

Zone 2 Link Pointer3

Figure 6-84. Z2OTP_LINKPOINTER3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2OTP_LINKPOINTER3																															
R-FFFFFFFFh																															

Table 6-95. Z2OTP_LINKPOINTER3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2OTP_LINKPOINTER3	R	FFFFFFFFh	Zone2 Link Pointer 3 location in USER OTP. Note: [1] ECC comparison is disabled for this location [2] When this value is loaded into DCSM, if the bits[31:14] !=0, device will remain in BLOCKED state. Before shipping parts to customers, TI would change the value of these bits to 0s. Reset type: N/A

6.9.7.4 Z2OTP_GPREG1 Register (Offset (x8) = 10h, Offset (x16) = 8h) [Reset = FFFFFFFFh]

Z2OTP_GPREG1 is shown in Figure 6-85 and described in Table 6-96.

Return to the Summary Table.

Zone 2 General Purpose Register 1

Figure 6-85. Z2OTP_GPREG1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2OTP_GPREG1																															
R-FFFFFFFFh																															

Table 6-96. Z2OTP_GPREG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2OTP_GPREG1	R	FFFFFFFFh	Zone2 General Purpose register location in USER OTP. Reset type: N/A

6.9.7.5 Z2OTP_GPREG2 Register (Offset (x8) = 14h, Offset (x16) = Ah) [Reset = FFFFFFFFh]

Z2OTP_GPREG2 is shown in Figure 6-86 and described in Table 6-97.

Return to the Summary Table.

Zone 2 General Purpose Register 2

Figure 6-86. Z2OTP_GPREG2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2OTP_GPREG2																															
R-FFFFFFFFh																															

Table 6-97. Z2OTP_GPREG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2OTP_GPREG2	R	FFFFFFFFh	Zone2 General Purpose register location in USER OTP. Reset type: N/A

6.9.7.6 Z2OTP_GPREG3 Register (Offset (x8) = 18h, Offset (x16) = Ch) [Reset = FFFFFFFFh]

Z2OTP_GPREG3 is shown in Figure 6-87 and described in Table 6-98.

Return to the Summary Table.

Zone 2 General Purpose Register 3

Figure 6-87. Z2OTP_GPREG3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2OTP_GPREG3																															
R-FFFFFFFFh																															

Table 6-98. Z2OTP_GPREG3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2OTP_GPREG3	R	FFFFFFFFh	Zone2 General Purpose register location in USER OTP. Reset type: N/A

6.9.7.7 Z2OTP_GPREG4 Register (Offset (x8) = 1Ch, Offset (x16) = Eh) [Reset = FFFFFFFFh]

Z2OTP_GPREG4 is shown in Figure 6-88 and described in Table 6-99.

Return to the Summary Table.

Zone 2 General Purpose Register 4

Figure 6-88. Z2OTP_GPREG4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2OTP_GPREG4																															
R-FFFFFFFFh																															

Table 6-99. Z2OTP_GPREG4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2OTP_GPREG4	R	FFFFFFFFh	Zone2 General Purpose register location in USER OTP. Reset type: N/A

6.9.7.8 Z2OTP_PSWDLOCK Register (Offset (x8) = 20h, Offset (x16) = 10h) [Reset = FFFFFFFFh]

Z2OTP_PSWDLOCK is shown in Figure 6-89 and described in Table 6-100.

Return to the Summary Table.

Secure Password Lock

Figure 6-89. Z2OTP_PSWDLOCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2OTP_PSWDLOCK																															
R-FFFFFFFFh																															

Table 6-100. Z2OTP_PSWDLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2OTP_PSWDLOCK	R	FFFFFFFFh	Zone2 password lock location in USER OTP. Note: When this value is loaded into DCSM, if the value is 32-bit all-1s, CSMPSWD will remain locked. Before shipping parts to customers, TI would change the value of this location in such a way that the ECC field remains all-1s and also LSB 4-bits remain 4'b1111. Reset type: N/A

6.9.7.9 Z2OTP_CRCLOCK Register (Offset (x8) = 24h, Offset (x16) = 12h) [Reset = FFFFFFFFh]

Z2OTP_CRCLOCK is shown in Figure 6-90 and described in Table 6-101.

Return to the Summary Table.

Secure CRC Lock

Figure 6-90. Z2OTP_CRCLOCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2OTP_CRCLOCK																															
R-FFFFFFFFh																															

Table 6-101. Z2OTP_CRCLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2OTP_CRCLOCK	R	FFFFFFFFh	Zone2 CRC lock location in USER OTP. Note: When this value is loaded into DCSM, if the value is 32-bit all-1s, VCU will not have ability to calculate CRC on secured memory content.. Before shipping parts to customers, TI would change the value of this location in such a way that the ECC field remains all-1s and also LSB 4-bits remain 4'b1111. Reset type: N/A

This page intentionally left blank.

Chapter 7

Background CRC-32 (BGCRC)

The Background CRC (BGCRC) module that helps to identify memory faults and corruption, is discussed in this chapter.

7.1 Introduction

The Background CRC (BGCRC) module computes a CRC-32 on a configurable block of memory. It accomplishes this by fetching the specified block of memory during idle cycles (when the CPU, CLA, or DMA is not accessing the memory block). The calculated CRC-32 value is compared against a golden CRC-32 value to indicate a pass or fail. In essence, the BGCRC helps identify memory faults and corruption. There are two BGCRC modules (CPU_CRC and CLA_CRC) per CPU subsystem. The two BGCRC modules differ only in the memories they test.

7.1.1 BGCRC Related Collateral

Getting Started Materials

- *CRC Engines in C2000 Devices Application Report*

7.1.2 Features

The BGCRC module has the following features:

- One cycle CRC-32 computation on 32 bits of data
- No CPU bandwidth impact for zero wait state memory
- Minimal CPU bandwidth impact for non-zero wait state memory
- Dual operation modes (CRC-32 mode and scrub mode)
- Watchdog timer to time CRC-32 completion
- Ability to pause and resume CRC-32 computation

7.1.3 Block Diagram

Figure 7-1 shows the BGCRC block diagram.

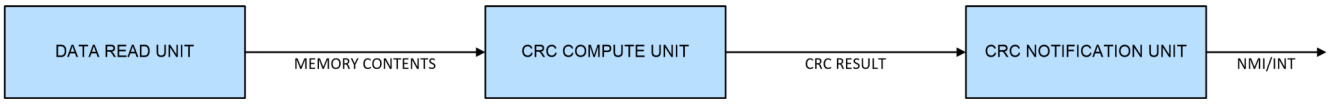


Figure 7-1. BGCRC Block Diagram

7.1.4 Memory Wait States and Memory Map

Figure 7-2 shows the memory map of the BGCRC module. M0, M1, D[x]RAM, MSGRAM, LS[x]RAM, and GS[x]RAM are all zero wait-state memories. BGCRC accesses these memories with minimal impact on normal program operation. For instance, if a BGCRC access is being made to a zero wait-state memory in the current cycle, the earliest the operating program can make access to the same memory location is in the next cycle. Similarly for the non-zero wait state memories SECROM, DATAROM and BOOTROM, the worst case delay for functional access after a BGCRC access is the wait-state amount.

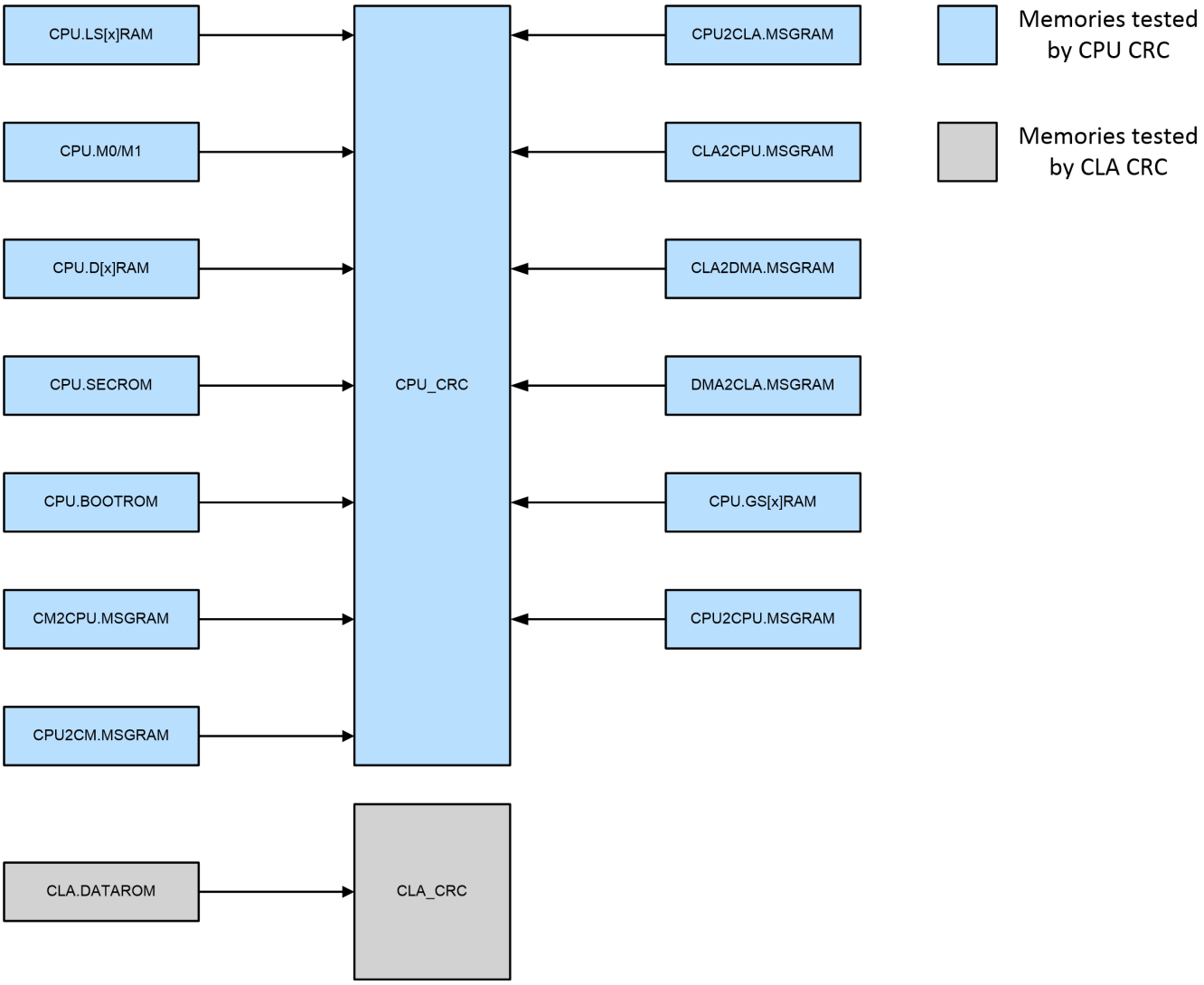


Figure 7-2. BGCRC Memory Map

7.2 Functional Description

The CRC-32 calculation of the BGCRC module can be kicked off by configuring the register BGCRC_EN.START to 1010. Once started, the module performs the background memory checks without CPU or CLA intervention. However, on completion of the CRC-32 calculation or in the event of a failure, the CPU or CLA is notified through an interrupt and task, respectively.

As highlighted in the overview, there are two BGCRC modules per CPU subsystem. CPU_CRC that is configurable by the CPU can only generate an interrupt. CLA_CRC however, can be configured by both the CPU and CLA and can generate an interrupt to the CPU or task for the CLA.

7.2.1 Data Read Unit

Once the CRC-32 calculation is started, the BGCRC module continuously reads data from memory as a background process. These reads happen during the idle times (when the CPU, CLA or DMA is not accessing the memory block) and so does not impact functional access. The data read unit only reads data if there is no pending functional access. The data read unit begins operation by reading a block of data BGCRC_CTRL2.BLOCK_SIZE from address BGCRC_START_ADDR. Note that BGCRC_START_ADDR must be 0x80 word aligned. For a non-0x80 word aligned BGCRC_START_ADDR, the LSB bits are zeroed out to get a 0x80 word aligned BGCRC_START_ADDR. For instance, if the programmed BGCRC_START_ADDR = 0x1AF3, the internal 0x80 word aligned start address is 0x1A80.

When the data read unit reads a block of data, ECC and parity are checked. Any ECC or parity errors that occur during the read is indicated by setting the respective NMI and generating an interrupt if configured as so. The BGCRC module, however, does not write back the corrected memory contents on the occurrence of a correctable ECC error. Writing back the corrected values should be handled by software.

7.2.2 CRC-32 Compute Unit

After the data read unit reads a block of data, the data read unit feeds this data to the CRC-32 compute unit. This unit computes the CRC-32 using the standard polynomial 0x04C11DB7 (x32 + x26 + x23 + x22 + x16 + x12 + x11 + x10 + x8 + x7 + x5 + x4 + x2 + x + 1). The CRC-32 unit retrieves 32-bit of data at a time from the block of data to compute the polynomial. This computation takes 1 cycle. For instance, the CRC-32 calculation for 1KB block of data is 256 cycles, 1 cycle for each of the 256 32-bit chunks. The initial value for the CRC-32 computation can be configured using BGCRC_SEED.

After the CRC-32 calculation is complete for a data block, the final result is loaded into BGCRC_RESULT. Note that BGCRC_RESULT only contains the final calculation for the whole data block; intermediate 32-bit calculations do not update BGCRC_RESULT. The value in BGCRC_RESULT is compared against the value in BGCRC_GOLDEN by hardware and the NMI/Interrupt flags are set accordingly by the CRC notification unit.

Once CRC-32 calculation is commenced, the calculation can be halted by setting BGCRC_CTRL2.TEST_HALT to 1010. Clearing this bit resumes CRC-32 calculation from the halt point.

7.2.3 CRC Notification Unit

After the CRC-32 compute unit completes the CRC-32 calculation for a block of data or fails to complete the calculation within BGCRC_WD_MIN and BGCRC_WD_MAX, if configured, the CRC-32 compute unit sends out a NMI/Interrupt on the occurrence of a pass or fail. In addition, during a data read by the data read unit, if an ECC or parity error occurs, the CRC notification unit can send out a NMI/Interrupt. In the case of an ECC or parity error, the BGCRC stops operation and BGCRC_CURR_ADDR contains the memory address that caused the ECC or parity error. The contents of BGCRC_CURR_ADDR in addition to the NMI/Interrupt flags can be used to debug a BGCRC failure.

7.2.3.1 CPU Interrupt, CLA Task and NMI

The BGCRC module has configurable interrupt and NMI lines. NMIs are enabled by default but can be disabled by writing 0xA to BGCRC_CTRL1.NMIDIS register. Conversely, all interrupts are disabled by default but can be enabled by writing 0x7E to BGCRC_INTEN register. When an error occurs in the BGCRC, the BGCRC can be configured to generate an NMI or interrupt. Since NMIs are enabled by default, all BGCRC errors cause an NMI. Figure 7-3 and Figure 7-4 show the NMI and Interrupt lines, respectively.

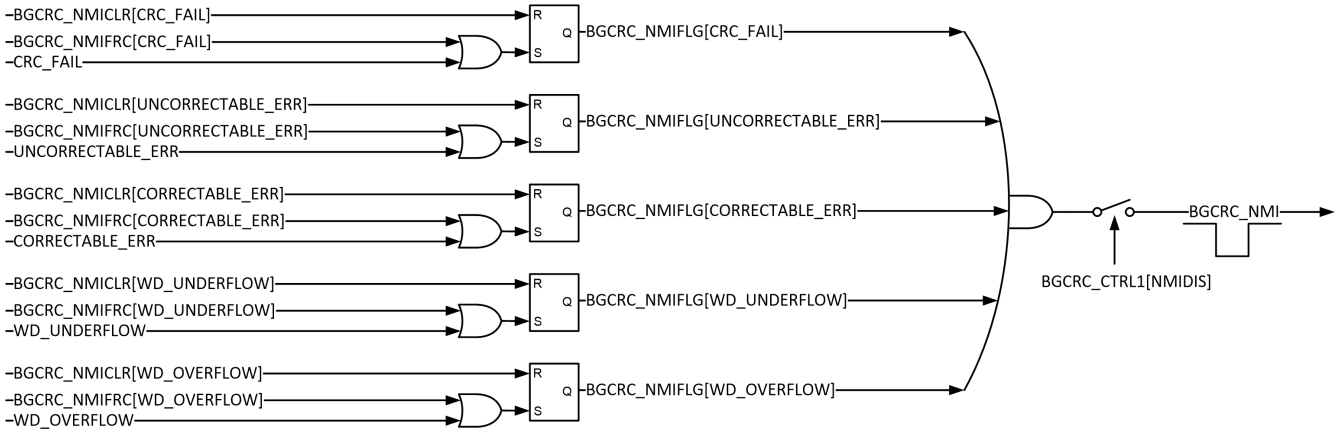


Figure 7-3. BGCRC NMI

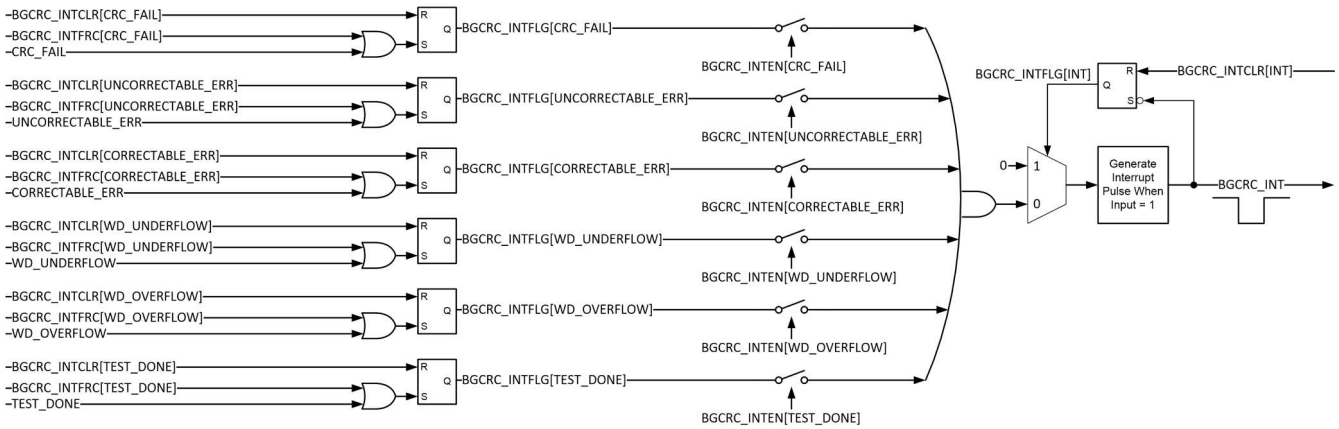


Figure 7-4. BGCRC Interrupt

7.2.4 Operating Modes

BGCRC module supports two modes of operation: CRC mode and scrub mode. The mode of operation can be configured by clearing or setting the register BGCRC_CTRL2.SCRUB_MODE.

7.2.4.1 CRC Mode

In CRC mode, the BGCRC module operates as explained in Section 7.2.2. CRC-32 calculation is performed on a block of data and the result compared against a golden value. ECC and parity errors are checked in this mode. Result mismatch, ECC or parity error can trigger an NMI/Interrupt.

7.2.4.2 Scrub Mode

In scrub mode, the CRC-32 result is not compared against the golden value and BGCRC_RESULT register is not updated. ECC and parity errors are also checked in this mode. However unlike CRC mode, NMI/Interrupt are only from ECC or parity error. In scrub mode, Parity and ECC bits of the memory block need to be initialized by the CPU and/or CLA.

7.2.5 BGCRC Watchdog

The BGCRC module has an embedded windowed watchdog that is used as a diagnostic to check memory test completion within the expected time window. This can protect against hardware defects that can cause the memory check not to complete in the allotted time, which may not be caught by the system watchdog as the error can be due to the CLA or DMA having continuous access. Windowing also helps detect additional failure modes in the watchdog operation, for example, stuck watchdog.

The BGCRC watchdog is enabled by default and starts when the BGCRC module begins reading from memory. The watchdog can be disabled using the register BGCRC_WD_CFG.WDDIS. The BGCRC watchdog counter is a 32-bit counter with the value reflected in BGCRC_WD_CNT register. The lower and upper window settings are configured using BGCRC_WD_MIN and BGCRC_WD_MAX, respectively. BGCRC_WD_MIN and BGCRC_WD_MAX need to be configured before the test is started and should not be changed while the BGCRC is operating. If configured, an NMI or interrupt is triggered if the memory test fails to complete within the configured time window. The counter stops on completion of the CRC-32 check done, CRC-32 check failure and ECC/Parity errors. The counter is reset when the next memory check begins.

The BGCRC watchdog can be halted by configuring the BGCRC_WD_CFG.WDDIS register. After the watchdog resumes from being halted, the counter starts counting from the previous count unless a new memory check operation is initiated. The counter is not halted when CRC-32 computation halts but by default halts during a debug halt. The behavior of the watchdog during emulation can be changed by configuring the appropriate BGCRC registers. In addition, due to the changing nature of memory contents during emulation, it is not recommended to run BGCRC during emulation. CRC-32 computation continues during a watchdog failure and software needs to address this condition.

7.2.6 Hardware and Software Faults Protection

The configuration registers are protected using a lock and commit configuration. Each of the configuration registers can be individually locked and committed. The register once locked, can no longer be updated until the lock is removed. However if the register lock is committed, no further writes is permitted until the device is reset. It is recommended to lock the registers after configuration to protect against corruption due to software faults. In addition, registers critical to the module functionality and fault detection are implemented using multi-bit fields to protect against hardware faults.

7.3 Application of the BGCRC

This section contains use case scenarios of how the BGCRC module can be configured to test a block of memory. However, the use case scenarios presented are for reference only and do not cover all the possible application scenarios. These use case scenarios could be used as a starting point to configure the BGCRC module for specific applications.

7.3.1 Software Configuration

The configuration registers for the BGCRC can be split into three groups (see Table 7-1):

- **CFG1:** Registers that determine the operating mode and configured at the beginning.
- **CFG2:** Registers that need to be updated during kickoff of a new test.
- **CFG3:** Registers used for test and error management.

CFG1 registers are expected to be locked and committed after initial configuration. It is recommended to lock the CFG2 and CFG3 registers after configuration. Figure 7-5 shows the BGCRC execution sequence.

Table 7-1. BGCRC Register Groups

CFG1 - One Time Configuration Registers	CFG2 - Periodic Configuration Registers	CFG3 - Registers Used for Test and Error Management
BGCRC_CTRL1	BGCRC_EN	BGCRC_NMICLR
BGCRC_WD_CFG	BGCRC_CTRL2	BGCRC_INTCLR
BGCRC_INTEN	BGCRC_START_ADDR	BGCRC_NMIFRC
BGCRC_SEED	BGCRC_GOLDEN	BGCRC_INTFRC
	BGCRC_WD_MIN	
	BGCRC_WD_MAX	

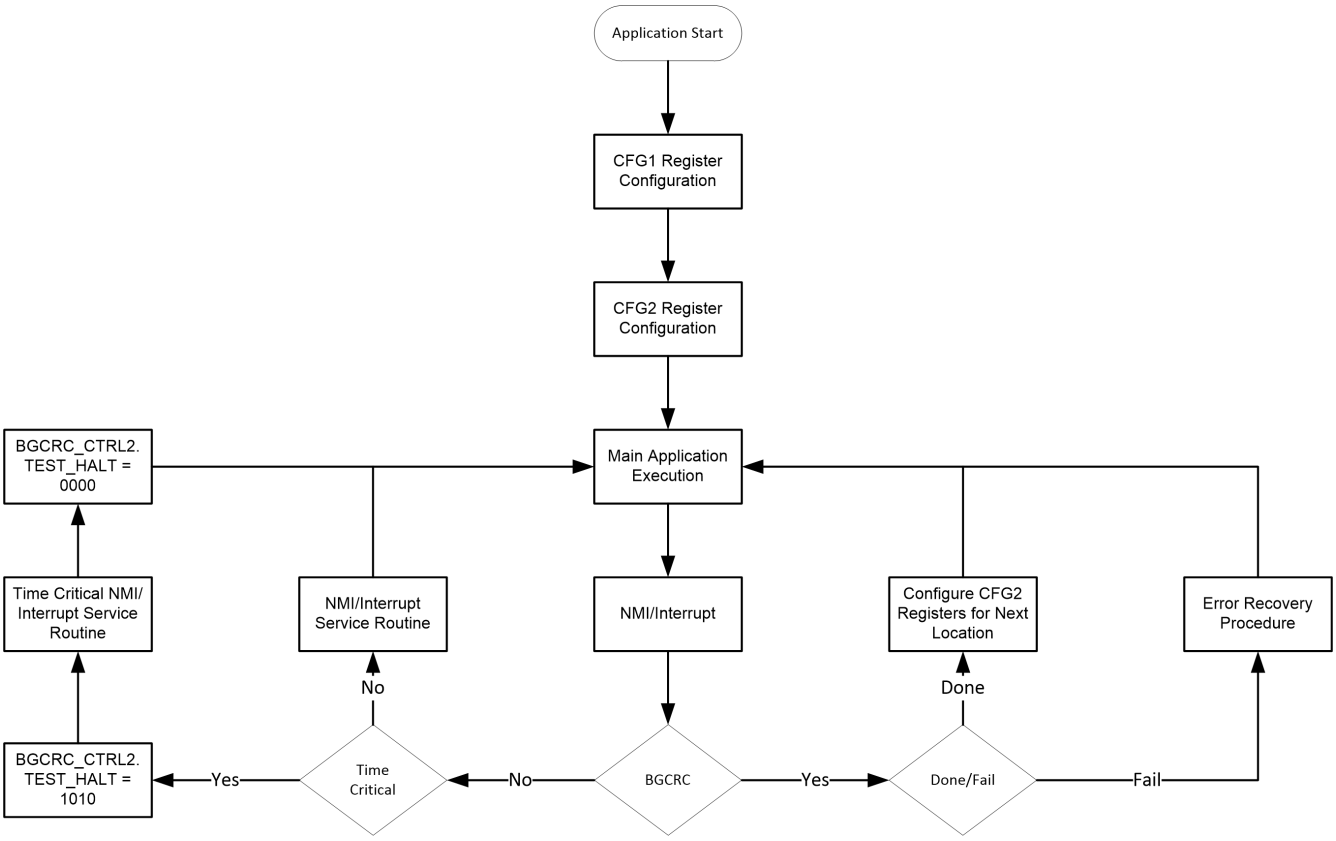


Figure 7-5. BGCRC Execution Sequence Flow

7.3.2 Decision on Error Response Severity

The error sources for BGCRC are:

- Test completion before BGCRC_WD_MIN.
- Test completed after BGCRC_WD_MAX.
- Mismatch between the calculated CRC and golden CRC.
- Uncorrectable error during data read (single bit error for parity memory or double bit error for ECC memory).
- Correctable error during memory read (single bit error for ECC memory).

Error severity can be chosen as either NMI or interrupt. By default, error severity is set to NMI. It is not possible to configure error response for each individual error source. The response can be chosen as either interrupt or NMI for all error sources. When error severity is chosen as NMI, ERROR_STS pin is asserted during an error. CPU and CLA should be assigned the same error severity.

7.3.3 Decision of Controller for CLA_CRC

CPU_CRC can only be kicked off by the CPU. However, CLA_CRC can be kicked off by the CPU or CLA. If BGCRC error severity is chosen as NMI, it is possible to handle the background test using the CLA and error response with the CPU. Once the controller for BGCRC is chosen, it is possible to prevent accesses from other controllers by using the system-level access protection configuration.

7.3.4 Execution of Time Critical Code from Wait-Stated Memories

BGCRC access to functionally wait-stated memories is also wait-stated by the same number. Since it is impossible to predict the next functional access, any ongoing BGCRC access has to complete before functional access is granted. To mitigate delay in functional access, the BGCRC can be halted when time-critical code that accesses the wait-stated memories is in progress. When BGCRC execution is halted, the BGCRC watchdog is not halted. This is consistent with the safety requirement to complete the background test in a predictable window irrespective of the user code. In such scenarios, it is recommended to adjust the upper BGCRC watchdog window limit to account for the halt duration during functional access. However, if required, the BGCRC watchdog can be disabled.

7.3.5 BGCRC Execution

Two notes about BGCRC execution are:

- BGCRC task can run as a background task once kicked off.
- BGCRC task does not impact functional execution for zero-wait stated memories. For memories with higher wait-states, the BGCRC engine can be halted to make functional execution predictable.

Figure 7-6 shows a few examples of BGCRC execution sequences.

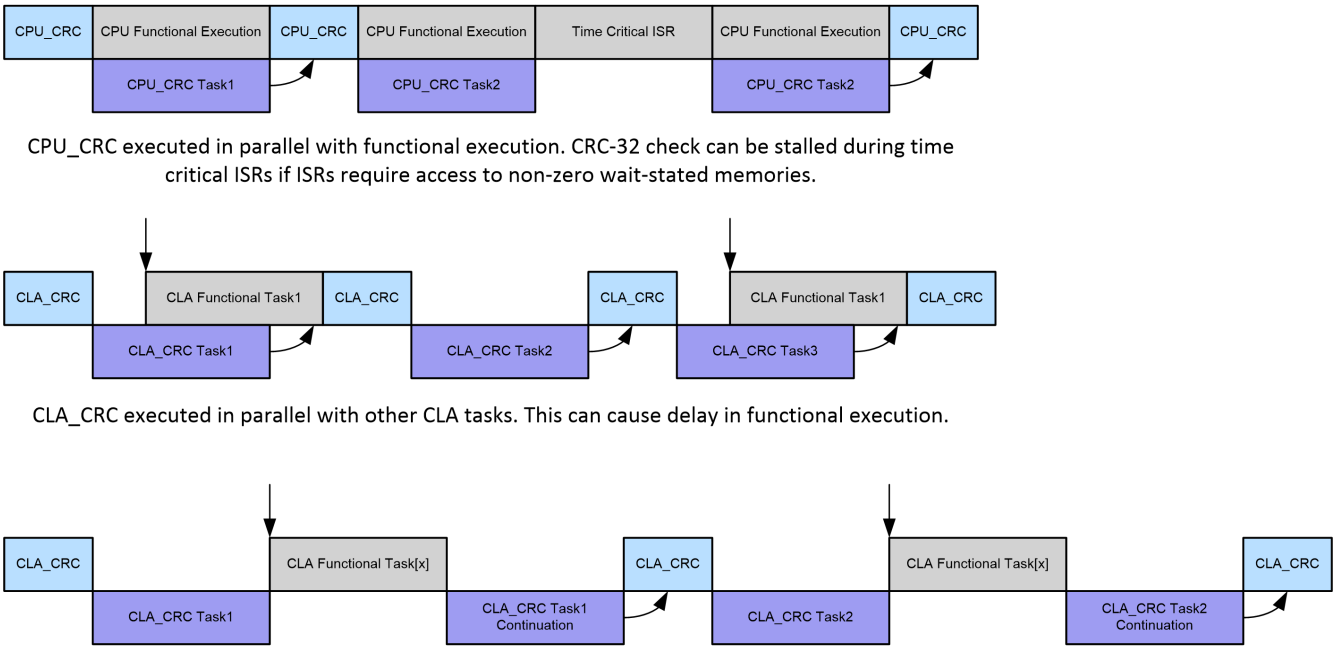


Figure 7-6. BGCRC Execution Sequence Example

7.3.6 Debug/Error Response for BGCRC Errors

The BGCRC error severity can be decoded by reading the BGCRC_INTFLG or BGCRC_NMIFLG. The errors can be:

- **WD_OVERFLOW/WD_UNDERFLOW:** The test did not complete in the programmed window. System timing needs to be checked to understand reason for non-completion of the test. BGCRC_CURR_ADDR and BGCRC_WD_CNT indicate how far the test progressed.
- **UNCORRECTABLE_ERR:** BGCRC_CURR_ADDR contains the address of the memory location causing the error. This error is due to single bit failure for Parity SRAM or double bit failure for ECC SRAM. The memory location can be reloaded (if possible, for cases where code is copied from Flash) to see if the problem resolves. If the problem persists, the problem can be a permanent defect.
- **CORRECTABLE_ERR:** This error is due to single bit failure for ECC SRAM. If the problem location is accessed again (execution from the location for execute only memory or reading the location in other cases), the expectation is that the single bit error is corrected. If the single bit error is not corrected, this can be an indication of a permanent defect.
- **CRC_FAIL:** This indicates a failure in the computation of the CRC-32 value. This error does not occur in scrub mode. For SRAMs with protection, in the absence of code bugs, this error is less likely since the error is most often manifest as a correctable/uncorrectable error. Code bugs can cause failure if the code inadvertently writes to a wrong address thus causing a CRC-32 error.

7.3.7 BGCRC Golden CRC-32 Value Computation

The C28x is a little-endian, 16-bit word addressable CPU. Therefore, the 32-bit value of 0x12345678 stored at address 0x100 is stored in the C28x memory as shown in Table 7-2.

Table 7-2. Data Address Location Example 1

Address	0x100	0x101
Data	0x5678	0x1234

The BGCRC order of byte calculations of the above example is 0x78, 0x56, 0x34, 0x12 and yields 0x6A330D2D. The 32-bit polynomial 0x04C11DB7 is used with an initialization vector of 0x00000000. The following code snippet shows the effective bit processing. Processing for all 32-bits within a word occurs in a single cycle within the BGCRC hardware.

```
seed = 0x0UL; //Initialize With Seed
poly = 0x04C11DB7UL: //32-Bit Polynomial
crc32 = seed;

for(i=0; i<dataSize; i++)
{
         byteSwappedData = ((data[i] & 0x000000FF) << 24)|
                         = ((data[i] & 0x0000FF00) << 8) |
                         = ((data[i] & 0x00FF0000) >> 8) |
                         = ((data[i] & 0xFF000000) >> 24);

        crc32 = byteSwappedData^crc32;
        for(j=0; j<32; j++)
        {
            if(crc32 & 0x80000000) crc32 = (crc32 << 1)^poly;
            else crc32 = crc32 << 1;

            crc32 = crc32 & 0xFFFFFFFF;
        }
}
```

A second example (Table 7-3) with two 32-bit words, 0x12345678 and 0x9ABCDEF0 at address 0x100 and 0x102 successively, would calculate the bytes in the order 0x78, 0x56, 0x34, 0x12, 0xDE, 0xBC, and 0x9A and yield 0x7E0B4164.

Table 7-3. Data Address Location Example 2

Address	0x100	0x101	0x102	0x103
Data	0x5678	0x1234	0xDEF0	0x9ABC

All data input to the BGCRC must align to a 32-bit boundary, both in the starting address and the size. It is possible to include 16-bit data within the span of data; however, when the data is read by the BGCRC, it always assume 32-bits and conform to the above calculation order. For example, if two 16-bit words (0xA0B1 and 0xC2D3) were placed in between the previous two 32-bit words (Table 7-4), the calculations would be performed in byte order 0x78, 0x56, 0x34, 0x12, 0xB1, 0xA0, 0xD3, 0xC2, 0xF0, 0xDE, 0xBC, and 0x9A and yield 0x2AEFD987.

Table 7-4. Data Address Location Example 3

Address	0x100	0x101	0x102	0x103	0x104	0x105
Data	0x5678	0x1234	0xA0B1	0xC2D3	0xDEF0	0x9ABC

7.4 Software

7.4.1 BGCRC Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/bgcrc

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

7.4.1.1 BGCRC CPU Interrupt Example

FILE: bgcrc_ex1_cpuinterrupt.c

This example demonstrates how to configure and trigger BGCRC from the CPU. BGCRC module is configured for 1 KB of GS0 RAM which is programmed with a known data. The pre-computed CRC value is used as the golden CRC value. Interrupt is generated once the computation is done and checks if no error flags are raised Calculation uses the 32-bit polynomial 0x04C11DB7 and seed value 0x00000000.

External Connections

- None.

Watch Variables

- pass - This should be 1.
- runStatus - BGCRC running status. This will be BGCRC_ACTIVE if the module is running, BGCRC_IDLE if the module is idle

7.4.1.2 BGCRC Example with Watchdog and Lock

FILE: bgcrc_ex2_cpubgcrc_basic.c

This example demonstrates how to configure and trigger BGCRC from the CPU. It also showcases how to configure the CRC watchdog and lock the registers after configuring the module. The watchdog is used as a diagnostic to check memory test completion within the expected time window. An error signal is generated if the test does not complete in the specified time window.

The module is configured for 1kB of GS0 RAM which is programmed with random data. The golden CRC value for comparison is computed using software method. Interrupt is generated once the computation is done and checks if no error flags are raised. The NMI is enabled and is triggered if an error is detected.

External Connections

- None.

Watch Variables

- pass
- bgcrcDone

7.4.1.3 CLA-BGCRC Example in CRC mode

FILE: bgcrc_ex3_clabgcrc_crcmode.c

This example demonstrates how to configure and trigger CLABGCRC from the CPU. It also showcases how to configure the CRC watchdog and lock the registers after configuring the module. The watchdog is used as a diagnostic to check memory test completion within the expected time window. An error signal is generated if the test does not complete in the specified time window.

The module is configured for 1kB of CLA ROM memory. The golden CRC value for comparison is computed using software method. Interrupt is generated once the computation is done and checks if no error flags are raised. The NMI is enabled and is triggered if an error is detected.

External Connections

- None.

Watch Variables

- pass
- bgcrcDone

7.4.1.4 CLA-BGCRC Example in Scrub Mode

FILE: bgcrc_ex4_clabgcrc_scrubmode.c

This example demonstrates how to configure and trigger CLA-BGCRC in Scrub mode. In Scrub mode, CRC of data is not compared with the golden CRC. Error check is done using the ECC/Parity logic. It also showcases how to configure the CRC watchdog and lock the registers after configuring the module. The watchdog is used as a diagnostic to check memory test completion within the expected time window. An error signal is generated if the test does not complete in the specified time window.

The module is configured for 256 bytes of CLA ROM memory. Interrupt is generated once the computation is done and checks if no error flags are raised. The NMI is enabled and is triggered if an error is detected.

External Connections

- None.

Watch Variables

- pass
- bgcrcDone

7.5 BGCRC Registers

This section describes the Background CRC registers.

7.5.1 BGCRC Base Address Table (C28)

Table 7-5. BGCRC Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
BgcrcCla1Regs	BGCRC_REGS	BGCRC_CLA1_BASE	0x0000_6380	YES	YES	-	YES	YES
BgcrcCpuRegs	BGCRC_REGS	BGCRC_CPU_BASE	0x0000_6340	YES	YES	-	-	YES

7.5.2 BGCRC_REGS Registers

Table 7-6 lists the memory-mapped registers for the BGCRC_REGS registers. All register offset addresses not listed in Table 7-6 should be considered as reserved locations and the register contents should not be modified.

Table 7-6. BGCRC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	BGCRC_EN	BGCRC Enable	EALLOW	Go
2h	BGCRC_CTRL1	BGCRC Control register 1	EALLOW	Go
4h	BGCRC_CTRL2	BGCRC Control register 2	EALLOW	Go
6h	BGCRC_START_ADDR	Start address for the BGCRC check	EALLOW	Go
8h	BGCRC_SEED	Seed for CRC calculation	EALLOW	Go
Eh	BGCRC_GOLDEN	Golden CRC to be compared against	EALLOW	Go
10h	BGCRC_RESULT	CRC calculated		Go
12h	BGCRC_CURR_ADDR	Current address regsiter		Go
1Ch	BGCRC_WD_CFG	BGCRC windowed watchdog configuration	EALLOW	Go
1Eh	BGCRC_WD_MIN	BGCRC windowed watchdog min value	EALLOW	Go
20h	BGCRC_WD_MAX	BGCRC windowed watchdog max value	EALLOW	Go
22h	BGCRC_WD_CNT	BGCRC windowed watchdog count		Go
2Ah	BGCRC_NMIFLG	BGCRC NMI flag register		Go
2Ch	BGCRC_NMICLR	BGCRC NMI flag clear register	EALLOW	Go
2Eh	BGCRC_NMIFRC	BGCRC NMI flag force register	EALLOW	Go
34h	BGCRC_INTEN	Interrupt enable	EALLOW	Go
36h	BGCRC_INTFLG	Interrupt flag		Go
38h	BGCRC_INTCLR	Interrupt flag clear	EALLOW	Go
3Ah	BGCRC_INTFRC	Interrupt flag force	EALLOW	Go
3Ch	BGCRC_LOCK	BGCRC register map lockconfiguration	EALLOW	Go
3Eh	BGCRC_COMMIT	BGCRC register map commit configuration	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 7-7 shows the codes that are used for access types in this section.

Table 7-7. BGCRC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 7-7. BGCRC_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

7.5.2.1 BGCRC_EN Register (Offset = 0h) [Reset = 0h]

BGCRC_EN is shown in Figure 7-7 and described in Table 7-8.

Return to the Summary Table.

BGCRC Enable

Figure 7-7. BGCRC_EN Register

31	30	29	28	27	26	25	24
RUN_STS	RESERVED						
R-0h	R-0-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				START			
R-0-0h				R-0/W-0h			

Table 7-8. BGCRC_EN Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RUN_STS	R	0h	Status bit: 0 : CRC module is IDLE 1 : CRC module is Active This bit will remain set during BGCRC_CTRL2.TEST_HALT = 1 Reset type: CPUx.SYSRSn
30-16	RESERVED	R-0	0h	Reserved
15-4	RESERVED	R-0	0h	Reserved
3-0	START	R-0/W	0h	Start Bit: "1010": Kick-off CRC calculations "any other value": ignored Notes: BGCRC_WD_CNT regsters will be reset CRCEN.START = "1010". BGCRC_INTFLG, BGCRC_NMIFLG will not be impacted by this configuration. This bit should not be set before the previous CRC computation completes. Reset type: CPUx.SYSRSn

7.5.2.2 BGCRC_CTRL1 Register (Offset = 2h) [Reset = 0h]

BGCRC_CTRL1 is shown in Figure 7-8 and described in Table 7-9.

Return to the Summary Table.

BGCRC Control register 1

Figure 7-8. BGCRC_CTRL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED				NMIDIS			
R-0-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED			FREE_SOFT	RESERVED			
R-0-0h			R/W-0h	R-0-0h			

Table 7-9. BGCRC_CTRL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R-0	0h	Reserved
19-16	NMIDIS	R/W	0h	1010 : NMI is disabled Any other value : NMI is enabled. Reset type: CPUx.SYSRSn
15-5	RESERVED	R-0	0h	Reserved
4	FREE_SOFT	R/W	0h	Emulation control bit : This bit controls behaviour of CRC calculation during emulation 0 : Soft, CRC module and CRC Watchdog stops immediately on DEBUG SUSPEND (of CRC-controller). 1 : Free, CRC calcuation and CRC watchdog is not affected by DEBUG HALT (of CRC-controller) Reset type: CPUx.SYSRSn
3-0	RESERVED	R-0	0h	Reserved

7.5.2.3 BGCRC_CTRL2 Register (Offset = 4h) [Reset = 0h]

BGCRC_CTRL2 is shown in Figure 7-9 and described in Table 7-10.

Return to the Summary Table.

BGCRC Control register 2

Figure 7-9. BGCRC_CTRL2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED				SCRUB_MODE			
R-0-0h				R/W-0h			
15	14	13	12	11	10	9	8
TEST_HALT				RESERVED		BLOCK_SIZE	
R/W-0h				R-0-0h		R/W-0h	
7	6	5	4	3	2	1	0
BLOCK_SIZE							
R/W-0h							

Table 7-10. BGCRC_CTRL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R-0	0h	Reserved
19-16	SCRUB_MODE	R/W	0h	Scrub mode configuration 1010 : Scrub mode, CRC of data is not compared with the golden CRC. Error check is done using the ECC/Parity logic. Any other value: CRC value is compared with golden CRC at the end in addition to the data correctness check by ECC/Parity logic. Notes: 1010 configuration is used for scrub mode (for data memories) where the memory value is read and ECC/Parity logic is used for the error detection. BGCRC_RESULT.CRC_VALUE is not updated in this configuration. Reset type: CPUx.SYSRSn
15-12	TEST_HALT	R/W	0h	Halt Bit : 1010 : Module operation is stopped Any other value : CRC calaculation will continue/resume from where it was halted Notes: BGCRC_EN.START = 1010 configuration with TEST_HALT = 1010 will halt the CRC calculation. The new check will resume when TEST_HALT is configured to a value other than 1010 Reset type: CPUx.SYSRSn
11-10	RESERVED	R-0	0h	Reserved
9-0	BLOCK_SIZE	R/W	0h	Configures the block size for the check 0x0 : 256 Byte (default) 0x1 : 512 Byte 0x2 : 768 Byte 0x3 : 1KB ... 0x3FF : 256KB (0xn : (n+1)*256Byte) Reset type: CPUx.SYSRSn

7.5.2.4 BGCRC_START_ADDR Register (Offset = 6h) [Reset = 0h]

BGCRC_START_ADDR is shown in Figure 7-10 and described in Table 7-11.

Return to the Summary Table.

Start address for the BGCRC check

Figure 7-10. BGCRC_START_ADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
START_ADDRESS																															
R/W-0h																															

Table 7-11. BGCRC_START_ADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	START_ADDRESS	R/W	0h	START_ADDRESS indicates the start point of the test. (For CPU_CRC, this will be the CPU address. For CLA_CRC, this will be the CLA address where the memory is mapped) Reset type: CPUx.SYSRSn

7.5.2.5 BGCRC_SEED Register (Offset = 8h) [Reset = 0h]

BGCRC_SEED is shown in Figure 7-11 and described in Table 7-12.

Return to the Summary Table.

Seed for CRC calculation

Figure 7-11. BGCRC_SEED Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEED																															
R/W-0h																															

Table 7-12. BGCRC_SEED Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SEED	R/W	0h	Initial value of CRC, this value is coiped to the CRC register on triggering CRC calculation by writing to START bit. Reset type: CPUx.SYSRSn

7.5.2.6 BGCRC_GOLDEN Register (Offset = Eh) [Reset = 0h]

BGCRC_GOLDEN is shown in Figure 7-12 and described in Table 7-13.

Return to the Summary Table.

Golden CRC to be compared against

Figure 7-12. BGCRC_GOLDEN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC_VALUE																															
R/W-0h																															

Table 7-13. BGCRC_GOLDEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CRC_VALUE	R/W	0h	Golden CRC register: If CRC check is enabled, the calcuated CRC value is compared with golden CRC and status is updated. Reset type: CPUx.SYSRSn

7.5.2.7 BGCRC_RESULT Register (Offset = 10h) [Reset = 0h]

BGCRC_RESULT is shown in Figure 7-13 and described in Table 7-14.

Return to the Summary Table.

CRC calculated

Figure 7-13. BGCRC_RESULT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC_VALUE																															
R-0h																															

Table 7-14. BGCRC_RESULT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CRC_VALUE	R	0h	CRC result register This reister value will be updated only on the completion of CRC check on a block of data as programmed by BGCRC_CTRL2.BLOCK_SIZE. Reset type: CPUx.SYSRSn

7.5.2.8 BGCRC_CURR_ADDR Register (Offset = 12h) [Reset = 0h]

BGCRC_CURR_ADDR is shown in Figure 7-14 and described in Table 7-15.

Return to the Summary Table.

Current address regsiter

Figure 7-14. BGCRC_CURR_ADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CURRENT_ADDR																															
R-0h																															

Table 7-15. BGCRC_CURR_ADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CURRENT_ADDR	R	0h	Current address from where the data is fetched. During a failure, the CURRENT_ADDR field indicates the value from where the last fetch happened. Reset type: CPUx.SYSRSn

7.5.2.9 BGCRC_WD_CFG Register (Offset = 1Ch) [Reset = 0h]

BGCRC_WD_CFG is shown in Figure 7-15 and described in Table 7-16.

Return to the Summary Table.

BGCRC windowed watchdog configuration

Figure 7-15. BGCRC_WD_CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												WDDIS			
R-0-0h												R/W-0h			

Table 7-16. BGCRC_WD_CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R-0	0h	Reserved
3-0	WDDIS	R/W	0h	1010: CRC Watchdog counter is disabled. Any other value: CRC watchdog is enabled Watchdog is an upcounter and starts counting when BGCRC_EN.START is asserted. Watchdog continues to count during TEST_HALT state also(BGCRC_CTRL2.TEST_HALT = "1010"). CRC watchdog can be disabled during TEST_HALT by explicit configuration. (BGCRC_WD_CFG.WDDIS = 1010). Once the watchdog is disabled and re-enabled, watchdog count resumes from the previous disabled point. Reset type: CPUx.SYSRSn

7.5.2.10 BGCRC_WD_MIN Register (Offset = 1Eh) [Reset = 0h]

BGCRC_WD_MIN is shown in Figure 7-16 and described in Table 7-17.

Return to the Summary Table.

BGCRC windowed watchdog min value

Figure 7-16. BGCRC_WD_MIN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MINVAL																															
R/W-0h																															

Table 7-17. BGCRC_WD_MIN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MINVAL	R/W	0h	If the CRC computation completes before BGCRC_WD_MIN.MINVAL FAIL_STATUS.WD_UNDERFLOW flag gets set. Reset type: CPUx.SYSRSn

7.5.2.11 BGCRC_WD_MAX Register (Offset = 20h) [Reset = FFFFFFFFh]

BGCRC_WD_MAX is shown in Figure 7-17 and described in Table 7-18.

Return to the Summary Table.

BGCRC windowed watchdog max value

Figure 7-17. BGCRC_WD_MAX Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAXVAL																															
R/W-FFFFFFFFh																															

Table 7-18. BGCRC_WD_MAX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MAXVAL	R/W	FFFFFFFFh	If the CRC computation doesn't complete before BGCRC_WD_MIN.MAXVAL FAIL_STATUS.WD_OVERFLOW flag gets set. Reset type: CPUx.SYSRSn

7.5.2.12 BGCRC_WD_CNT Register (Offset = 22h) [Reset = 0h]

BGCRC_WD_CNT is shown in Figure 7-18 and described in Table 7-19.

Return to the Summary Table.

BGCRC windowed watchdog count

Figure 7-18. BGCRC_WD_CNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WD_CNT																															
R-0h																															

Table 7-19. BGCRC_WD_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WD_CNT	R	0h	CRC windowed watchdog counter value Counter value freezes at the end of CRC computation and will be reloaded only by BGCRC_EN.START = "1010" configuration. BGCRC_WD_CNT register freezes when a failure occurs. Reset type: CPUx.SYSRSn

7.5.2.13 BGCRC_NMIFLG Register (Offset = 2Ah) [Reset = 0h]

BGCRC_NMIFLG is shown in Figure 7-19 and described in Table 7-20.

Return to the Summary Table.

BGCRC NMI flag register

Figure 7-19. BGCRC_NMIFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED	WD_OVERFLOW	WD_UNDERFLOW	CORRECTABLE_ERR	UNCORRECTABLE_ERR	CRC_FAIL	RESERVED	RESERVED
R-0-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0-0h	R-0-0h

Table 7-20. BGCRC_NMIFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R-0	0h	Reserved
6	WD_OVERFLOW	R	0h	Windowed watchdog Overflow. 1 : Test did not complete before BGCRC_WD_MAX.MAXVAL 0 : No such errors Reset type: CPUx.SYSRSn
5	WD_UNDERFLOW	R	0h	Windowed watchdog underflow. 1 : Test completed before BGCRC_WD_MIN.MINVAL 0 : No such errors Reset type: CPUx.SYSRSn
4	CORRECTABLE_ERR	R	0h	Correctable error indication: 0 : No ECC correctable error during memory read 1 : Correctable ECC error during memory read Note: ECC computation is done during every memory read. (Correctable errors are not ignored since the module doesn't write-back corrected value. The error remains in the memory and required corrective action need to be taken by CPU/CLA as part of ISR) Reset type: CPUx.SYSRSn
3	UNCORRECTABLE_ERR	R	0h	Uncorrectable error indication: 0 : No ECC-uncorrectable/Parity error during memory read 1 : ECC-uncorrectable/Parity error during memory read Note: ECC/Parity check is done during every memory read. Reset type: CPUx.SYSRSn
2	CRC_FAIL	R	0h	CRC FAIL interrupt 0 : No failure in CRC check. 1 : CRC check failure Note: Comparion is enabled only after CRC calc is completed Reset type: CPUx.SYSRSn
1	RESERVED	R-0	0h	Reserved
0	RESERVED	R-0	0h	Reserved

7.5.2.14 BGCRC_NMICLR Register (Offset = 2Ch) [Reset = 0h]

BGCRC_NMICLR is shown in Figure 7-20 and described in Table 7-21.

Return to the Summary Table.

BGCRC NMI flag clear register

Figure 7-20. BGCRC_NMICLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED	WD_OVERFLOW	WD_UNDERFLOW	CORRECTABLE_ERR	UNCORRECTABLE_ERR	CRC_FAIL	RESERVED	RESERVED
R-0-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0-0h	R-0-0h

Table 7-21. BGCRC_NMICLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R-0	0h	Reserved
6	WD_OVERFLOW	R-0/W1S	0h	Clear WD_OVERFLOW NMI flag 0 No effect 1 Clears NMI flag Reset type: CPUx.SYSRSn
5	WD_UNDERFLOW	R-0/W1S	0h	Clear WD_UNDERFLOW NMI flag 0 No effect 1 Clears NMI flag Reset type: CPUx.SYSRSn
4	CORRECTABLE_ERR	R-0/W1S	0h	Clear CORRECTABLE_ERR NMI flag 0 No effect 1 Clears NMI flag Reset type: CPUx.SYSRSn
3	UNCORRECTABLE_ERR	R-0/W1S	0h	Clear UNCORRECTABLE_ERROR NMI flag 0 No effect 1 Clears NMI flag Reset type: CPUx.SYSRSn
2	CRC_FAIL	R-0/W1S	0h	Clear CRC_FAIL NMI flag 0 No effect 1 Clears NMI flag Reset type: CPUx.SYSRSn
1	RESERVED	R-0	0h	Reserved
0	RESERVED	R-0	0h	Reserved

7.5.2.15 BGCRC_NMIFRC Register (Offset = 2Eh) [Reset = 0h]

BGCRC_NMIFRC is shown in Figure 7-21 and described in Table 7-22.

Return to the Summary Table.

BGCRC NMI flag force register

Figure 7-21. BGCRC_NMIFRC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED	WD_OVERFLOW	WD_UNDERFLOW	CORRECTABLE_ERR	UNCORRECTABLE_ERR	CRC_FAIL	RESERVED	RESERVED
R-0-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0-0h	R-0-0h

Table 7-22. BGCRC_NMIFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R-0	0h	Reserved
6	WD_OVERFLOW	R-0/W1S	0h	Force WD_OVERFLOW NMI flag 0 No effect 1 force NMI flag Reset type: CPUx.SYSRSn
5	WD_UNDERFLOW	R-0/W1S	0h	Force WD_UNDERFLOW NMI flag 0 No effect 1 force NMI flag Reset type: CPUx.SYSRSn
4	CORRECTABLE_ERR	R-0/W1S	0h	Force CORRECTABLE_ERR NMI flag 0 No effect 1 force NMI flag Reset type: CPUx.SYSRSn
3	UNCORRECTABLE_ERR	R-0/W1S	0h	Force UNCORRECTABLE_ERR NMI flag 0 No effect 1 force NMI flag Reset type: CPUx.SYSRSn
2	CRC_FAIL	R-0/W1S	0h	Force CRC_FAIL NMI flag 0 No effect 1 force NMI flag Reset type: CPUx.SYSRSn
1	RESERVED	R-0	0h	Reserved
0	RESERVED	R-0	0h	Reserved

7.5.2.16 BGCRC_INTEN Register (Offset = 34h) [Reset = 0h]

BGCRC_INTEN is shown in Figure 7-22 and described in Table 7-23.

Return to the Summary Table.

Interrupt enable

Figure 7-22. BGCRC_INTEN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	WD_OVERFLOW	WD_UNDERFLOW	CORRECTABLE_ERR	UNCORRECTABLE_ERR	CRC_FAIL	TEST_DONE	RESERVED
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0-0h

Table 7-23. BGCRC_INTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	WD_OVERFLOW	R/W	0h	0 WD_OVERFLOW Interrupt disabled 1 WD_OVERFLOW Interrupt enabled Reset type: CPUx.SYSRSn
5	WD_UNDERFLOW	R/W	0h	0 WD_UNDERFLOW Interrupt disabled 1 WD_UNDERFLOW Interrupt enabled Reset type: CPUx.SYSRSn
4	CORRECTABLE_ERR	R/W	0h	0 CORRECTABLE_ERR Interrupt disabled 1 CORRECTABLE_ERR Interrupt enabled Reset type: CPUx.SYSRSn
3	UNCORRECTABLE_ERR	R/W	0h	0 UNCORRECTABLE_ERR Interrupt disabled 1 UNCORRECTABLE_ERR Interrupt enabled Reset type: CPUx.SYSRSn
2	CRC_FAIL	R/W	0h	0 CRC_FAIL Interrupt disabled 1 CRC_FAIL Interrupt enabled Reset type: CPUx.SYSRSn
1	TEST_DONE	R/W	0h	0 TEST_DONE Interrupt disabled 1 TEST_DONE Interrupt enabled Reset type: CPUx.SYSRSn
0	RESERVED	R-0	0h	Reserved

7.5.2.17 BGCRC_INTFLG Register (Offset = 36h) [Reset = 0h]

BGCRC_INTFLG is shown in Figure 7-23 and described in Table 7-24.

Return to the Summary Table.

Interrupt flag

Figure 7-23. BGCRC_INTFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED	WD_OVERFLOW	WD_UNDERFLOW	CORRECTABLE_ERR	UNCORRECTABLE_ERR	CRC_FAIL	TEST_DONE	INT
R-0-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 7-24. BGCRC_INTFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R-0	0h	Reserved
6	WD_OVERFLOW	R	0h	Windowed watchdog Overflow. 1 : Test did not completed before BGCRC_WD_MAX.MAXVAL 0 : No such errors Reset type: CPUx.SYSRSn
5	WD_UNDERFLOW	R	0h	Windowed watchdog underflow. 1 : Test completed before BGCRC_WD_MIN.MINVAL 0 : No such errors Reset type: CPUx.SYSRSn
4	CORRECTABLE_ERR	R	0h	Correctable error indication: 0 : No ECC correctable error during memory read 1 : Correctable ECC error during memory read Note: ECC computation is done during every memory read. (Correctable errors are not ignored since the module doesn't write-back corrected value. The error remains in the memory and required corrective action need to be taken by CPU/CLA as part of ISR) Reset type: CPUx.SYSRSn
3	UNCORRECTABLE_ERR	R	0h	uncorrectable error indication: 0 : No ECC-uncorrectable/Parity error during memory read 1 : ECC-uncorrectable/Parity error during memory read Note: ECC/Parity check is done during every memory read. Reset type: CPUx.SYSRSn
2	CRC_FAIL	R	0h	CRC fail interrupt 0 : No failure of CRC check 1 : CRC check failure Note: Comparion is enabled only after CRC calc is completed Reset type: CPUx.SYSRSn
1	TEST_DONE	R	0h	Done Interrupt Status flag 0 CRC calculation is in progress or CRC module is idle. 1 CRC calculation is done. Note: TEST_DONE flag will get set on CRC calculation completion even in case of CRC mismatch Reset type: CPUx.SYSRSn

Table 7-24. BGCRC_INTFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INT	R	0h	Global Interrupt Status flag 0 No interrupt generated 1 Interrupt was generated Reset type: CPUx.SYSRSn

7.5.2.18 BGCRC_INTCLR Register (Offset = 38h) [Reset = 0h]

BGCRC_INTCLR is shown in Figure 7-24 and described in Table 7-25.

Return to the Summary Table.

Interrupt flag clear

Figure 7-24. BGCRC_INTCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED	WD_OVERFLOW	WD_UNDERFLOW	CORRECTABLE_ERR	UNCORRECTABLE_ERR	CRC_FAIL	TEST_DONE	INT
R-0-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 7-25. BGCRC_INTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R-0	0h	Reserved
6	WD_OVERFLOW	R-0/W1S	0h	Clear interrupt flag 0 No effect 1 Clears the interrupt flag Reset type: CPUx.SYSRSn
5	WD_UNDERFLOW	R-0/W1S	0h	Clear interrupt flag 0 No effect 1 Clears the interrupt flag Reset type: CPUx.SYSRSn
4	CORRECTABLE_ERR	R-0/W1S	0h	Clear interrupt flag 0 No effect 1 Clears the interrupt flag Reset type: CPUx.SYSRSn
3	UNCORRECTABLE_ERR	R-0/W1S	0h	Clear interrupt flag 0 No effect 1 Clears the interrupt flag Reset type: CPUx.SYSRSn
2	CRC_FAIL	R-0/W1S	0h	Clear interrupt flag 0 No effect 1 Clears the interrupt flag Reset type: CPUx.SYSRSn
1	TEST_DONE	R-0/W1S	0h	Clear interrupt flag 0 No effect 1 Clears the interrupt flag Reset type: CPUx.SYSRSn
0	INT	R-0/W1S	0h	Global Interrupt Clear 0 No effect 1 Clears the interrupt flag and enables further interrupts to be generated if an event flags is set to 1. Reset type: CPUx.SYSRSn

7.5.2.19 BGCRC_INTFRC Register (Offset = 3Ah) [Reset = 0h]

BGCRC_INTFRC is shown in Figure 7-25 and described in Table 7-26.

Return to the Summary Table.

Interrupt flag force

Figure 7-25. BGCRC_INTFRC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED	WD_OVERFLOW	WD_UNDERFLOW	CORRECTABLE_ERR	UNCORRECTABLE_ERR	CRC_FAIL	TEST_DONE	RESERVED
R-0-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0-0h

Table 7-26. BGCRC_INTFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R-0	0h	Reserved
6	WD_OVERFLOW	R-0/W1S	0h	Force interrupt flag 0 No effect 1 force the interrupt flag Reset type: CPUx.SYSRSn
5	WD_UNDERFLOW	R-0/W1S	0h	Force interrupt flag 0 No effect 1 force the interrupt flag Reset type: CPUx.SYSRSn
4	CORRECTABLE_ERR	R-0/W1S	0h	Force interrupt flag 0 No effect 1 force the interrupt flag Reset type: CPUx.SYSRSn
3	UNCORRECTABLE_ERR	R-0/W1S	0h	Force interrupt flag 0 No effect 1 force the interrupt flag Reset type: CPUx.SYSRSn
2	CRC_FAIL	R-0/W1S	0h	Force interrupt flag 0 No effect 1 force the interrupt flag Reset type: CPUx.SYSRSn
1	TEST_DONE	R-0/W1S	0h	Force interrupt flag 0 No effect 1 force the interrupt flag Reset type: CPUx.SYSRSn
0	RESERVED	R-0	0h	Reserved

7.5.2.20 BGCRC_LOCK Register (Offset = 3Ch) [Reset = 0h]

BGCRC_LOCK is shown in Figure 7-26 and described in Table 7-27.

Return to the Summary Table.

BGCRC register map lockconfiguration

Figure 7-26. BGCRC_LOCK Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	BGCRC_INTFRC	RESERVED	RESERVED	BGCRC_INTEN	RESERVED	RESERVED
R-0-0h	R-0-0h	R/W-0h	R-0-0h	R-0-0h	R/W-0h	R-0-0h	R-0-0h
23	22	21	20	19	18	17	16
BGCRC_NMIFRC	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	BGCRC_WD_MAX
R/W-0h	R-0-0h	R-0-0h	R-0-0h	R-0-0h	R-0-0h	R-0-0h	R/W-0h
15	14	13	12	11	10	9	8
BGCRC_WD_MIN	BGCRC_WD_CFG	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R-0-0h	R-0-0h	R-0-0h	R-0-0h	R-0-0h	R-0-0h
7	6	5	4	3	2	1	0
BGCRC_GOLDEN	RESERVED	RESERVED	BGCRC_SEED	BGCRC_START_ADDR	BGCRC_CTRL2	BGCRC_CTRL1	BGCRC_EN
R/W-0h	R-0-0h	R-0-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 7-27. BGCRC_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R-0	0h	Reserved
30	RESERVED	R-0	0h	Reserved
29	BGCRC_INTFRC	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
28	RESERVED	R-0	0h	Reserved
27	RESERVED	R-0	0h	Reserved
26	BGCRC_INTEN	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
25	RESERVED	R-0	0h	Reserved
24	RESERVED	R-0	0h	Reserved
23	BGCRC_NMIFRC	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
22	RESERVED	R-0	0h	Reserved
21	RESERVED	R-0	0h	Reserved
20	RESERVED	R-0	0h	Reserved
19	RESERVED	R-0	0h	Reserved
18	RESERVED	R-0	0h	Reserved
17	RESERVED	R-0	0h	Reserved
16	BGCRC_WD_MAX	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn

Table 7-27. BGCRC_LOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15	BGCRC_WD_MIN	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
14	BGCRC_WD_CFG	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
13	RESERVED	R-0	0h	Reserved
12	RESERVED	R-0	0h	Reserved
11	RESERVED	R-0	0h	Reserved
10	RESERVED	R-0	0h	Reserved
9	RESERVED	R-0	0h	Reserved
8	RESERVED	R-0	0h	Reserved
7	BGCRC_GOLDEN	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
6	RESERVED	R-0	0h	Reserved
5	RESERVED	R-0	0h	Reserved
4	BGCRC_SEED	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
3	BGCRC_START_ADDR	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
2	BGCRC_CTRL2	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
1	BGCRC_CTRL1	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
0	BGCRC_EN	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn

7.5.2.21 BGCRC_COMMIT Register (Offset = 3Eh) [Reset = 0h]

BGCRC_COMMIT is shown in Figure 7-27 and described in Table 7-28.

Return to the Summary Table.

BGCRC register map commit configuration

Figure 7-27. BGCRC_COMMIT Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	BGCRC_INTFRC	RESERVED	RESERVED	BGCRC_INTEN	RESERVED	RESERVED
R-0-0h	R-0-0h	R/WSonce-0h	R-0-0h	R-0-0h	R/WSonce-0h	R-0-0h	R-0-0h
23	22	21	20	19	18	17	16
BGCRC_NMIFRC	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	BGCRC_WD_MAX
R/WSonce-0h	R-0-0h	R-0-0h	R-0-0h	R-0-0h	R-0-0h	R-0-0h	R/WSonce-0h
15	14	13	12	11	10	9	8
BGCRC_WD_MIN	BGCRC_WD_CFG	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/WSonce-0h	R/WSonce-0h	R-0-0h	R-0-0h	R-0-0h	R-0-0h	R-0-0h	R-0-0h
7	6	5	4	3	2	1	0
BGCRC_GOLDEN	RESERVED	RESERVED	BGCRC_SEED	BGCRC_START_ADDR	BGCRC_CTRL2	BGCRC_CTRL1	BGCRC_EN
R/WSonce-0h	R-0-0h	R-0-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 7-28. BGCRC_COMMIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R-0	0h	Reserved
30	RESERVED	R-0	0h	Reserved
29	BGCRC_INTFRC	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
28	RESERVED	R-0	0h	Reserved
27	RESERVED	R-0	0h	Reserved
26	BGCRC_INTEN	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
25	RESERVED	R-0	0h	Reserved
24	RESERVED	R-0	0h	Reserved
23	BGCRC_NMIFRC	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
22	RESERVED	R-0	0h	Reserved
21	RESERVED	R-0	0h	Reserved
20	RESERVED	R-0	0h	Reserved
19	RESERVED	R-0	0h	Reserved
18	RESERVED	R-0	0h	Reserved

Table 7-28. BGCRC_COMMIT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	RESERVED	R-0	0h	Reserved
16	BGCRC_WD_MAX	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
15	BGCRC_WD_MIN	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
14	BGCRC_WD_CFG	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
13	RESERVED	R-0	0h	Reserved
12	RESERVED	R-0	0h	Reserved
11	RESERVED	R-0	0h	Reserved
10	RESERVED	R-0	0h	Reserved
9	RESERVED	R-0	0h	Reserved
8	RESERVED	R-0	0h	Reserved
7	BGCRC_GOLDEN	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
6	RESERVED	R-0	0h	Reserved
5	RESERVED	R-0	0h	Reserved
4	BGCRC_SEED	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
3	BGCRC_START_ADDR	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
2	BGCRC_CTRL2	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
1	BGCRC_CTRL1	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
0	BGCRC_EN	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn

7.5.3 BGCRC Registers to Driverlib Functions

Table 7-29. BGCRC Registers to Driverlib Functions

File	Driverlib Function
EN	
bgcrc.h	BGCRC_start
bgcrc.h	BGCRC_getRunStatus
CTRL1	
bgcrc.h	BGCRC_setConfig
CTRL2	
bgcrc.h	BGCRC_setRegion
bgcrc.h	BGCRC_halt
bgcrc.h	BGCRC_resume
START_ADDR	
bgcrc.h	BGCRC_setRegion
SEED	
bgcrc.h	BGCRC_setSeedValue
GOLDEN	
bgcrc.h	BGCRC_setGoldenCRCValue
RESULT	
bgcrc.h	BGCRC_getResult
CURR_ADDR	
bgcrc.h	BGCRC_getCurrentAddress
WD_CFG	
bgcrc.h	BGCRC_enableWatchdog
bgcrc.h	BGCRC_disableWatchdog
WD_MIN	
bgcrc.h	BGCRC_setWatchdogWindow
WD_MAX	
bgcrc.h	BGCRC_setWatchdogWindow
WD_CNT	
bgcrc.h	BGCRC_getWatchdogCounterValue
NMIFLG	
bgcrc.h	BGCRC_getNMIStatus
NMICLR	
bgcrc.h	BGCRC_clearNMIStatus
NMIFRC	
bgcrc.h	BGCRC_forceNMI
INTEN	
bgcrc.h	BGCRC_enableInterrupt
bgcrc.h	BGCRC_disableInterrupt
INTFLG	
bgcrc.h	BGCRC_getInterruptStatus
INTCLR	
bgcrc.h	BGCRC_clearInterruptStatus
INTFRC	
bgcrc.h	BGCRC_forceInterrupt
LOCK	

Table 7-29. BGCRC Registers to Driverlib Functions (continued)

File	Driverlib Function
bgcrc.h	BGCRC_lockRegister
bgcrc.h	BGCRC_unlockRegister
COMMIT	
bgcrc.h	BGCRC_commitRegisterLock

Chapter 8

Control Law Accelerator (CLA)

The Control Law Accelerator (CLA) Type-2 is an independent, fully-programmable, 32-bit floating-point math processor that brings concurrent control-loop execution to the C28x family. The low interrupt latency of the CLA allows the CLA to read ADC samples "just-in-time." This significantly reduces the ADC sample to output delay to enable faster system response and higher MHz control loops. By using the CLA to service time-critical control loops, the main CPU is free to perform other system tasks such as communications and diagnostics. This chapter provides an overview of the architectural structure and components of the control law accelerator.

8.1 Introduction

The Control Law Accelerator extends the capabilities of the C28x CPU by adding parallel processing. Time-critical control loops serviced by the CLA can achieve low ADC sample to output delay. Thus, the CLA enables faster system response and higher frequency control loops. Utilizing the CLA for time-critical tasks frees up the main CPU to perform other system and communication functions concurrently.

8.1.1 Features

The following is a list of major features of the CLA:

- C compilers are available for CLA software development.
- Clocked at the same rate as the main CPU (SYSCLKOUT).
- An independent architecture allowing CLA algorithm execution independent of the main C28x CPU.
  - Complete bus architecture:
    - Program Address Bus (PAB) and Program Data Bus (PDB)
    - Data Read Address Bus (DRAB), Data Read Data Bus (DRDB), Data Write Address Bus (DWAB), and Data Write Data Bus (DWDB)
  - Independent eight stage pipeline.
  - 16-bit program counter (MPC)
  - Four 32-bit result registers (MR0-MR3)
  - Two 16-bit auxiliary registers (MAR0, MAR1)
  - Status register (MSTF)
- Instruction set includes:
  - IEEE single-precision (32-bit) floating-point math operations
  - Floating-point math with parallel load or store
  - Floating-point multiply with parallel add or subtract
  - 1/X and 1/sqrt(X) estimations
  - Data type conversions.
  - Conditional branch and call
  - Data load/store operations
- The CLA program code can consist of up to eight tasks or interrupt service routines, or seven tasks and a main background task.
  - The start address of each task is specified by the MVECT registers.
  - No limit on task size as long as the tasks fit within the configurable CLA program memory space.
  - One task is serviced at a time until completion. There is no nesting of tasks.
  - Upon task completion a task-specific interrupt is flagged within the PIE.
  - When a task finishes the next highest-priority pending task is automatically started.
  - The Type-2 CLA can have a main task that runs continuously in the background, while other high priority events trigger a foreground task.
- Task trigger mechanisms:
  - C28x CPU using the IACK instruction
  - Task1 to Task8: up to 256 possible trigger sources from peripherals connected to the shared bus on which the CLA assumes secondary ownership.
  - Task8 can be set to be the background task, while Tasks 1 through 7 take peripheral triggers.
- Memory and Shared Peripherals:
  - Two dedicated message RAMs for communication between the CLA and the main CPU.
  - Two dedicated message RAMs for communication between the CLA and the DMA.
  - The C28x CPU can map CLA program and data memory to the main CPU space or CLA space.

8.1.2 CLA Related Collateral

Foundational Materials

- C2000 Academy - CLA

- C2000 CLA C Compiler Series (Video)
- CLA Hands On Workshop (Video)
- CLA usage in Valley Switching Boost Power Factor Correction (PFC) Reference Design (Video)
- *Enhancing the Computational Performance of the C2000™ Microcontroller Family Application Report*

Getting Started Materials

- CLA Software Development Guide
- *Software Examples to Showcase Unique Capabilities of TI's C2000™ CLA Application Report*

Expert Materials

- *Digital Control of Two Phase Interleaved PFC and Motor Drive Using MCU With CLA Application Report*
- *Sensorless Field Oriented Control:3-Phase Perm.Magnet Synch. Motors With CLA Application Report*

8.1.3 Block Diagram

Figure 8-1 is a block diagram of the CLA.

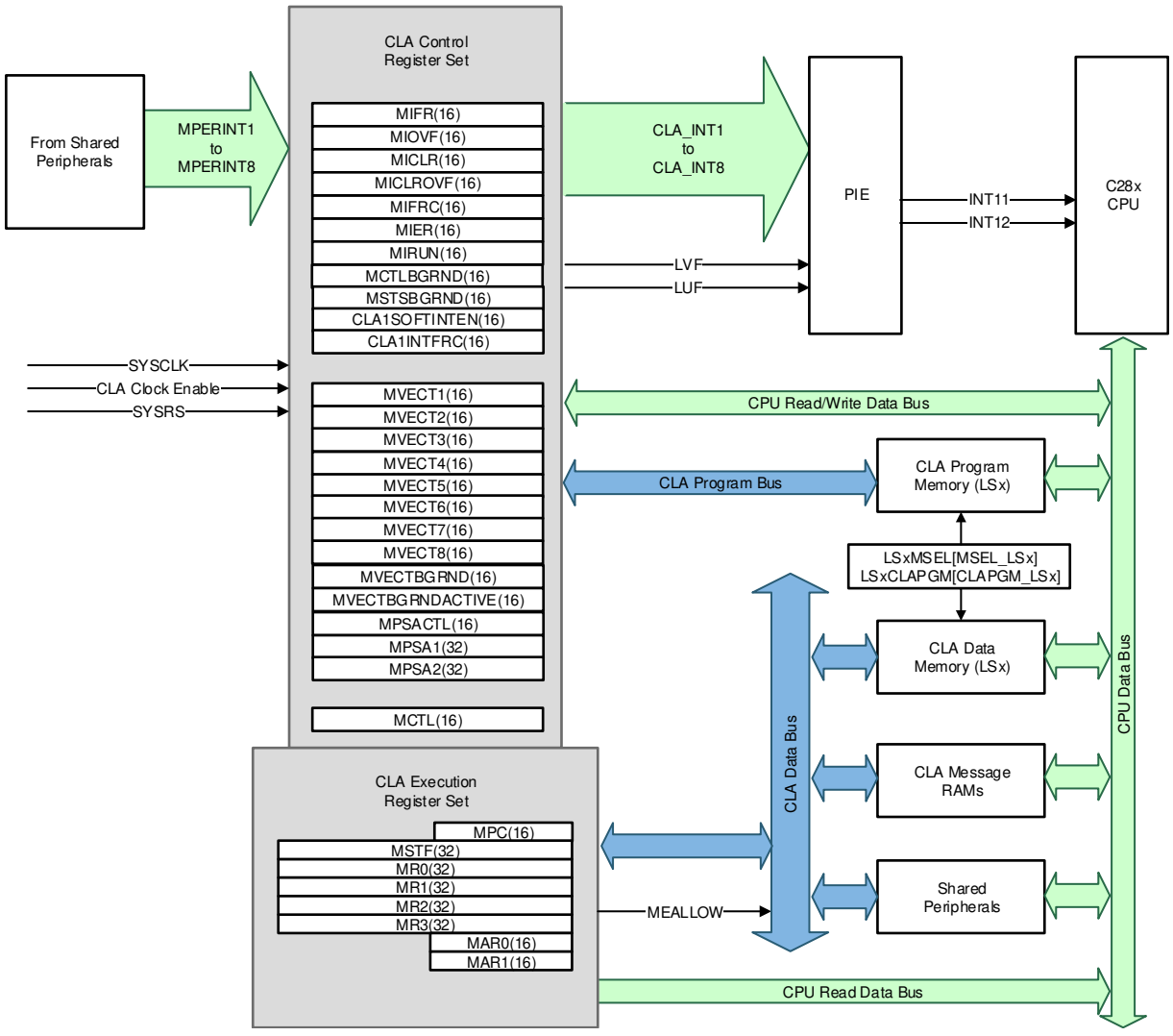


Figure 8-1. CLA (Type 2) Block Diagram

8.2 CLA Interface

This section describes how the C28x main CPU can interface to the CLA and conversely.

8.2.1 CLA Memory

The CLA can access three types of memory: program, data and message RAMs. The behavior and arbitration for each type of memory is described in this chapter. The CLA RAMs are protected by the DCSM module. Refer to the *Dual Code Security Module (DCSM)* section of the *System Control and Interrupts* chapter for more details on the security scheme.

- **CLA Program Memory**

  The CLA program can be loaded with any of the local shared memories (LSxRAM). At reset, all memory blocks are mapped to the CPU. While mapped to the CPU space, the CPU can copy the CLA program code into the memory. During debug, the memory can also be loaded directly by the Code Composer Studio™ IDE.

  Once the memory is initialized with CLA code, the CPU maps the memory to the CLA program space by:
  1. Assigning ownership of the memory block to the CLA by writing a 1 to the memory block's MemCfgRegs.LSxMSEL[MSEL_LSx] bit.
  2. Specifying the memory block as a code block for the CLA by writing a 1 to the MemCfgRegs.LSxCLAPGM[CLAPGM_LSx] bit.

  When a memory block is configured as CLA program memory, debug accesses are allowed only on cycles where the CLA is not fetching a new instruction. A detailed explanation of the memory configurations and access arbitration (CPU, CLA, and DEBUG) process can be found in the *Memory Controller Module* section of the *System Control and Interrupts* chapter.

  All CLA program fetches are performed as 32-bit read operations and all opcodes must be aligned to an even address. Since all CLA opcodes are 32-bits, this alignment occurs naturally.

- **CLA Data Memory**

  Any of the device's LSxRAMs can serve as data memory blocks to the CLA. At reset, all blocks are mapped to the CPU memory space, whereby the CPU can initialize the memory with data tables, coefficients, and so on, for the CLA to use.

  Once the memory is initialized with CLA data, the CPU maps the memory to the CLA data space by:
  1. Assigning ownership of the memory block to the CLA by writing a 1 to the memory block's MemCfgRegs.LSxMSEL[MSEL_LSx] bit.
  2. Specifying the memory block as a data block for the CLA by writing a 0 to the MemCfgRegs.LSxCLAPGM[CLAPGM_LSx] bit. The value of this bit at reset is 0.

  When a memory block is configured as CLA data memory, CLA read and write accesses are arbitrated along with CPU accesses. The user has the option of turning on CPU fetch or write protection to the memory by writing to the appropriate bits of the MemCfgRegs.LSxACCPROTx registers. A detailed explanation of the memory configurations and access arbitration (CPU, CLA, and DEBUG) process can be found in the *Memory Controller Module* section of the *System Control and Interrupts* chapter.
- **CLA Shared Message RAMs**

  There are two memory blocks for data sharing and communication between the CLA and the CPU . The message RAMs are always mapped to both CPU and CLA memory spaces, and only data access is allowed; no program fetches can be performed.
  - **CLA to CPU Message RAM:** The CLA can use this block to pass data to the CPU. This block is both readable and writable by the CLA. This block is also readable by the CPU but writes by the CPU are ignored.
  - **CPU to CLA Message RAM:** The CPU can use this block to pass data and messages to the CLA. This message RAM is both readable and writable by the CPU. The CLA can perform reads but writes by the CLA are ignored.

8.2.2 CLA Memory Bus

The CLA has dedicated bus architecture similar to that of the C28x CPU where there are separate program read, data read, and data write buses. Thus, there can be simultaneous instruction fetch, data read, and data write in a single cycle. Like the C28x CPU, the CLA expects memory logic to align any 32-bit read or write to an even address. If the address-generation logic generates an odd address, the CLA can begin reading or writing at the previous even address. This alignment does not affect the address values generated by the address-generation logic.

- **CLA Program Bus**

  The CLA program bus has an access range of 32K 32-bit instructions. Since all CLA instructions are 32 bits, this bus always fetches 32 bits at a time and the opcodes must be even-word aligned. The amount of program space available for the CLA is limited to the number of available LSxRAM blocks. This number is device-dependent and can be described in the device-specific data manual.

- **CLA Data Read Bus**

  The CLA data read bus has a 64K x 16 address range. The bus can perform 16 or 32-bit reads and can automatically stall if there are memory access conflicts. The data read bus has access to both the message RAMs, CLA data memory, and the shared peripherals.

- **CLA Data Write Bus**

  The CLA data write bus has a 64K x 16 address range. This bus can perform 16 or 32-bit writes. The bus can automatically stall if there are memory access conflicts. The data write bus has access to the CLA to CPU message RAM, CLA data memory, and the shared peripherals.

8.2.3 Shared Peripherals and EALLOW Protection

For a given CPU subsystem, the CPU, CLA, and DMA can share access to some peripherals. There is a 3-way arbitration among the different master's that is described in Section 8.3. Each peripheral has an access control register with two bit fields, CPUnAC, CLAnAC, and DMAnAC (n being the instance) that determine what kind of access is given to that particular master.

Note

The CLA read access time to the bus is 2-wait states while write access is 0-wait.

Refer to the device data sheet for the list of peripherals connected to the bus.

Several peripheral control registers are protected from spurious 28x CPU writes by the EALLOW protection mechanism. These same registers are also protected from spurious CLA writes. The EALLOW bit in the CPU status register 1 (ST1) indicates the state of protection for the CPU. Likewise, the MEALLOW bit in the CLA status register (MSTF) indicates the state of write protection for the CLA. The MEALLOW CLA instruction enables write access by the CLA to EALLOW protected registers. Likewise, the MEDIS CLA instruction disables write access. This way the CLA can enable and disable write access independent of the CPU.

The ADC offers the option to generate an early interrupt pulse at the start of a sample conversion. If this option is used to start an ADC-triggered CLA task, use the intervening cycles until the completion of the conversion to perform preliminary calculations or loads and stores before finally reading the ADC value. The CLA pipeline activity for this scenario is shown in Section 8.5.

8.2.4 CLA Tasks and Interrupt Vectors

The CLA program code is divided up into tasks or interrupt service routines. Tasks do not have a fixed starting location or length. The CLA program memory can be divided up as desired. The CLA uses the contents of the interrupt vectors (MVECT1 to MVECT8) to determine where a task begins; tasks are terminated by the MSTOP instruction.

The CLA supports eight tasks. Task 1 has the highest priority and task 8 has the lowest priority. The Type-2 CLA offers the option of setting the lowest priority task, for example, task 8, as a background task that, once triggered, runs continuously until the user either terminates the task or resets the CLA or the device. The remaining tasks, 1 through 7, maintain their priority levels and interrupt the background task when triggered.

The background task is enabled by setting the BGEN bit in the MCTLBGRND register; this causes the hardware to disable task 8 in the MIER register. The background task derives the interrupt vector from the MVECTBGRND register instead of MVECT8.

A task can be requested by a peripheral interrupt or by software:

- **Peripheral interrupt trigger**

  Each task can be triggered by software-selectable interrupt sources. The trigger for each task is defined by writing an appropriate value to the DmaClaSrcSelRegs.CLA1TASKSRCSELx[TASKx] bit field. Each option specifies an interrupt source from a specific peripheral on the shared bus. The peripheral interrupt triggers are listed in Table 8-1.

  For example, task 1 (MVECT1) can be set to trigger on EPWMINT1 by writing 36 to DmaClaSrcSelRegs.CLA1TASKSRCSEL1.TASK1. To disable the triggering of a task by a peripheral, set the DmaClaSrcSelRegs.CLA1TASKSRCSELx[TASKx] bit field to 0. Note that a CLA task only triggers on a level transition (an edge) of the configured interrupt source.

Table 8-1. Configuration Options

Select Value	CLA Trigger Source
0	CLA_SOFTWARE_TRIGGER
1	ADCAINT1
2	ADCAINT2
3	ADCAINT3
4	ADCAINT4
5	ADCA_EVT_INT
6	ADCBINT1
7	ADCBINT2
8	ADCBINT3
9	ADCBINT4
10	ADCB_EVT_INT
11	ADCCINT1
12	ADCCINT2
13	ADCCINT3
14	ADCCINT4
15	ADCC_EVT_INT
16	ADCDINT1
17	ADCDINT2
18	ADCDINT3

Table 8-1. Configuration Options (continued)

Select Value	CLA Trigger Source
19	ADCDINT4
20	ADCD_EVT_INT
21-28	Reserved
29	XINT1
30	XINT2
31	XINT3
32	XINT4
33	XINT5
34-35	Reserved
36	EPWM1_INT
37	EPWM2_INT
38	EPWM3_INT
39	EPWM4_INT
40	EPWM5_INT
41	EPWM6_INT
42	EPWM7_INT
43	EPWM8_INT
44	EPWM9_INT
45	EPWM10_INT
46	EPWM11_INT
47	EPWM12_INT
48	EPWM13_INT
49	EPWM14_INT
50	EPWM15_INT
51	EPWM16_INT
52	MCANA_FEVT0
53	MCANA_FEVT1
54	MCANA_FEVT2
55-67	Reserved
68	CPU_TINT0
69	CPU_TINT1
70	CPU_TINT2
71	MCBSPA_TX
72	MCBSPA_RX
73	MCBSPB_TX
74	MCBSPB_RX
75	ECAP1_INT
76	ECAP2_INT

Table 8-1. Configuration Options (continued)

Select Value	CLA Trigger Source
77	ECAP3_INT
78	ECAP4_INT
79	ECAP5_INT
80	ECAP6_INT
81	ECAP7_INT
82	Reserved
83	EQEP1_INT
84	EQEP2_INT
85	EQEP3_INT
86-91	Reserved
92	ECAP6_INT2
93	ECAP7_INT3
94	Reserved
95	SD1FLT1_DRINT
96	SD1FLT2_DRINT
97-102	Reserved
103	ECAT_SYNC0
104	ECAT_SYNC1
105	PMBUSA_INT
106-108	Reserved
109	SPIA_TXINT
110	SPIA_RXINT
111	SPIB_TXINT
112	SPIB_RXINT
113	SPIC_TXINT
114	SPIC_RXINT
115	SPID_TXINT
116	SPID_RXINT
117	CLB5_INT
118	CLB6_INT
119	CLB7_INT
120	CLB8_INT
121	BGCRC_INT
122	Reserved
123	FSITXA_INT1
124	FSITXA_INT2
125	FSIRXA_INT1
126	FSIRXA_INT2

Table 8-1. Configuration Options (continued)

Select Value	CLA Trigger Source
127	CLB1_INT
128	CLB2_INT
129	CLB3_INT
130	CLB4_INT
131-142	Reserved
143	SD1FLT1_DRINT
144	SD1FLT2_DRINT
145	SD1FLT3_DRINT
146	SD1FLT4_DRINT
147	SD2FLT1_DRINT
148	SD2FLT2_DRINT
149	SD2FLT3_DRINT
150	SD2FLT4_DRINT
151-154	Reserved
155	FSITXB_INT1
156	FSITXB_INT2
157	FSIRXB_INT1
158	FSIRXB_INT2
159	FSIRXC_INT1
160	FSIRXC_INT2
161	FSIRXD_INT1
162	FSIRXD_INT2
163	FSIRXE_INT1
164	FSIRXE_INT2
165	FSIRXF_INT1
166	FSIRXF_INT2
167	FSIRXG_INT1
168	FSIRXG_INT2
169	FSIRXH_INT1
170	FSIRXH_INT2
171-255	Reserved

- **Software Trigger**

  CPU software can trigger tasks by writing to the MIFRC register or by the IACK instruction. Using the IACK instruction is more efficient because the instruction does not require the need to issue an EALLOW to set MIFR bits. Set the MCTL[IACKE] bit to enable the IACK feature. Each bit in the operand of the IACK instruction corresponds to a task. For example, IACK #0x0001 sets bit 0 in the MIFR register to start task 1. Likewise, IACK #0x0003 set bits 0 and 1 in the MIFR register to start task 1 and task 2.
- **Background Task**

The Type-2 CLA allows the use of Task 8 as a background task that runs continuously until Task 8 disables the task or resets the device (or the CLA using a soft reset). The background task vector is given by the MVECTBGRND register and the operation is controlled by the MCTLBGRND register; the task is enabled by setting the BGEN bit to 1. Then start the task through software by writing a 1 to the BGSTART bit (TRIGEN must be 0), or through a peripheral by setting the TRIGEN bit to 1 and then setting the trigger source in the bit-field, DmaClaSrcSelRegs.CLA1TASKSRCSEL2.bit.TASK8. By default, the background task is interruptible; the highest priority pending task is executed first. When a task completes and there are not any pending tasks, the execution returns to the background task. The CLA keeps track of the branching point by saving the return address to the MVECTBGRNDACTIVE register, and then popping this address to the MPC when execution returns. Choose to make sections of the background task uninterruptible by possibly doing this with the MSETC BGINTM assembly instruction.

Subsequently, enabling interrupts with the MCLRC BGINTM instruction.

The background interrupt mask bit, BGINTM, can be queried in the MSTSBGRND register. This register also provides the current status of the background task. If the task is currently executing, the RUN bit is set to 1, if another trigger for the background task is received while the task has already started, the overflow (BGOVF) bit is set.

The CLA has their own fetch mechanism and can run and execute a task independently of the CPU. Only one task is serviced at a time; there is no nesting of tasks unless the background task in enabled, then one level of nesting is possible. The task currently running is indicated in the MIRUN register; if the background task is enabled and running, the task is reflected in the MSTSBGRND register (the RUN bit).

Interrupts that have been received but not yet serviced are indicated in the flag register (MIFR). If an interrupt request from a peripheral is received and that same task is already flagged, then the overflow flag bit is set. Overflow flags remain set until the flags are cleared by the CPU. If the CLA is idle (no task is currently running) or is executing the background task, then the highest priority interrupt request that is both flagged (MIFR) and enabled (MIER) starts.

The flow is as follows:

1. The associated RUN register bit is set (MIRUN) and the flag bit (MIFR) is cleared.
2. The CLA begins execution at the location indicated by the associated interrupt vector (MVECTx). MVECT contains the absolute 16-bit address of the task in the lower 64K memory space. If a task is interrupting the background task then the current program address is stored in the MVECTBGRNDACTIVE register before execution jumps to the task; this saved address is restored to the MPC when the task completes and execution returns to the background task.
3. The CLA executes instructions until the MSTOP instruction is found. This indicates the end of the task.
4. The MIRUN bit is cleared.
5. The task-specific interrupt to the PIE is issued. This informs the main CPU that the task has completed.
6. The CLA returns to idle (or to the background task, if enabled). Once a task completes the next highest-priority pending task is automatically serviced and this sequence repeats.

8.2.5 CLA Software Interrupt to CPU

The CLA can issue a software interrupt to the C28x CPU at any point in the code through the use of the CLA1SOFTINTEN and CLA1INTFRC registers. See Section 8.8 for a description of these registers. If a software interrupt is selected for a CLA task, then an end-of-task interrupt is not issued to the C28x CPU when that task completes.

8.3 CLA, DMA, and CPU Arbitration

Typically, CLA activity is independent of the CPU activity. Under the circumstance where the CLA, DMA, or CPU attempt to concurrently access memory or a peripheral register within the same interface, an arbitration procedure will occur. This section describes this arbitration.

The arbitration follows a fixed arbitration scheme with highest priority first:

1. DMA WRITE
2. DMA READ
3. CLA WRITE
4. CLA READ
5. CPU WRITE
6. CPU READ

Refer to the Memory Controller Module section of the *System Control and Interrupts* chapter.

8.3.1 CLA Message RAM

Message RAMs consist of four blocks:

- CLA to CPU Message RAM
- CPU to CLA Message RAM
- DMA to CLA Message RAM
- CLA to DMA Message RAM

These blocks are useful for passing data between the CLA and CPU or CLA and DMA. No opcode fetches, from either the CLA or CPU, are allowed from the message RAMs. A write protection violation is not generated if the CLA attempts to write to the CPU to CLA or DMA to CLA message RAM, but the write is ignored. The arbitration scheme for the message RAMs are the same as those for the shared memories, described in the Memory Controller Module section of the *System Control and Interrupts* chapter.

The message RAMs have the following characteristics:

- CLA to CPU Message RAM:
  The following accesses are allowed:
  - CPU reads
  - CLA data reads and writes
  - CPU debug reads and writes

  The following accesses are ignored:
  - CPU writes
- CPU to CLA Message RAM:
  The following accesses are allowed:
  - CPU reads and writes
  - CLA reads
  - CPU debug reads and writes

  The following accesses are ignored:
  - CLA writes

8.3.2 Peripheral Registers (ePWM, HRPWM, Comparator)

Accesses to the registers follow these rules:

- If both the CPU and CLA request access at the same time, then the CLA has priority and the main CPU is stalled.
- If a CPU access is in-progress and another CPU access is pending, then the CLA has priority over the pending CPU access. In this case, the CLA access begins when the current CPU access completes.
- While a CPU access is in-progress, any incoming CLA access is stalled.
- While a CLA access is in-progress, any incoming CPU access is stalled.
- A CPU write operation has priority over a CPU read operation.
- A CLA write operation has priority over a CLA read operation.
- If the CPU is performing a read-modify-write operation and the CLA performs a write to the same location, the CLA write can be lost if the operation occurs in-between the CPU read and write. For this reason, do not mix CPU and CLA accesses to same location.

8.4 CLA Configuration and Debug

This section discusses the steps necessary to configure and debug the CLA.

8.4.1 Building a CLA Application

The control law accelerator can be programmed in either CLA assembly code, using the instructions described in Section 8.7, or a reduced subset of the C language. CLA assembly code resides in the same project with C28x code. The only restriction is the CLA code must be in the assembly section. This can be easily done using the **.sect** assembly directive. This does not prevent CLA and C28x code from being linked into the same memory region in the linker command file.

System and CLA initialization are performed by the main CPU. This can typically be done in C or C++ but can also include C28x assembly code. The main CPU also copies the CLA code to the program memory and, if needed, initialize the CLA data RAMs. Once system initialization is complete and the application begins, the CLA services the interrupts using the CLA assembly code (or tasks). The main CPU can perform other tasks concurrently with CLA program execution.

The CLA Type 2 requires Codegen V16.9.1.LTS or later with the compiler switch: --cla_support=cla2.

8.4.2 Typical CLA Initialization Sequence

A typical CLA initialization sequence is performed by the main CPU as described in this section.

1. **Copy CLA code into the CLA program RAM**

   The source for the CLA code can initially reside in the Flash or a data stream from a communications peripheral or anywhere the main CPU can access. The debugger can also be used to load code directly to the CLA program RAM during development.
2. **Initialize CLA data RAM, if necessary**

   Populate the CLA data RAM with any required data coefficients or constants.
3. **Configure the CLA registers**

   Configure the CLA registers, but keep interrupts disabled until later (leave MIER = 0):
   - **Enable the CLA peripheral clock using the assigned PCLKCRn register**

     The peripheral clock control (PCLKCRn) registers are defined in the *System Control and Interrupts* chapter.
   - **Populate the CLA task interrupt vectors**
     - MVECT1 to MVECT8

       Each vector needs to be initialized with the start address of the task to be executed when the CLA receives the associated interrupt. This address is the full 16-bit starting address of the task in the lower 64K section of memory.
     - MVECT1 to MVECT7, and MVECTBGRND

       When using the background task, the vector (MVECTBGRND) must be loaded with the start address of the task in lower 64K of memory. Note that Task 8 is ignored when the background task is enabled.
   - **Select the task interrupt sources**

     For each task select the interrupt source in the CLA1TASKSRCSELx register. If a task is software triggered, select no interrupt. Since the background task takes the place of Task 8, the task uses the same peripheral trigger source as task 8.
   - **Enable IACK to start a task from software, if desired**

     To enable the IACK instruction to start a task set the MCTL[IACKE] bit. Using the IACK instruction avoids having to set and clear the EALLOW bit. If the background task is enabled, the IACK bit for task 8 is ignored; the user must, instead, write to the BGSTART bit of the MCTLBGRND register to start the background task (TRIGEN can be 0 to avoid a peripheral trigger from causing an overflow, for example, MSTSBGRND.BGOVF is set to 1).

- **Map CLA data RAM to CLA space, if necessary**

  Map the data RAM to the CLA space by first, assigning ownership of the memory block to the CLA by writing a 1 to the memory block's MemCfgRegs.LSxMSEL[MSEL_LSx] bit, and then specifying the memory block as a CLA data block by writing a 0 to the MemCfgRegs.LSxCLAPGM[CLAPGM_LSx] bit. When an LSx memory is configured as a CLA data memory, the CLA read/write accesses are arbitrated along with CPU accesses. The user has the option of turning on CPU fetch or write protection to the memory by writing to the appropriate bits of the MemCfgRegs.LSxACCPROTx registers.
- **Map CLA program RAM to CLA space**

  Map the CLA program RAM to CLA space by first assigning ownership of the memory block to the CLA by writing a 1 to the memory block's MemCfgRegs.LSxMSEL[MSEL_LSx] bit, and then specifying the memory block as CLA code memory by writing a 1 to the MemCfgRegs.LSxCLAPGM[CLAPGM_LSx] bit. When an LSx memory is configured as CLA program memory, only debug accesses are allowed on cycles in which the CLA is not fetching a new instruction.

4. **Initialize the PIE vector table and registers**

   When a CLA task completes, the associated interrupt in the PIE is flagged. The CLA overflow and underflow flags also have associated interrupts within the PIE.
5. **Enable CLA tasks/interrupts**

   Set appropriate bits in the interrupt enable register (MIER) to allow the CLA to service interrupts. Note that a CLA task only triggers on a level transition (a falling edge) of the configured interrupt source. If a peripheral is enabled and an interrupt fires before the CLA is configured, then the CLA does not recognize the interrupt edge and does not respond. To avoid this, configure the CLA before the peripherals or clear any pending peripheral interrupts before setting bits in the MIER register.
6. **Initialize other peripherals**

   Initialize any peripherals (such as ePWM, ADC, and others) that generate interrupt triggers for enabled CLA tasks.

   The CLA is now ready to service interrupts and the message RAMs can be used to pass data between the CPU and the CLA. Mapping of the CLA program and data RAMs typically occurs only during the initialization process. If the RAM mapping needs to be changed after initialization, the CLA interrupts must be disabled and all tasks must be completed (by checking the MIRUN register) prior to modifying the RAM ownership.

8.4.3 Debugging CLA Code

Debugging the CLA code is a simple process that occurs independently of the main CPU. The type 2 CLA adds a true software breakpoint feature.

8.4.3.1 Software Breakpoint Support (MDEBUGSTOP1)

The MDEBUGSTOP1 instruction is meant to be used as a software breakpoint; the instruction on which the execution must halt is replaced with this instruction.

The MDEBUGSTOP1 and MDEBUGSTOP instructions differ in how the CLA pipeline is treated. When halted, the MDEBUGSTOP1 instruction flushes all the instructions that have already been fetched; on a single-step or run-free command, the CLA refetches the same instruction that it replaced. Table 8-2 illustrates the pipeline behavior.

Table 8-2. Pipeline Behavior of the MDEBUGSTOP1 Instruction

Cycles	F1	F2	D1	D2	R1	R2	E	W	Comments
1	i1								
2	i2	i1							
3	i3	i2	i1						
4	i4	i3	i2	i1					
5	MDEBUG STOP1	i4	i3	i2	i1				
6	i6	MDEBUG STOP1	i4	i3	i2	i1			
7	i7	i6	MDEBUG STOP1	i4	i3	i2	i1		
9	i8	Flushed (MNOP)	Flushed (MNOP)	Flushed (MNOP)	i4	i3	i2	i1	CLA halted
10	i5(MDEBUGS TOP1)	Flushed (MNOP)	Flushed (MNOP)	Flushed (MNOP)	Flushed (MNOP)	i4	i3	i2	CLA step/run
11	i6	i5(MDEBUGS TOP1)	Flushed (MNOP)	Flushed (MNOP)	Flushed (MNOP)	Flushed (MNOP)	i4	i3	CLA step/run
12	i7	i6	i5(MDEBUGS TOP1)	Flushed (MNOP)	Flushed (MNOP)	Flushed (MNOP)	Flushed (MNOP)	i4	CLA step/run
13	i8	i7	i6	i5(MDEBUGS TOP1)	Flushed (MNOP)	Flushed (MNOP)	Flushed (MNOP)	Flushed (MNOP)	CLA step/run

A software breakpoint is placed at instruction i5. The instruction, i5, is then replaced with MDEBUGSTOP1. It takes 3 cycles for the MDEBUGSTOP1 to reach the D2 phase at which point the instructions i6, i7, and i8 that were previously fetched are now flushed from the pipeline. The instruction, i5, is then re-fetched and execution continues as before.

8.4.3.2 Legacy Breakpoint Support (MDEBUGSTOP)

1. **Insert a breakpoint in CLA code**

   Insert a CLA breakpoint (MDEBUGSTOP instruction) into the code where the CLA is to halt, then rebuild and reload the code. Because the CLA does not flush the pipeline when in single-step, the MDEBUGSTOP instruction must be inserted as part of the code. The debugger cannot insert the MDEBUGSTOP instruction as needed.

   If CLA breakpoints are not enabled, then the MDEBUGSTOP instruction is ignored and is treated as a MNOP. The MDEBUGSTOP instruction can be placed anywhere in the CLA code as long as the MDEBUGSTOP instruction is not within three instructions of a MBCNDD, MCCNDD, or MRCNDD instruction. When programming in C, the user can use the __mdebugstop() intrinsic instead; the compiler makes sure that the placement of the MDEBUSTOP instruction in the generated assembly does not violate any of the pipeline restrictions.
2. **Enable CLA breakpoints**

   Enable the CLA breakpoints in the debugger. In the Code Composer Studio™ IDE, this is done by connecting to the CLA core (or tap) from the debug perspective. Breakpoints are disabled when the core is disconnected.
3. **Start the task**

   There are three ways to start the task:
   a. The peripheral can assert an interrupt,
   b. The main CPU can execute an IACK instruction, or
   c. The user can manually write to the MIFRC register in the debugger window

   When the task starts, the CLA executes instructions until the MDEBUGSTOP is in the D2 phase of the pipeline. At this point, the CLA halts and the pipeline is frozen. The MPC register reflects the address of the MDEBUGSTOP instruction.

4. **Single-step the CLA code**

   Once halted, the user can single-step the CLA code. The behavior of a CLA single-step is different than the main C28x. When issuing a CLA single-step, the pipeline is clocked only one cycle and then again frozen. On the C28x CPU, the pipeline is flushed for each single-step.

   Run to the next MDEBUGSTOP or to the end of the task. If another task is pending, the task automatically starts when run to the end of the task.

   Note

   A CLA fetch has higher priority than CPU debug reads. For this reason, it is possible for the CLA to permanently block CPU debug accesses if the CLA is executing in a loop. This can occur when initially developing CLA code due to a bug that causes an infinite loop. To avoid locking up the main CPU, the program memory returns all 0x0000 for CPU debug reads when the CLA is running. When the CLA is halted or idle, then normal CPU debug read and write access to CLA program memory can be performed.

   If the CLA gets caught in an infinite loop, use a soft or hard reset to exit the condition. A debugger reset also exits the condition.

   There are special cases that can occur when single-stepping a task such that the program counter, MPC, reaches the MSTOP instruction at the end of the task.

   - **MPC halts at or after the MSTOP with a task already pending**

     If single-stepping or halted in "task A" and "task B" comes in before the MPC reaches the MSTOP, then "task B" starts if continuing to step through the MSTOP instruction. Basically, if "task B" is pending before the MPC reaches MSTOP in "task A" then there is no issue in "task B" starting and no special action is required.
   - **MPC halts at or after the MSTOP with no task pending**

     In this case, if single-stepped or halted in "task A" and the MPC has reached the MSTOP with no tasks pending. If "task B" comes in at this point, "task B" is flagged in the MIFR register but "task B" can or cannot start if continuing to single-step through the MSTOP instruction of "task A."

     Depending on exactly when the new task comes in, to reliably start "task B", perform a soft reset and reconfigure the MIER bits. Once this is done, start single-stepping "task B."

     This case can be handled slightly differently if there is control over when "task B" comes in (for example, using the IACK instruction to start the task). In this case, the task is single-stepped or halted in "task A" and the MPC has reached the MSTOP with no tasks pending. Before forcing "task B," run free to force the CLA out of the debug state. Once this is done, force "task B" and continue debugging.

5. **Disable CLA breakpoints, if desired**

   In the Code Composer Studio™ IDE, disable the CLA breakpoints by disconnecting the CLA core in the debug perspective. Make sure to first issue a run or reset; otherwise, the CLA is halted and no other tasks start.

8.4.4 CLA Illegal Opcode Behavior

If the CLA fetches an opcode that does not correspond to a legal instruction, it will behave as follows:

- The CLA will halt with the illegal opcode in the D2 phase of the pipeline as if it were a breakpoint. This will occur whether CLA breakpoints are enabled or not.
- The CLA will issue the task-specific interrupt to the PIE.
- The MIRUN bit for the task will remain set.

Further single-stepping is ignored once execution halts due to an illegal op-code. To exit this situation, issue either a soft or hard reset of the CLA as described in Section 8.4.5.

8.4.5 Resetting the CLA

There are times when resetting the CLA is needed. For example, during code debug the CLA can enter an infinite loop due to a code bug. The CLA has two types of resets: hard and soft. Both of these resets can be performed by the debugger or by the main CPU.

- **Hard Reset** Writing a 1 to the MCTL[HARDRESET] bit performs a hard reset of the CLA. The behavior of a hard reset is the same as a system reset (using $\overline{\text{XRS}}$ or the debugger). In this case, all CLA configuration and execution registers can be set to their default state and CLA execution halts.
- **Soft Reset** Writing a 1 to the MCTL[SOFTRESET] bit performs a soft reset of the CLA. If a task is executing, the task halts and the associated MIRUN bit is cleared. All bits within the interrupt enable (MIER) register are also cleared, so that no new tasks start. In addition, the background task's start bit (MCTLBGRN.BGSTART) and trigger enable bit (MCTLBGRND.TRIGEN) are reset. The MVECTBGRNACTIVE is set to the value of MVECTBGRND, and the status register (MSTSBGRND) is also reset.

8.5 Pipeline

This section describes the CLA pipeline stages and presents cases where pipeline alignment must be considered.

8.5.1 Pipeline Overview

The CLA pipeline is very similar to the C28x pipeline with eight stages:

1. **Fetch 1 (F1):** During the F1 stage the program read address is placed on the CLA program address bus.
2. **Fetch 2 (F2):** During the F2 stage the instruction is read using the CLA program data bus.
3. **Decode 1 (D1):** During D1 the instruction is decoded.
4. **Decode 2 (D2):** Generate the data read address. Changes to MAR0 and MAR1 due to post-increment using indirect addressing takes place in the D2 phase. Conditional branch decisions are also made at this stage based on the MSTF register flags.
5. **Read 1 (R1):** Place the data read address on the CLA data-read address bus. If a memory conflict exists, the R1 stage will be stalled.
6. **Read 2 (R2):** Read the data value using the CLA data read data bus.
7. **Execute (EXE):** Execute the operation. Changes to MAR0 and MAR1 due to loading an immediate value or value from memory take place in this stage.
8. **Write (W):** Place the write address and write data on the CLA write data bus. If a memory conflict exists, the W stage will be stalled.

8.5.2 CLA Pipeline Alignment

The majority of the CLA instructions do not require any special pipeline considerations. This section lists the few operations that do require special consideration.

- **Write Followed by Read**

  In both the C28x pipeline and the CLA pipeline, the read operation occurs before the write. This means that if a read operation immediately follows a write, then the read completes first as shown in Table 8-3. In most cases this does not cause a problem since the contents of one memory location does not depend on the state of another. For accesses to peripherals where a write to one location can affect the value in another location, the code must wait for the write to complete before issuing the read as shown in Table 8-4.

  This behavior is different for the C28x CPU. For the C28x CPU, any write followed by read to the same location is protected by what is called write-followed-by-read protection. This protection automatically stalls the pipeline so that the write completes before the read. In addition, some peripheral frames are protected such that a C28x CPU write to one location within the frame always completes before a read to the frame. The CLA does not have this protection mechanism. Instead, the code must wait to perform the read.

Table 8-3. Write Followed by Read - Read Occurs First

Instruction	F1	F2	D1	D2	R1	R2	E	W
I1 MMOV16 @Reg1, MR3	I1							
I2 MMOV16 MR2, @Reg2	I2	I1						
		I2	I1					
			I2	I1				
				I2	I1			
					I2	I1		
						I2	I1	
							I2	I1

Table 8-4. Write Followed by Read - Write Occurs First

Instruction	F1	F2	D1	D2	R1	R2	E	W
I1 MMOV16 @Reg1, MR3	I1							
I2	I2	I1						
I3	I3	I2	I1					
I4	I4	I3	I2	I1				
I5 MMOV16 MR2, @Reg2	I5	I4	I3	I2	I1			
		I5	I4	I3	I2	I1		
			I5	I4	I3	I2	I1	
				I5	I4	I3	I2	I1
					I5	I4	I3	
						I5	I4	
							I5	

- **Delayed Conditional instructions: MBCNDD, MCCNDD, and MRCNDD**

  Referring to Example 8-1, the following applies to delayed conditional instructions:
  - **I1:** I1 is the last instruction that can effect the CNDF flags for the branch, call, or return instruction. The CNDF flags are tested in the D2 phase of the pipeline. That is, a decision is made whether to branch or not when MBCNDD, MCCNDD, or MRCNDD is in the D2 phase.
  - **I2, I3, and I4:** The three instructions preceding MBCNDD can change the MSTF flags but have no effect on whether the MBCNDD instruction branches or not. This is because the flag modification occurs after the D2 phase of the branch, call, or return instruction. These three instructions must not be a MSTOP, MDEBUGSTOP, MBCNDD, MCCNDD, or MRCNDD.
  - **I5, I6, and I7:** The three instructions following a branch, call, or return are always executed irrespective of whether the condition is true or not. These instructions must not be MSTOP, MDEBUGSTOP, MBCNDD, MCCNDD, or MRCNDD.

  For a more detailed description, refer to the description for MBCNDD, MCCNDD, and MRCNDD.

Example 8-1. Code Fragment For MBCNDD, MCCNDD, or MRCNDD

```
  <Instruction 1>    ; I1 Last instruction that can affect flags for
                     ;     the branch, call or return operation
  <Instruction 2>    ; I2 Cannot be stop, branch, call or return
  <Instruction 3>    ; I3 Cannot be stop, branch, call or return
  <Instruction 4>    ; I4 Cannot be stop, branch, call or return
  <branch/call/ret>  ; MBCNDD, MCCNDD or MRCNDD
                     ; I5-I7: Three instructions after are always
                     ; executed whether the branch/call or return is
                     ; taken or not
  <Instruction 5>    ; I5 Cannot be stop, branch, call or return
  <Instruction 6>    ; I6 Cannot be stop, branch, call or return
  <Instruction 7>    ; I7 Cannot be stop, branch, call or return
  <Instruction 8>    ; I8
  <Instruction 9>    ; I9
  ....
```

- **Stop or Halting a Task: MSTOP and MDEBUGSTOP**

  The MSTOP and MDEBUGSTOP instructions cannot be placed three instructions before or after a conditional branch, call or return instruction (MBCNDD, MCCNDD, or MRCNDD). Refer to Example 8-1. To single-step through a branch/call or return, insert the MDEBUGSTOP at least four instructions back and step from there.

- **Loading MAR0 or MAR1**
  A load of auxiliary register MAR0 or MAR1 occurs in the EXE phase of the pipeline. Any post increment of MAR0 or MAR1 using indirect addressing occurs in the D2 phase of the pipeline. Referring to Example 8-2, the following applies when loading the auxiliary registers:
  – **I1 and I2**

    The two instructions following the load instruction use the value in MAR0 or MAR1 before the update occurs.
  – **I3**

    Loading of an auxiliary register occurs in the EXE phase while updates due to post-increment addressing occur in the D2 phase. Thus I3 cannot use the auxiliary register or there is a conflict. In the case of a conflict, the update due to address-mode post increment wins and the auxiliary register is not updated with #_X.
  – **I4**

    Starting with the 4th instruction MAR0 or MAR1 has the new value.

Example 8-2. Code Fragment for Loading MAR0 or MAR1

```
; Assume MAR0 is 50 and #_X is 20

   MMOVI16 MAR0, #_X ; Load MAR0 with address of X (20)
   <Instruction 1>   ; I1 Will use the old value of MAR0 (50)
   <Instruction 2>   ; I2 Will use the old value of MAR0 (50)
   <Instruction 3>   ; I3 Cannot use MAR0
   <Instruction 4>   ; I4 Will use the new value of MAR0 (20)
   <Instruction 5>   ; I5 Will use the new value of MAR0 (20
   ....
```

- **Background Task Interrupted Close to a Branch**

  When the background task is running, if another task request happens (and MSTSBGRND.BGINTM is not set), then the following sequence of operations happen.

  – A check is made to determine if the following instructions are not in the pipeline (D2 – R2).
    - MBCNDD
    - MCCNDD
    - MRCNDD

  If any of the above instructions are present in the pipeline, the back ground task continues to execute until such time when the condition is satisfied. Once the condition is satisfied, the following actions are performed:
  – The MPC value of the D1 phase instruction is saved to the MVECTBGRNDACTIVE register.
  – The run status bit of the background task (MSTSBGRND.RUNSTS) is cleared.
  – An MSTOP instruction is forced into the D2 phase of the pipeline; causing the background task to terminate.

  When the background task terminates, the CLA picks the next highest pending task and begins execution. Note that while the background task is pending, the background task has the lowest priority and, therefore, yields to any other pending task. Once all pending non-background tasks have completed execution, the CLA restores the program counter (MPC), that is, loads the address from the MVECTBGRNDACTIVE register to the MPC, sets the background status to RUN (MSTSBGRND.RUN = 1), and continues execution from that point.

- **MSTOP in the Background Task**
  If an MSTOP instruction occurs in the D1 phase while the background task is running, the following sequence of operations happens:
  - The RUN bit (MSTSBGRND.RUN) is cleared.
  - The address stored in MVECTBGRND is copied over to MVECTBGRNDACTIVE.
  - An interrupt, signaling the background task has completed execution, is generated. This interrupt signal is ANDed with the interrupt from Task 8 and fed to the PIE. Note that if an illegal instruction occurs inside the background task, the interrupt for task 8 is fired.
  - When the background task terminates, the CLA resumes arbitration among the pending tasks.

8.5.2.1 ADC Early Interrupt to CLA Response

The ADC can be configured to generate an early interrupt pulse before the ADC conversion completes. If this option is used to start a CLA task, the CLA is able to read the result as soon as the conversion result is available in the ADC result register. This combination of just-in-time sampling along with the low interrupt response of the CLA enable faster system response and higher frequency control loops. The CLA task trigger to first instruction fetch interrupt latency is 4 cycles.

Timings for ADC conversions are shown in the timing diagrams of the ADC chapter. If the ADCCLK is a divided down version of the SYSCLK, the user has to account for the conversion time in SYSCLK cycles.

For example, if using the 12-bit ADC with ADCCLK at SYSCLK / 4, the ADC can take 10.5 ADCCLK x 4 SYSCLK = 42 SYSCLK cycles to complete a conversion. If using the ADC in 16-bit mode at the same ADCCLK, the ADC can take 29.5 ADCCLK x 4 SYSCLK = 118 SYSCLK cycles, and so on.

From a CLA perspective, the pipeline activity is shown in Table 8-5 for an N-cycle (SYSCLK) ADC conversion. The N-2 instruction arrives in the R2 phase just in time to read the result register. While the prior instructions enter the R2 phase of the pipeline too soon to read the conversion, the instructions can be efficiently used for pre-processing calculations needed by the task.

Table 8-5. ADC to CLA Early Interrupt Response

ADC Activity	CLA Activity	F1	F2	D1	D2	R1	R2	E	W
Sample									
Sample									
...									
Sample									
$Conversion_{(Cycle\ 1)}$	Interrupt Received								
$Conversion_{(Cycle\ 2)}$	Task Startup								
$Conversion_{(Cycle\ 3)}$	Task Startup								
$Conversion_{(Cycle\ 4)}$	$I_{(Cycle\ 4)}$	$I_{(Cycle\ 4)}$							
$Conversion_{(Cycle\ 5)}$	$I_{(Cycle\ 5)}$	$I_{(Cycle\ 5)}$	$I_{(Cycle\ 4)}$						
$Conversion_{(...)}$	...	...	...	...	...	...	...		
$Conversion_{(Cycle\ N-6)}$	$I_{(Cycle\ N-6)}$	$I_{(Cycle\ N-6)}$	$I_{(Cycle\ N-7)}$	$I_{(Cycle\ N-8)}$	$I_{(Cycle\ N-9)}$	$I_{(Cycle\ N-10)}$	$I_{(Cycle\ N-11)}$		
$Conversion_{(Cycle\ N-5)}$	$I_{(Cycle\ N-5)}$	$I_{(Cycle\ N-5)}$	$I_{(Cycle\ N-6)}$	$I_{(Cycle\ N-7)}$	$I_{(Cycle\ N-8)}$	$I_{(Cycle\ N-9)}$	$I_{(Cycle\ N-10)}$		
$Conversion_{(Cycle\ N-4)}$	$I_{(Cycle\ N-4)}$	$I_{(Cycle\ N-4)}$	$I_{(Cycle\ N-5)}$	$I_{(Cycle\ N-6)}$	$I_{(Cycle\ N-7)}$	$I_{(Cycle\ N-8)}$	$I_{(Cycle\ N-9)}$		
$Conversion_{(Cycle\ N-3)}$	$I_{(Cycle\ N-3)}$	$I_{(Cycle\ N-3)}$	$I_{(Cycle\ N-4)}$	$I_{(Cycle\ N-5)}$	$I_{(Cycle\ N-6)}$	$I_{(Cycle\ N-7)}$	$I_{(Cycle\ N-8)}$		
$Conversion_{(Cycle\ N-2)}$	**Read RESULT**	**Read RESULT**	$I_{(Cycle\ N-3)}$	$I_{(Cycle\ N-4)}$	$I_{(Cycle\ N-5)}$	$I_{(Cycle\ N-6)}$	$I_{(Cycle\ N-7)}$		
$Conversion_{(Cycle\ N-1)}$			**Read RESULT**	$I_{(Cycle\ N-3)}$	$I_{(Cycle\ N-4)}$	$I_{(Cycle\ N-5)}$	$I_{(Cycle\ N-6)}$		
$Conversion_{(Cycle\ N-0)}$				**Read RESULT**	$I_{(Cycle\ N-3)}$	$I_{(Cycle\ N-4)}$	$I_{(Cycle\ N-5)}$		
Conversion Complete					**Read RESULT**	$I_{(Cycle\ N-3)}$	$I_{(Cycle\ N-4)}$		
RESULT Latched						**Read RESULT**	$I_{(Cycle\ N-3)}$		
RESULT Available							**Read RESULT**		

The ADCINTCYCLE register of the ADC can be programmed by the application to adjust the generation of the interrupt pulse to align with the ADC read operation. For example, if the first instruction in the task reads the ADC and the conversion time is N SYSCLK cycles, then the delay programmed is (N-2) - 4 = N-6.

8.5.3 Parallel Instructions

Parallel instructions are single opcodes that perform two operations in parallel. The following types of parallel instructions are available: math operation in parallel with a move operation, or two math operations in parallel. Both operations complete in a single cycle and there are no special pipeline alignment requirements.

Example 8-3. Math Operation with Parallel Load

```
;  MADDF32 || MMOV32 instruction: 32-bit floating-point add with parallel move
;  MADDF32 is a 1 cycle operation
;  MMOV32 is a 1 cycle operation
         MADDF32    MR0,  MR1, #2     ; MR0 = MR1 + 2,
      || MMOV32   MR1,  @Val           ; MR1 gets the contents of Val
                                       ; <-- MMOV32 completes here (MR1 is valid)
                                       ; <-- DDF32 completes here (MR0 is valid)
         MMPYF32 MR0, MR0, MR1         ; Any instruction, can use MR1 and/or MR0
```

Example 8-4. Multiply with Parallel Add

```
;  MMPYF32 || MADDF32 instruction: 32-bit floating-point multiply with parallel add
;  MMPYF32 is a 1 cycle operation
;  MADDF32 is a 1 cycle operation
     MMPYF32 MR0, MR1, MR3         ; MR0 = MR1 * MR3
 ||  MADDF32 MR1, MR2, MR0         ; MR1 = MR2 + MR0 (Uses value of MR0 before MMPYF32)
                                   ; <-- MMPYF32 and MADDF32 complete here (MR0 and MR1 are valid)
     MMPYF32 MR1, MR1, MR0         ; Any instruction, can use MR1 and/or MR0
```

8.5.4 CLA Task Execution Latency

The CLA task execution latency depends on the state of the system:

- CLA task trigger of new task (normal or background) without background task active:

  Task takes 8 cycles from CLA task trigger to first instruction of task to reach the D2 phase of pipeline.

  Note

  If background task has been configured in the system, then the compiler during code compilation adds context save instructions at the start of each regular task and restore instructions at end of each task so that register content can be saved and restored in case a background task is executing while the regular task is triggered. When a regular task is entered, this compiler-generated context save instruction is the first instruction of the task.

- CLA task trigger of normal task when background task is active:

  Task takes 9 cycles from CLA task trigger to first instruction of normal task to reach the D2 phase of pipeline. There is a difference of one clock cycle to force the MSTOP in the D2 phase of the background task before the task exits as compared to a new task trigger without the background task active.

  Note

  If the MBCNDD/MCCNDD/MRCNDD instructions in the background task are in the D2 phase of the pipeline when a new task gets triggered, the task takes a minimum of 3 more cycles to complete these uninterruptible instructions adding to the delay.

- Returning to background task from normal task:

  The task takes 5 cycles to return from a normal task to resume the background task instruction at the D2 phase of the pipeline.

8.6 Software

8.6.1 CLA Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/cla

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

8.6.1.1 CLA arcsine(x) using a lookup table (cla_asin_cpu01) - C28X_DUAL

FILE: cla_ex1_asin_cpu1.c

In this example, cpu1 will be used to initialize the clocks Task 1 of the CLA on cpu2 will calculate the arcsine of an input argument in the range (-1.0 to 1.0) using a lookup table. It is recommended to run the c28x1 core first, followed by the C28x2 core.

Memory Allocation

- CLA1 Math Tables (RAMLS0)
  - CLAasinTable - Lookup table
- CLA1 to CPU Message RAM
  - fResult - Result of the lookup algorithm
- CPU to CLA1 Message RAM
  - fVal - Sample input to the lookup algorithm

Watch Variables

- fVal - Argument to task 1
- fResult - Result of arcsin(fVal)

8.6.1.2 CLA Arcsine Example. - C28X_DUAL

FILE: cla_ex1_asin_cpu2.c Dual Core arcsine example. This example demonstrates how to run CLA tasks on cpu2.cla1 It is recommended to run the c28x1 core first, followed by the C28x2 core.

8.6.1.3 CLA arcsine(x) using a lookup table (cla_asin_cpu01)

FILE: cla_ex1_asin.c

In this example, Task 1 of the CLA will calculate the arcsine of an input argument in the range (-1.0 to 1.0) using a lookup table.

Note that since this example does not use background CLA task, the compile flag cla_background_task is turned off for this project. Set this flag as on to enable background CLA task. The option is available in Project Properties -> C2000 Build -> C2000 Compiler -> Advanced Options -> Runtime Model Options.

Memory Allocation

- CLA1 Math Tables (RAMLS0)
- CLA1 to CPU Message RAM
  - fResult - Result of the lookup algorithm
- CPU to CLA1 Message RAM
  - fVal - Sample input to the lookup algorithm

Watch Variables

- fVal - Argument to task 1
- fResult - Result of arcsin(fVal)

8.6.1.4 CLA 2 Pole 2 Zero Infinite Impulse Response Filter (cla_iir2p2z_cpu01) - C28X_DUAL

FILE: cla_ex2_iir2p2z_cpu1.c

This example implements a Transposed Direct Form II IIR filter, commonly known as a Biquad. The input vector is a software simulated noisy signal that is fed to the biquad one sample at a time, filtered and then stored in an output buffer for storage. It is recommended to run the c28x1 core first, followed by the C28x2 core.

Memory Allocation

- CLA1 Data RAM 1 (RAML2)
  - S1_A - Feedback coefficients
  - S1_B - Feedforward coefficients
- CLA1 to CPU Message RAM
  - yn - Output of the Biquad
- CPU to CLA1 Message RAM
  - xn - Sample input to the filter

Watch Variables

- fBiquadOutput
- pass
- fail

8.6.1.5 CLA 2-pole 2-zero IIR Filter Example for F2837xD. - C28X_DUAL

FILE: cla_ex2_iir2p2z_cpu2.c Dual Core iir2p2z example. This example demonstrates how to run CLA tasks on cpu2.cla1 It is recommended to run the c28x1 core first, followed by the C28x2 core.

8.6.1.6 CLA arctangent(x) using a lookup table (cla_atan_cpu01)

FILE: cla_ex2_atan.c

In this example, Task 1 of the CLA will calculate the arctangent of an input argument using a lookup table.

Note that since this example does not use background CLA task, the compile flag cla_background_task is turned off for this project. Set this flag as on to enable background CLA task. The option is available in Project Properties -> C2000 Build -> C2000 Compiler -> Advanced Options -> Runtime Model Options.

Memory Allocation

- CLA1 Math Tables (RAMLS0)
- CLA1 to CPU Message RAM
  - fResult - Result of the lookup algorithm
- CPU to CLA1 Message RAM
  - fNum - Numerator of sample input
  - fDen - Denominator of sample input

Watch Variables

- fVal - Argument to task 1
- fResult - Result of arctan(fVal)

8.6.1.7 CLA background nesting task

FILE: cla_ex3_background_nesting_task.c

This example configures CLA task 1 to be triggered by EPWM1 running at 2 Hz (period = 0.5s). A background task is configured to be triggered by CPU timer running at .5 Hz (period = 2s). CLA task 1 toggles LED1 at the start and end of the task and the background task toggles LED2 at the start and end of the task. Background task will be preempted by Task1 and hence LED1 will be toggling even while LED2 is ON.

Note that the compile flag cla_background_task is turned on in this project. Enabling background task adds additional context save/restore cycles during task switching thus increasing the overall trigger-to-task latency. If the application does not use the background CLA task, it is recommended to turn this flag off for better performance. The option is available in Project Properties -> C2000 Build -> C2000 Compiler -> Advanced Options -> Runtime Model Options.

External Connections

- None

Watch Variables

- None

8.6.1.8 Controlling PWM output using CLA

FILE: cla_ex4_pwm_control.c

This example showcases how to update PWM signal output using CLA. EPWM1 is configured to generate complementary signals on both of its channels of fixed frequency 100 KHz. EPWM4 is configured to trigger a periodic CLA control task of frequency 10 KHz. The CLA task implements a very simple logic to vary the duty of the EPWM1 outputs by increasing it by 0.1 in every iteration and maintaining it in the range of 0.1-0.9. For actual use-cases, the control logic could be modified to much more complex depending upon the application. The other CLA task (CLA task 8) is triggered by software at beginning to initialize the CLA global variables

External Connections

- Observe GPIO0 (EPWM1A) on oscilloscope
- Observe GPIO1 (EPWM1B) on oscilloscope

Watch Variables

- duty

8.6.1.9 Just-in-time ADC sampling with CLA

FILE: cla_ex5_adc_just_in_time.c

This example showcases how to utilize early-interrupt feature of ADC in combination with the low interrupt response of CLA to enable faster system response and achieve high frequency control loops. EPWM1 is configured to generate a PWM output signal of frequency 1 MHz and this is also used to trigger the ADC sampling at each cycle. ADCA is configured to sample the input on Channel 0 and to generate the early interrupt at the end of S/H + offset cycles. This interrupt is used to trigger the CLA control task. The CLA task implements the control logic to update the duty of the PWM output based on reading the ADC sample data just-in-time i.e. as soon as the ADC results gets latched.The early interrupt feature and low interrupt latency of CLA allows to do some pre-processing as well before reading the ADC data and still completes updating the PWM output before the next interrupts comes in i.e. data read and PWM update is done within a 1 MHz cycle. For illustration purposes, 3-point moving average filter is used to simulate some processing and few steps of the filtering code are done before reading the ADC result which we consider as pre-processing code. The ADC interrupt offset is programmed based on the cycles consumed by the pre-processing code.

The calculation for interrupt offset value is as follows :- -ADC acquisition cycles programmed = 10 SYSCLKS -Conversion time for 12-bit data = 10.5 ADCCLKS = N = 42 SYSCLKS -CLA task trigger to first instruction in Fetch delay = 4 -Let the interrupt offset value be 'x' -The code inside CLA control task before ADC read takes below cycles : Setting up profiling gpio : 3 cycles Pre-processing : 13 cycles Total = 3 + 13 = 16 cycles

As described in device TRM, in order to read just-in-time the total delay before reading ADC should be (N-2) cycles = 40 i.e. : x + 4 + 16 = 40 : x = 20

NOTE :- The optimization is off for this project and the cycles quoted above corresponds to that case.

GPIO2 is used for profiling purposes. GPIO2 is set at the beginning of CLA task 1 and is reset at the end of the task. Thus ON time of GPIO2 indicates the CLA activity. In order to validate the example functionality , observe the GPIO0 (PWM output) and GPIO2 (profiling GPIO) on CRO. The cycles difference between the rising edge of the GPIO0 and GPIO2 indicate the total delay from the time of ADC trigger to setting up of profiling GPIO inside CLA task which should be around 44 cycles (220 ns) based on the above calculation.

External Connections

- Provide constant DC input on ADCA0 for quick validation. GND -> Should observe PWM output duty = 0.1 3.3V -> Should observe PWM output duty = 0.9 Can also provide analog input in range 0 - 3.3V upto fs / 10 = 100 KHz for observing continuous duty variations
- Observe GPIO0 on oscilloscope
- Observe GPIO2 on oscilloscope

Watch Variables

- None

8.6.1.10 Optimal offloading of control algorithms to CLA

FILE: cla_ex6_cpu_offloading.c

This example showcases how to optimally offload the control algorithms from CPU to CLA in order to meet the system requirements. In this example, two control loops are simulated, the faster one (loop1) running at 200 KHz and the slower one (loop2) running at 20 KHz. Loop1 senses the first parameter at ADCA Channel 0, runs the PI controller to achieve the target and contributes to the duty of EPWM1A output with 80% weightage. Loop2 senses the second parameter at ADCB Channel 2, runs the PI controller and contributes to the duty of EPWM1A output with 20% weightage. It is important to note that since these are just software simulated control loops but there is no actual physical process involved and hence updating the duty is not going to have any affect on sampled inputs. ADCA is configured to oversample the first parameter using SOCs 0-3 to suppress the noise and similarly ADCB is used to oversample the second parameter. EPWM4 and EPWM5 are configured to trigger the ADCA and ADCB sampling at loop1 and loop2 frequencies respectively. Once the conversion of all 4 SOCs complete, a CPU ISR or a CLA task is triggered based on the user-configuration. There is also a background task running in the main loop which disables the entire system including PWM output and the control loops when "system_OFF" is set to 1. The system gets enabled again once "system_OFF" is restored back to 0. By default system_OFF is set to 0 but it's value can be updated dynamically by adding it to expression window and writing to it. DCL library is included in the project to make use of optimal PI controllers used in both the loops. User-configurable pre-defined symbol "run_loop1_cla" has been added to the project options in order to specify whether to run the loop1 on C28x or CLA. GPIO2 and GPIO3 are used to profile the execution of loop1 and loop2.

For run_loop1_cla == 0 i.e. both loops running on CPU -> Loop1 Utilization = ~77.5% (measured using profiling GPIO2) -> Loop2 Utilization = ~6% (measured using profiling GPIO3) -> Background task in a while loop -> Total CPU utilization is greater than Utilization bound (UB) Hence the system is non-schedulable, lower priority task (Loop2) execution never completes (no toggling observed on GPIO3) and also background task never gets chance to execute

For run_loop1_cla == 1 i.e. high frequency control loop (loop1) is offloaded to CLA while loop2 runs on CPU -> Loop1 Utilization (CLA) = ~73% -> Loop2 Utilization (CPU)= ~6% -> Total CPU utilization has come down to just ~6% Hence the system is perfectly schedulable, no miss happens for any of the loops and offloading of loop1 to CLA saves CPU bandwidth to execute background tasks as well

For quick inspection of the example functionality, constant DC HIGH/LOW inputs can be provided to the analog channels instead of varying analog voltages. The target value for both the loops are set as some intermediate value i.e. 3500 corresponds to ~2.8V. Now since the sensed inputs are constant and not same as target so the controller outputs will get saturated soon to either 1 or 0. Thus the "duty" variable can take only fixed values based on the equations used in the loops. Infact the duty output would be very intutive, for instance if both inputs are LOW(GND), the controller will try to produce the maximum duty as the target is higher than sensed value hence the duty should be 1.0(0.2 + 0.8) but will get saturated to 0.9(the maximum value defined). Similarly if both inputs are made HIGH, the duty will be 0.1 (the minimum saturation value defined). The final duty table is shown below :

External Connections

- Observe GPIO2 (Loop1 Profiling) on oscilloscope
- Observe GPIO3 (Loop2 Profiling) on oscilloscope
- Observe GPIO0 (EPWM1A Output) on oscilloscope
- Provide constant HIGH(3.3V)/LOW(0V) on both ADCA Ch0 and ADCB Ch2 for quick validation, the following duty value should be observable at EPWM1A for various combinations if the system is perfectly schedulable i.e. both loops gets chance to execute properly :- A0 B2 duty GND GND 0.9 3.3V GND 0.2 GND 3.3V 0.8 3.3V 3.3V 0.1 Note :- The optimization is OFF for this project and all the profiling data quoted above corresponds to this case.

8.6.1.11 Handling shared resources across C28x and CLA

FILE: cla_ex7_shared_resource_handling.c

This example showcases how to handle shared resource challenges across C28x and CLA. As the peripherals are shared between CLA and the CPU, overlapping read-modify-write to the registers by them can lead to data race conditions ultimately leading to data violation or incorrect functionality. In this example, CPU ISR and CLA tasks runs independently. CPU ISR gets triggered by EPWM4 @10KHz and toggles the EPWM1B output via software by controlling CSFB bits of AQCSFRC. CLA task gets triggered by EPWM5 @100Khz and toggles the EPWM1A output via software by controlling CSFA bits of AQCSFRC. Thus in this process both CPU and CLA do read-modify -write to AQCSFRC register independently at different frequencies so there is chance of race condition and updates due to one of them can get lost/. overwritten. This can be clearly observed by updating "phase_shift_ON" to 0U and probing the EPWM1A and 1B outputs on a scope.

This is a standard critical section problem and can be handled by software handshaking mechanism like mutex etc. But most of the real-time control applications are time-sensitive and cannot afford addition software overhead hence this example suggests an alternative hardware based technique to avoid shared resource conflicts between CPU and CLA. The phase shifting mechanism of the EPWM modules is utilized to schedule the CLA task and CPU ISR as desired. EPWM4 generates a synchronous pulse every ZERO event and provides a phase shift of 20 cycles to EPWM5. This way both CLA task and C28x ISR runs at original frequencies i.e. 100KHz and 10KHz but CLA task leads with a phase offset of 20 cycles wrt CPU ISR. Hence concurrent read-modify-writes to AQCSFRC never happens and the EPWM1A and EPWM1B outputs behave as desired i.e. consistent 50 KHz PWM output on EPWM1A and 5 KHz PWM output on EPWM1B with a duty ~50% on both should be generated. In order to utilize this phase shifting mechanism in this example, please make sure "phase_shift_ON" is set to 1.

External Connections

- Observe GPIO0 (EPWM1A Output) on oscilloscope
- Observe GPIO1 (EPWM1B Output) on oscilloscope
- Observe GPIO2 (CLA Task Profiling) on oscilloscope
- Observe GPIO3 (CPU ISR Profiling) on oscilloscope

Note :- The phase offset value can easily be configured by updating TBPHS register to schedule the CLA task and C28x ISR as desired depending upon the application need so as to avoid overlapping register writes by CPU and CLA

Note :- The optimization is on and set to O2 for the project and all the results quoted correspond to this case.

8.7 Instruction Set

This section describes the assembly language instructions of the control law accelerator. Also described are parallel operations, conditional operations, resource constraints, and addressing modes. The instructions listed here are independent from C28x and C28x+FPU instruction sets.

8.7.1 Instruction Descriptions

This section gives detailed information on the instruction set. Each instruction may present the following information:

- Operands
- Opcode
- Description
- Exceptions
- Pipeline
- Examples
- See also

The example INSTRUCTION is shown to familiarize you with the way each instruction is described. The example describes the kind of information you will find in each part of the individual instruction description and where to obtain more information. CLA instructions follow the same format as the C28x; the source operand(s) are always on the right and the destination operand(s) are on the left.

The explanations for the syntax of the operands used in the instruction descriptions for the C28x CLA are given in Table 8-6.

Table 8-6. Operand Nomenclature

Symbol	Description
#16FHi	16-bit immediate (hex or float) value that represents the upper 16-bits of an IEEE 32-bit floating-point value. Lower 16-bits of the mantissa are assumed to be zero.
#16FHiHex	16-bit immediate hex value that represents the upper 16-bits of an IEEE 32-bit floating-point value. Lower 16-bits of the mantissa are assumed to be zero.
#16FLoHex	A 16-bit immediate hex value that represents the lower 16-bits of an IEEE 32-bit floating-point value
#32Fhex	32-bit immediate value that represents an IEEE 32-bit floating-point value
#32F	Immediate float value represented in floating-point representation
#0.0	Immediate zero
#SHIFT	Immediate value of 1 to 32 used for arithmetic and logical shifts.
addr	Opcode field indicating the addressing mode
CNDF	Condition to test the flags in the MSTF register
FLAG	Selected flags from MSTF register (OR) 8 bit mask indicating which floating-point status flags to change
MAR0	auxiliary register 0
MAR1	auxiliary register 1
MARx	Either MAR0 or MAR1
mem16	16-bit memory location accessed using direct, indirect, or offset addressing modes
mem32	32-bit memory location accessed using direct, indirect, or offset addressing modes
MRa	MR0 to MR3 registers
MRb	MR0 to MR3 registers
MRc	MR0 to MR3 registers
MRd	MR0 to MR3 registers
MRe	MR0 to MR3 registers
MRf	MR0 to MR3 registers
MSTF	CLA Floating-point Status Register
shift	Opcode field indicating the number of bits to shift.

Table 8-6. Operand Nomenclature (continued)

Symbol	Description
VALUE	Flag value of 0 or 1 for selected flag (OR) 8 bit mask indicating the flag value; 0 or 1

Each instruction has a table that gives a list of the operands and a short description. Instructions always have their destination operand(s) first followed by the source operand(s).

Table 8-7. INSTRUCTION dest, source1, source2 Short Description

	Description
dest1	Description for the 1st operand for the instruction
source1	Description for the 2nd operand for the instruction
source2	Description for the 3rd operand for the instruction
Opcode	This section shows the opcode for the instruction
Description	Detailed description of the instruction execution is described. Any constraints on the operands imposed by the processor or the assembler are discussed.
Restrictions	Any constraints on the operands or use of the instruction imposed by the processor are discussed.
Pipeline	This section describes the instruction in terms of pipeline cycles as described in Section 8.5
Example	Examples of instruction execution. If applicable, register and memory values are given before and after instruction execution. Some examples are code fragments while other examples are full tasks that assume the CLA is correctly configured and the main CPU has passed it data.
Operands	Each instruction has a table that gives a list of the operands and a short description. Instructions always have their destination operand(s) first followed by the source operand(s).

8.7.2 Addressing Modes and Encoding

The CLA uses the same address to access data and registers as the main CPU. For example, if the main CPU accesses an ePWM register at address 0x00 6800, then the CLA accesses the register using address 0x6800. Since all CLA accessible memory and registers are within the low 64k x 16 of memory, only the low 16-bits of the address are used by the CLA.

To address the CLA data memory, message RAMs and shared peripherals, the CLA supports two addressing modes:

- Direct addressing mode: Uses the address of the variable or register directly.
- Indirect addressing with 16-bit post increment. This mode uses either XAR0 or XAR1.

The CLA does not use a data page pointer or a stack pointer. The two addressing modes are encoded as shown Table 8-8.

Table 8-8. Addressing Modes

Addressing Mode	'addr' Opcode Field Encode(1)	Description
@dir	**0000**	**Direct Addressing Mode** Example 1: MMOV32 MR1, @_VarA Example 2: MMOV32 MR1, @_EPwm1Regs.CMPA.all In this case, the 'mmmm mmmm mmmm mmmm' opcode field is populated with the 16-bit address of the variable. This is the low 16-bits of the address to access the variable using the main CPU. For example, @_VarA populates the address of the variable VarA. and @_EPwm1Regs.CMPA.all populates the address of the CMPA register.
***MAR0[#imm16]++**	**0001**	**MAR0 Indirect Addressing with 16-bit Immediate Post Increment**
***MAR1[#imm16]++**	**0010**	**MAR1 Indirect Addressing with 16-bit Immediate Post Increment** addr = MAR0 (or MAR1) Access memory using the address stored in MAR0 (or MAR1). MAR0 (or MAR1) += #imm16 Then post increment MAR0 (or MAR1) by #imm16. Example 1: MMOV32 MR0, *MAR0[2]++ Example 2: MMOV32 MR1, *MAR1[-2]++ For a post increment of 0, the assembler accepts both *MAR0 and *MAR0[0]++. The 'mmmm mmmm mmmm mmmm' opcode field is populated with the signed 16-bit pointer offset. For example, if #imm16 is 2, then the opcode field is 0x0002. Likewise, if #imm16 is -2, then the opcode field is 0xFFFE. If addition of the 16-bit immediate causes overflow, then the value wraps around on a 16-bit boundary.
***MAR0+[#imm16]**	**0101**	**MAR0 Offset Addressing with 16-bit Immediate Offset**
***MAR1+[#imm16]**	**0110**	**MAR1 Offset Addressing with 16-bit Immediate Offset** addr = MAR0 (or MAR1) + #imm16to the base Add the offset #imm16 address stored in MAR0(MAR1) to access the desired memory location Example 1: MMOV32 MR0, *MAR0+[2] Example 1: MMOV32 MR1, *MAR1+[-2] The ‘mmmm mmmm mmmm mmmm’ opcode field is populated with the signed 16-bit pointer offset. For example, if #imm16 is 2, then the opcode field is 0x0002. Likewise, if #imm16 is -2, then the opcode field is 0xFFFE. If the addition of the 16-bit immediate causes overflow, the value wraps around on a 16-bit boundary.

(1) Values not shown are reserved.

Encoding for the shift fields in the MASR32, MLSR32 and MLSL32 instructions is shown in Table 8-9.

Table 8-9. Shift Field Encoding

Shift Value	'shift' Opcode Field Encode
1	0000
2	0001
3	0010
....	....
32	1111

For instructions that use MRx (where x can be 'a' through 'f') as operands, the trailing alphabet appears in the opcode as a two-bit field. For example:

```
MMPYF32 MRa, MRb, MRc ||
MADDF32 MRd, MRe, MRf
```

whose opcode is,

```
LSW: 0000 ffee ddcc bbaa
MSW: 0111 1010 0000 0000
```

The two-bit field specifies one of four working registers according to Table 8-10.

Table 8-10. Operand Encoding

Two-Bit Field	Working Register
00	MR0
01	MR1
10	MR2
11	MR3

Table 8-11 shows the condition field encoding for conditional instructions such as MNEGF, MSWAPF, MBCNDD, MCCNDD, and MRCNDD.

Table 8-11. Condition Field Encoding

Encode(1)	CNDF	Description	MSTF Flags Tested
0000	NEQ	Not equal to zero	ZF == 0
0001	EQ	Equal to zero	ZF == 1
0010	GT	Greater than zero	ZF == 0 AND NF == 0
0011	GEQ	Greater than or equal to zero	NF == 0
0100	LT	Less than zero	NF == 1
0101	LEQ	Less than or equal to zero	ZF == 1 OR NF == 1
1010	TF	Test flag set	TF == 1
1011	NTF	Test flag not set	TF == 0
1100	LU	Latched underflow	LUF == 1
1101	LV	Latched overflow	LVF == 1
1110	UNC	Unconditional	None
1111	UNCF(2)	Unconditional with flag modification	None

(1) Values not shown are reserved.
(2) This is the default operation, if no CNDF field is specified. This condition allows the ZF and NF flags to be modified when a conditional operation is executed. All other conditions do not modify these flags.

8.7.3 Instructions

The instructions are listed alphabetically.

Instruction Set Summary

MABSF32 MRa, MRb

32-Bit Floating-Point Absolute Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1110 0010 0000
```

Description

The absolute value of MRb is loaded into MRa. Only the sign bit of the operand is modified by the MABSF32 instruction.

```
if (MRb < 0) {MRa = -MRb};
        else {MRa =  MRb};
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified as follows:

```
NF = 0;
ZF = 0;
if ( MRa(30:23) == 0) ZF = 1;
```

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ MR0, #-2.0 ; MR0 = -2.0 (0xC0000000)
MABSF32 MR0, MR0  ; MR0 = 2.0 (0x40000000), ZF = NF = 0
MMOVIZ MR0, #5.0  ; MR0 = 5.0 (0x40A00000)
MABSF32 MR0, MR0  ; MR0 = 5.0 (0x40A00000), ZF = NF = 0
MMOVIZ MR0, #0.0  ; MR0 = 0.0
MABSF32 MR0, MR0  ; MR0 = 0.0 ZF = 1, NF = 0
```

See also

MNEGF32 MRa, MRb {, CNDF}

MADD32 MRa, MRb, MRc

32-Bit Integer Add

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point destination register (MR0 to MR3)
MRc	CLA floating-point destination register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 000cc bbaa
MSW: 0111 1110 1100 0000
```

Description

32-bit integer addition of MRb and MRc.

```
MRa(31:0) = MRb(31:0) + MRc(31:0);
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified based on the integer results of the operation.

```
NF = MRa(31);
ZF = 0;
if(MRa(31:0) == 0) { ZF = 1; };
```

Pipeline

This is a single-cycle instruction.

Example

```
; Given A = (int32)1
;       B = (int32)2
;       C = (int32)-7
;
; Calculate Y2 = A + B + C
;
_Cla1Task1:
       MMOV32 MR0, @_A       ; MR0 = 1 (0x00000001)
       MMOV32 MR1, @_B       ; MR1 = 2 (0x00000002)
       MMOV32 MR2, @_C       ; MR2 = -7 (0xFFFFFFF9)
       MADD32 MR3, MR0, MR1 ; A + B
       MADD32 MR3, MR2, MR3 ; A + B + C = -4 (0xFFFFFFFC)
       MMOV32 @_y2, MR3      ; Store y2
       MSTOP                 ; end of task
```

See also

MAND32 MRa, MRb, MRc
MASR32 MRa, #SHIFT
MLSL32 MRa, #SHIFT
MLSR32 MRa, #SHIFT
MOR32 MRa, MRb, MRc
MXOR32 MRa, MRb, MRc
MSUB32 MRa, MRb, MRc

MADDF32 MRa, #16FHi, MRb

32-Bit Floating-Point Addition

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
#16FHi	A 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0.
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 0111 1100 bbaa
```

Description

Add MRb to the floating-point value represented by the immediate operand. Store the result of the addition in MRa.

#16FHi is a 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0. #16FHi is most useful for representing constants where the lowest 16-bits of the mantissa are 0. Some examples are 2.0 (0x40000000), 4.0 (0x40800000), 0.5 (0x3F000000), and -1.5 (0xBFC00000). The assembler accepts either a hex or float as the immediate value. That is, the value -1.5 can be represented as #-1.5 or #0xBFC0.

```
MRa = MRb + #16FHi:0;
```

This instruction can also be written as MADDF32 MRa, MRb, #16FHi.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MADDF32 generates an underflow condition.
- LVF = 1 if MADDF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

Example

```
; Add to MR1 the value 2.0 in 32-bit floating-point format
; Store the result in MR0
  MADDF32 MR0, #2.0, MR1    ; MR0 = 2.0 + MR1
; Add to MR3 the value -2.5 in 32-bit floating-point format
; Store the result in MR2
  MADDF32 MR2, #-2.5, MR3   ; MR2 = -2.5 + MR3
; Add to MR3 the value 0x3FC00000 (1.5)
; Store the result in MR3
  MADDF32 MR3, #0x3FC0, MR3 ; MR3 = 1.5 + MR3
```

See also

MADDF32 MRa, MRb, #16FHi
MADDF32 MRa, MRb, MRc
MADDF32 MRd, MRe, MRf || MMOV32 MRa, mem32
MADDF32 MRd, MRe, MRf || MMOV32 mem32, MRa
MMPYF32 MRa, MRb, MRc || MADDF32 MRd, MRe, MRf

MADDF32 MRa, MRb, #16FHi

32-Bit Floating-Point Addition

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)
#16FHi	A 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0.

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 0111 1100 bbaa
```

Description

Add MRb to the floating-point value represented by the immediate operand. Store the result of the addition in MRa.

#16FHi is a 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0. #16FHi is most useful for representing constants where the lowest 16-bits of the mantissa are 0. Some examples are 2.0 (0x40000000), 4.0 (0x40800000), 0.5 (0x3F000000), and -1.5 (0xBFC00000). The assembler accepts either a hex or float as the immediate value. That is, the value -1.5 can be represented as #-1.5 or #0xBFC0.

```
MRa = MRb + #16FHi:0;
```

This instruction can also be written as MADDF32 MRa, #16FHi, MRb.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MADDF32 generates an underflow condition.
- LVF = 1 if MADDF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

MADDF32 MRa, MRb, #16FHi (continued)

32-Bit Floating-Point Addition

Example 1

```
; X is an array of 32-bit floating-point values
; Find the maximum value in an array X
; and store the value in Result
;
_Cla1Task1:
    MMOVI16    MAR1,#_X          ; Start address
    MUI16TOF32 MR0, @_len        ; Length of the array
    MNOP                         ; delay for MAR1 load
    MNOP                         ; delay for MAR1 load
    MMOV32     MR1, *MAR1[2]++   ; MR1 = X0
LOOP
    MMOV32     MR2, *MAR1[2]++   ; MR2 = next element
    MMAXF32    MR1,  MR2         ; MR1 = MAX(MR1, MR2)
    MADDF32    MR0, MR0, #-1.0   ; Decrement the counter
    MCMPF32    MR0 #0.0          ; Set/clear flags for MBCNDD
    MNOP
    MNOP
    MNOP
    MBCNDD LOOP, NEQ             ; Branch if not equal to zero
    MMOV32 @_Result, MR1         ; Always executed
    MNOP                         ; Always executed
    MNOP                         ; Always executed
    MSTOP                        ; End of task
```

Example 2

```
; Show the basic operation of MADDF32
;
; Add to MR1 the value 2.0 in 32-bit floating-point format
; Store the result in MR0
    MADDF32 MR0, MR1, #2.0    ; MR0 = MR1 + 2.0
; Add to MR3 the value -2.5 in 32-bit floating-point format
; Store the result in MR2
    MADDF32 MR2, MR3, #-2.5   ; MR2 = MR3 + (-2.5)
; Add to MR0 the value 0x3FC00000 (1.5)
; Store the result in MR0
   MADDF32 MR0, MR0, #0x3FC0  ; MR0 = MR0 + 1.5
```

See also

MADDF32 MRa, #16FHi, MRb
MADDF32 MRa, MRb, MRc
MADDF32 MRd, MRe, MRf || MMOV32 MRa, mem32
MADDF32 MRd, MRe, MRf || MMOV32 mem32, MRa
MMPYF32 MRa, MRb, MRc || MADDF32 MRd, MRe, MRf

MADDF32 MRa, MRb, MRc

32-Bit Floating-Point Addition

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)
MRc	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 000 0000 00cc bbaa
MSW: 0111 1100 0010 0000
```

Description

Add the contents of MRc to the contents of MRb and load the result into MRa.

```
MRa = MRb + MRc;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MADDF32 generates an underflow condition.
- LVF = 1 if MADDF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

Example

```
; Given M1, X1, and B1 are 32-bit floating-point numbers
; Calculate Y1 = M1*X1+B1
;
_Cla1Task1:
   MMOV32 MR0,@M1      ; Load MR0 with M1
   MMOV32 MR1,@X1      ; Load MR1 with X1
   MMPYF32 MR1,MR1,MR0 ; Multiply M1*X1
|| MMOV32 MR0,@B1      ; and in parallel load MR0 with B1
   MADDF32 MR1,MR1,MR0 ; Add M*X1 to B1 and store in MR1
   MMOV32 @Y1,MR1      ; Store the result
   MSTOP               ; end of task
```

See also

MADDF32 MRa, #16FHi, MRb
MADDF32 MRa, MRb, #16FHi
MADDF32 MRd, MRe, MRf || MMOV32 MRa, mem32
MADDF32 MRd, MRe, MRf || MMOV32 mem32, MRa
MMPYF32 MRa, MRb, MRc || MADDF32 MRd, MRe, MRf

MADDF32 MRd, MRe, MRf||MMOV32 mem32, MRa

32-Bit Floating-Point Addition with Parallel Move

Operands

MRd	CLA floating-point destination register for the MADDF32 (MR0 to MR3)
MRe	CLA floating-point source register for the MADDF32 (MR0 to MR3)
MRf	CLA floating-point source register for the MADDF32 (MR0 to MR3)
mem32	32-bit memory location accessed using one of the available addressing modes. This is the destination of the MMOV32.
MRa	CLA floating-point source register for the MMOV32 (MR0 to MR3)

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0101 ffee ddaa addr
```

Description

Perform an MADDF32 and a MMOV32 in parallel. Add MRf to the contents of MRe and store the result in MRd. In parallel move the contents of MRa to the 32-bit location mem32.

```
MRd = MRe + MRf;
[mem32] = MRa;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MADDF32 generates an underflow condition.
- LVF = 1 if MADDF32 generates an overflow condition.

Pipeline

Both MADDF32 and MMOV32 complete in a single cycle.

Example

```
; Given A, B, and C are 32-bit floating-point numbers
; Calculate Y2 = (A * B)
;           Y3 = (A * B) + C
;
_Cla1Task2:
   MMOV32   MR0, @_A         ; Load MR0 with A
   MMOV32   MR1, @_B         ; Load MR1 with B
   MMPYF32  MR1, MR1, MR0    ; Multiply A*B
|| MMOV32   MR0, @_C         ;     and in parallel load MR0 with C
   MADDF32  MR1, MR1, MR0    ; Add (A*B) to C
|| MMOV32   @_Y2, MR1        ;     and in parallel store A*B
   MMOV32   @_Y3, MR1        ; Store the A*B + C
   MSTOP                     ; end of task
```

See also

MADDF32 MRa, #16FHi, MRb
MADDF32 MRa, MRb, #16FHi
MADDF32 MRa, MRb, MRc
MMPYF32 MRa, MRb, MRc || MADDF32 MRd, MRe, MRf
MADDF32 MRd, MRe, MRf || MMOV32 MRa, mem32

MADDF32 MRd, MRe, MRf ||MMOV32 MRa, mem32

32-Bit Floating-Point Addition with Parallel Move

Operands

MRd	CLA floating-point destination register for the MADDF32 (MR0 to MR3). MRd cannot be the same register as MRa.
MRe	CLA floating-point source register for the MADDF32 (MR0 to MR3)
MRf	CLA floating-point source register for the MADDF32 (MR0 to MR3)
MRa	CLA floating-point destination register for the MMOV32 (MR0 to MR3). MRa cannot be the same register as MRd.
mem32	32-bit memory location accessed using one of the available addressing modes. This is the source for the MMOV32.

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0001 ffee ddaa addr
```

Description

Perform an MADDF32 and a MMOV32 operation in parallel. Add MRf to the contents of MRe and store the result in MRd. In parallel move the contents of the 32-bit location mem32 to MRa.

```
MRd = MRe + MRf;
MRa = [mem32];
```

Restrictions

The destination register for the MADDF32 and the MMOV32 must be unique. That is, MRa and MRd cannot be the same register.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MADDF32 generates an underflow condition.
- LVF = 1 if MADDF32 generates an overflow condition.

The MMOV32 Instruction sets the NF and ZF flags as follows:

```
NF = MRa(31);
ZF = 0;
if(MRa(30:23) == 0) { ZF = 1; NF = 0; };
```

Pipeline

The MADDF32 and the MMOV32 both complete in a single cycle.

MADDF32 MRd, MRe, MRf ||MMOV32 MRa, mem32 (continued)

32-Bit Floating-Point Addition with Parallel Move

Example 1

```
; Given A, B, and C are 32-bit floating-point numbers
; Calculate Y1 = A + 4B
;           Y2 = A + C
;
_Cla1Task1:
    MMOV32 MR0, @A          ; Load MR0 with A
    MMOV32 MR1, @B          ; Load MR1 with B
    MMPYF32 MR1, MR1, #4.0 ; Multiply 4 * B
||  MMOV32 MR2, @C            and in parallel load C
    MADDF32 MR3, MR0, MR1  ; Add A + 4B
    MADDF32 MR3, MR0, MR2  ; Add A + C
||  MMOV32 @Y1, MR3         ; and in parallel store A+4B
    MMOV32 @Y2, MR3         ; store A + C MSTOP
                            ; end of task
```

Example 2

```
; Given A, B, and C are 32-bit floating-point numbers
; Calculate Y3 = (A + B)
;           Y4 = (A + B) * C
;
_Cla1Task2:
     MMOV32 MR0, @A          ; Load MR0 with A
     MMOV32 MR1, @B          ; Load MR1 with B
     MADDF32 MR1, MR1, MR0  ; Add A+B
||   MMOV32 MR0, @C          ; and in parallel load MR0 with C
     MMPYF32 MR1, MR1, MR0  ; Multiply (A+B) by C
||   MMOV32 @Y3, MR1         ; and in parallel store A+B
     MMOV32 @Y4, MR1         ; Store the (A+B) * C
     MSTOP                   ; end of task
```

See also

MADDF32 MRa, #16FHi, MRb
MADDF32 MRa, MRb, #16FHi
MADDF32 MRa, MRb, MRc
MADDF32 MRd, MRe, MRf || MMOV32 mem32, MRa
MMPYF32 MRa, MRb, MRc || MADDF32 MRd, MRe, MRf

MAND32 MRa, MRb, MRc

Bitwise AND

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)
MRc	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 00cc bbaa
MSW: 0111 1100 0110 0000
```

Description

Bitwise AND of MRb with MRc.

```
MRa(31:0) = MRb(31:0) AND MRc(31:0);
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified based on the integer results of the operation.

```
NF = MRa(31);
ZF = 0;
if(MRa(31:0) == 0) { ZF = 1; }
```

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ   MR0,    #0x5555  ; MR0 = 0x5555AAAA
MMOVXI   MR0,    #0xAAAA
MMOVIZ   MR1,    #0x5432  ; MR1 = 0x5432FEDC
MMOVXI   MR1,    #0xFEDC
; 0101 AND 0101 = 0101 (5)
; 0101 AND 0100 = 0100 (4)
; 0101 AND 0011 = 0001 (1)
; 0101 AND 0010 = 0000 (0)
; 1010 AND 1111 = 1010 (A)
; 1010 AND 1110 = 1010 (A)
; 1010 AND 1101 = 1000 (8)
; 1010 AND 1100 = 1000 (8)
MAND32 MR2, MR1, MR0      ; MR3 = 0x5410AA88
```

See also

MADD32 MRa, MRb, MRc
MASR32 MRa, #SHIFT
MLSL32 MRa, #SHIFT
MLSR32 MRa, #SHIFT
MOR32 MRa, MRb, MRc
MXOR32 MRa, MRb, MRc
MSUB32 MRa, MRb, MRc

MASR32 MRa, #SHIFT

Arithmetic Shift Right

Operands

MRa	CLA floating-point source/destination register (MR0 to MR3)
#SHIFT	Number of bits to shift (1 to 32)

Opcode

```
LSW: 0000 0000 0shi ftaa
MSW: 0111 1011 0100 0000
```

Description

Arithmetic shift right of MRa by the number of bits indicated. The number of bits can be 1 to 32.

```
MARa(31:0) = Arithmetic Shift(MARa(31:0) by #SHIFT bits);
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified based on the integer results of the operation.

```
NF = MRa(31);
ZF = 0;
if(MRa(31:0) == 0) { ZF = 1; }
```

Pipeline

This is a single-cycle instruction.

Example

```
; Given m2 = (int32)32
;       x2 = (int32)64
;       b2 = (int32)-128
;
; Calculate
;       m2 = m2/2
;       x2 = x2/4
;       b2 = b2/8
;
_Cla1Task2:
   MMOV32 MR0, @_m2 ; MR0 = 32 (0x00000020)
   MMOV32 MR1, @_x2 ; MR1 = 64 (0x00000040)
   MMOV32 MR2, @_b2 ; MR2 = -128 (0xFFFFFF80)
   MASR32 MR0, #1   ; MR0 = 16 (0x00000010)
   MASR32 MR1, #2   ; MR1 = 16 (0x00000010)
   MASR32 MR2, #3   ; MR2 = -16 (0xFFFFFFF0)
   MMOV32 @_m2, MR0 ; store results
   MMOV32 @_x2, MR1
   MMOV32 @_b2, MR2
   MSTOP ; end of task
```

See also

MADD32 MRa, MRb, MRc
MAND32 MRa, MRb, MRc
MLSL32 MRa, #SHIFT
MLSR32 MRa, #SHIFT
MOR32 MRa, MRb, MRc
MXOR32 MRa, MRb, MRc
MSUB32 MRa, MRb, MRc

MBCNDD 16BitDest {, CNDF}

Branch Conditional Delayed

Operands

16BitDest	16-bit destination if condition is true
CNDF	Optional condition tested

Opcode

```
LSW: dest dest dest dest
MSW: 0111 1001 1000 cndf
```

Description

If the specified condition is true, then branch by adding the signed 16BitDest value to the MPC value. Otherwise, continue without branching. If the address overflows, the address wraps around. Therefore, a value of "0xFFFE" puts the MPC back to the MBCNDD instruction.

Refer to the Pipeline section for important information regarding this instruction.

```
if (CNDF == TRUE) MPC += 16BitDest;
```

CNDF is one of the following conditions:

Encode[1]	CNDF	Description	MSTF Flags Tested
0000	NEQ	Not equal to zero	ZF == 0
0001	EQ	Equal to zero	ZF == 1
0010	GT	Greater than zero	ZF == 0 AND NF == 0
0011	GEQ	Greater than or equal to zero	NF == 0
0100	LT	Less than zero	NF == 1
0101	LEQ	Less than or equal to zero	ZF == 1 OR NF == 1
1010	TF	Test flag set	TF == 1
1011	NTF	Test flag not set	TF == 0
1100	LU	Latched underflow	LUF == 1
1101	LV	Latched overflow	LVF == 1
1110	UNC	Unconditional	None
1111	UNCF[2]	Unconditional with flag modification	None

(1) Values not shown are reserved.
(2) This is the default operation, if no CNDF field is specified. This condition allows the ZF and NF flags to be modified when a conditional operation is executed. All other conditions do not modify these flags.

Restrictions

The MBCNDD instruction is not allowed three instructions before or after a MBCNDD, MCCNDD, or MRCNDD instruction. Refer to the Pipeline section for more information.

Flags

This instruction does not modify flags in the MSTF register.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

MBCNDD 16BitDest {, CNDF} (continued)

Branch Conditional Delayed

Pipeline

The MBCNDD instruction alone is a single-cycle instruction. As shown in Table 8-12, 6 instruction slots are executed for each branch; 3 slots before the branch instruction (I2-I4) and 3 slots after the branch instruction (I5-I7). The total number of cycles for a branch taken or not taken depends on the usage of these slots. That is, the number of cycles depends on how many slots are filled with a MNOP as well as which slots are filled. The effective number of cycles for a branch can, therefore, range from 1 to 7 cycles. The number of cycles for a branch taken cannot be the same as for a branch not taken.

Referring to Table 8-12 and Table 8-13, the instructions before and after MBCNDD have the following properties:

- **I1**
  - I1 is the last instruction that can effect the CNDF flags for the MBCNDD instruction. The CNDF flags are tested in the D2 phase of the pipeline. That is, a decision is made whether to branch or not when MBCNDD is in the D2 phase.
  - There are no restrictions on the type of instruction for I1.
- **I2, I3, and I4**
  - The three instructions proceeding MBCNDD can change MSTF flags but have no effect on whether the MBCNDD instruction branches or not. This is because the flag modification occurs after the D2 phase of the MBCNDD instruction.
  - These instructions must not be the following: MSTOP, MDEBUGSTOP, MBCNDD, MCCNDD, or MRCNDD.
- **I5, I6, and I7**
  - The three instructions following MBCNDD are always executed irrespective of whether the branch is taken or not.
  - These instructions must not be the following: MSTOP, MDEBUGSTOP, MBCNDD, MCCNDD, or MRCNDD.

```
<Instruction 1>   ; I1 Last instruction that can affect flags for
                  ; the MBCNDD operation
<Instruction 2>   ; I2 Cannot be stop, branch, call or return
<Instruction 3>   ; I3 Cannot be stop, branch, call or return
<Instruction 4>   ; I4 Cannot be stop, branch, call or return
MBCNDD _Skip, NEQ ; Branch to Skip if not eqal to zero
                  ; Three instructions after MBCNDD are always
                  ; executed whether the branch is taken or not
<Instruction 5>   ; I5 Cannot be stop, branch, call or return
<Instruction 6>   ; I6 Cannot be stop, branch, call or return
<Instruction 7>   ; I7 Cannot be stop, branch, call or return
<Instruction 8>   ; I8
<Instruction 9>   ; I9
....
_Skip:
 <Destination 1>  ; d1 Can be any instruction
 <Destination 2>  ; d2
 <Destination 3>  ; d3
....
....
MSTOP
....
```

MBCNDD 16BitDest {, CNDF} (continued)

Branch Conditional Delayed

Table 8-12. Pipeline Activity for MBCNDD, Branch Not Taken

Instruction	F1	F2	D1	D2	R1	R2	E	W
I1	I1							
I2	I2	I1						
I3	I3	I2	I1					
I4	I4	I3	I2	I1				
MBCNDD	MBCNDD	I4	I3	I2	I1			
I5	I5	MBCNDD	I4	I3	I2	I1		
I6	I6	I5	MBCNDD	I4	I3	I2	I1	
I7	I7	I6	I5	MBCNDD	I4	I3	I2	
I8	I8	I7	I6	I5	-	I4	I3	
I9	I9	I8	I7	I6	I5	-	I4	
I10	I10	I9	I8	I7	I6	I5	-	
		I10	I9	I8	I7	I6	I5	
			I10	I9	I8	I7	I6	
				I10	I9	I8	I7	
					I10	I9	I8	
						I10	I9	
							I10	

Table 8-13. Pipeline Activity for MBCNDD, Branch Taken

Instruction	F1	F2	D1	D2	R1	R2	E	W
I1	I1							
I2	I2	I1						
I3	I3	I2	I1					
I4	I4	I3	I2	I1				
MBCNDD	MBCNDD	I4	I3	I2	I1			
I5	I5	MBCNDD	I4	I3	I2	I1		
I6	I6	I5	MBCNDD	I4	I3	I2	I1	
I7	I7	I6	I5	MBCNDD	I4	I3	I2	
d1	d1	I7	I6	I5	-	I4	I3	
d2	d2	d1	I7	I6	I5	-	I4	
d3	d3	d2	d1	I7	I6	I5	-	
		d3	d2	d1	I7	I6	I5	
			d3	d2	d1	I7	I6	
				d3	d2	d1	I7	
					d3	d2	d1	
						d3	d2	
							d3	

MBCNDD 16BitDest {, CNDF} (continued)

Branch Conditional Delayed

Example 1

```
; if (State == 0.1)
; RampState = RampState || RAMPMASK
; else if (State == 0.01)
; CoastState = CoastState || COASTMASK
; else
; SteadyState = SteadyState || STEADYMASK
;
_Cla1Task1:
 MMOV32 MR0, @State
 MCMPF32 MR0, #0.1        ; Affects flags for 1st MBCNDD (A)
 MNOP
 MNOP
 MNOP
 MBCNDD Skip1, NEQ        ; (A) If State != 0.1, go to Skip1
 MNOP ; Always executed
 MNOP ; Always executed
 MNOP ; Always executed
 MMOV32 MR1, @RampState   ; Execute if (A) branch not taken
 MMOVXI MR2, #RAMPMASK    ; Execute if (A) branch not taken
 MOR32 MR1, MR2           ; Execute if (A) branch not taken
 MMOV32 @RampState, MR1   ; Execute if (A) branch not taken
 MSTOP                    ; end of task if (A) branch not taken
Skip1:
 MCMPF32 MR0,#0.01        ; Affects flags for 2nd MBCNDD (B)
 MNOP
 MNOP
 MNOP
 MBCNDD Skip2,NEQ         ; (B) If State != 0.01, go to Skip2
 MNOP ; Always executed
 MNOP ; Always executed
 MNOP ; Always executed
 MMOV32 MR1, @CoastState  ; Execute if (B) branch not taken
 MMOVXI MR2, #COASTMASK   ; Execute if (B) branch not taken
 MOR32 MR1, MR2           ; Execute if (B) branch not taken
 MMOV32 @CoastState, MR1  ; Execute if (B) branch not taken
 MSTOP
Skip2:
 MMOV32 MR3, @SteadyState ; Executed if (B) branch taken
 MMOVXI MR2, #STEADYMASK  ; Executed if (B) branch taken
 MOR32 MR3, MR2           ; Executed if (B) branch taken
 MMOV32 @SteadyState, MR3 ; Executed if (B) branch taken
 MSTOP
```

MBCNDD 16BitDest {, CNDF} (continued)

Branch Conditional Delayed

Example 2

```
; This example is the same as Example 1, except
; the code is optimized to take advantage of delay slots
;
; if (State == 0.1)
; RampState = RampState || RAMPMASK
; else if (State == 0.01)
; CoastState = CoastState || COASTMASK
; else
; SteadyState = SteadyState || STEADYMASK
;
_Cla1Task2:
 MMOV32 MR0, @State
 MCMPF32 MR0, #0.1          ; Affects flags for 1st MBCNDD (A)
 MCMPF32 MR0, #0.01         ; Check used by 2nd MBCNDD (B)
 MTESTTF EQ                 ; Store EQ flag in TF for 2nd MBCNDD (B)
 MNOP
 MBCNDD Skip1, NEQ          ; (A) If State != 0.1, go to Skip1
 MMOV32 MR1, @RampState     ; Always executed
 MMOVXI MR2, #RAMPMASK      ; Always executed
 MOR32 MR1, MR2             ; Always executed
 MMOV32 @RampState, MR1     ; Execute if (A) branch not taken
 MSTOP                      ; end of task if (A) branch not taken
Skip1:
 MMOV32 MR3, @SteadyState
 MMOVXI MR2, #STEADYMASK
 MOR32 MR3, MR2
 MBCNDD Skip2, NTF          ; (B) if State != .01, go to Skip2
 MMOV32 MR1, @CoastState    ; Always executed
 MMOVXI MR2, #COASTMASK     ; Always executed
 MOR32 MR1, MR2             ; Always executed
 MMOV32 @CoastState, MR1    ; Execute if (B) branch not taken
 MSTOP                      ; end of task if (B) branch not taken
Skip2:
 MMOV32 @SteadyState, MR3   ; Executed if (B) branch taken
 MSTOP
```

See also

MCCNDD 16BitDest, CNDF
MRCNDD CNDF

MCCNDD 16BitDest {, CNDF}

Call Conditional Delayed

Operands

16BitDest	16-bit destination if condition is true
CNDF	Optional condition to be tested

Opcode

```
LSW: dest dest dest dest
MSW: 0111 1001 1001 cndf
```

Description

If the specified condition is true, then store the return address in the RPC field of MSTF and make the call by adding the signed 16BitDest value to the MPC value. Otherwise, continue code execution without making the call. If the address overflows, the address wraps around. Therefore a value of "0xFFFE" puts the MPC back to the MCCNDD instruction.

Refer to the Pipeline section for important information regarding this instruction.

```
if (CNDF == TRUE)
{
    RPC = return address;
    MPC += 16BitDest;
};
```

CNDF is one of the following conditions:

Encode(1)	CNDF	Description	MSTF Flags Tested
0000	NEQ	Not equal to zero	ZF == 0
0001	EQ	Equal to zero	ZF == 1
0010	GT	Greater than zero	ZF == 0 AND NF == 0
0011	GEQ	Greater than or equal to zero	NF == 0
0100	LT	Less than zero	NF == 1
0101	LEQ	Less than or equal to zero	ZF == 1 OR NF == 1
1010	TF	Test flag set	TF == 1
1011	NTF	Test flag not set	TF == 0
1100	LU	Latched underflow	LUF == 1
1101	LV	Latched overflow	LVF == 1
1110	UNC	Unconditional	None
1111	UNCF(2)	Unconditional with flag modification	None

(1) Values not shown are reserved.
(2) This is the default operation if no CNDF field is specified. This condition allows the ZF and NF flags to be modified when a conditional operation is executed. All other conditions do not modify these flags.

Restrictions

The MCCNDD instruction is not allowed three instructions before or after a MBCNDD, MCCNDD, or MRCNDD instruction. Refer to the Pipeline section for more details.

Flags

This instruction does not modify flags in the MSTF register.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

MCCNDD 16BitDest {, CNDF} (continued)

Call Conditional Delayed

Pipeline

The MCCNDD instruction alone is a single-cycle instruction. As shown in Table 8-14, 6 instruction slots are executed for each call; 3 before the call instruction (I2-I4) and 3 after the call instruction (I5-I7). The total number of cycles for a call taken or not taken depends on the usage of these slots. That is, the number of cycles depends on how many slots are filled with a MNOP as well as which slots are filled. The effective number of cycles for a call can, therefore, range from 1 to 7 cycles. The number of cycles for a call taken cannot be the same as for a call not taken.

Referring to the following code fragment and the pipeline diagrams in Table 8-14 and Table 8-15, the instructions before and after MCCNDD have the following properties:

- **I1**
  - I1 is the last instruction that can effect the CNDF flags for the MCCNDD instruction. The CNDF flags are tested in the D2 phase of the pipeline. That is, a decision is made whether to branch or not when MCCNDD is in the D2 phase.
  - There are no restrictions on the type of instruction for I1.
- **I2, I3, and I4**
  - The three instructions proceeding MCCNDD can change MSTF flags but have no effect on whether the MCCNDD instruction makes the call or not. This is because the flag modification occurs after the D2 phase of the MCCNDD instruction.
  - These instructions must not be the following: MSTOP, MDEBUGSTOP, MBCNDD, MCCNDD, or MRCNDD.
- **I5, I6, and I7**
  - The three instructions following MBCNDD are always executed irrespective of whether the branch is taken or not.
  - These instructions must not be the following: MSTOP, MDEBUGSTOP, MBCNDD, MCCNDD, or MRCNDD.

MCCNDD 16BitDest {, CNDF} (continued)

Call Conditional Delayed

```
<Instruction 1>    ; I1 Last instruction that can affect flags for
                   ;    the MCCNDD operation
<Instruction 2>    ; I2 Cannot be stop, branch, call or return
<Instruction 3>    ; I3 Cannot be stop, branch, call or return
<Instruction 4>    ; I4 Cannot be stop, branch, call or return
MCCNDD _func, NEQ  ; Call to func if not eqal to zero
                   ; Three instructions after MCCNDD are always
                   ; executed whether the call is taken or not
<Instruction 5>    ; I5 Cannot be stop, branch, call or return
<Instruction 6>    ; I6 Cannot be stop, branch, call or return
<Instruction 7>    ; I7 Cannot be stop, branch, call or return
<Instruction 8>    ; I8 The address of this instruction is saved
                   ;    in the RPC field of the MSTF register.
                   ;    Upon return this value is loaded into MPC
                   ;    and fetching continues from this point.
<Instruction 9>    ; I9
....
_func:
<Destination 1>    ; d1 Can be any instruction
<Destination 2>    ; d2
<Destination 3>    ; d3
<Destination 4>    ; d4 Last instruction that can affect flags for
                   ;    the MRCNDD operation
<Destination 5>    ; d5 Cannot be stop, branch, call or return
<Destination 6>    ; d6 Cannot be stop, branch, call or return
<Destination 7>    ; d7 Cannot be stop, branch, call or return
MRCNDD UNC         ; Return to <Instruction 8>, unconditional
                   ; Three instructions after MRCNDD are always
                   ; executed whether the return is taken or not
<Destination 8>    ; d8 Cannot be stop, branch, call or return
<Destination 9>    ; d9 Cannot be stop, branch, call or return
<Destination 10>   ; d10 Cannot be stop, branch, call or return
<Destination 11>   ; d11
....
MSTOP
```

Table 8-14. Pipeline Activity for MCCNDD, Call Not Taken

Instruction	F1	F2	D1	D2	R1	R2	E	W
I1	I1							
I2	I2	I1						
I3	I3	I2	I1					
I4	I4	I3	I2	I1				
MCCNDD	MCCNDD	I4	I3	I2	I1			
I5	I5	MCCNDD	I4	I3	I2	I1		
I6	I6	I5	MCCNDD	I4	I3	I2	I1	
I7	I7	I6	I5	MCCNDD	I4	I3	I2	
I8	I8	I7	I6	I5	-	I4	I3	
I9	I9	I8	I7	I6	I5	-	I4	
I10	I10	I9	I8	I7	I6	I5	-	
etc		I10	I9	I8	I7	I6	I5	
....			I10	I9	I8	I7	I6	
....				I10	I9	I8	I7	
....					I10	I9	I8	
						I10	I9	
							I10	

MCCNDD 16BitDest {, CNDF} (continued)

Call Conditional Delayed

Table 8-15. Pipeline Activity for MCCNDD, Call Taken

Instruction	F1	F2	D1	D2	R1	R2	E	W
I1	I1							
I2	I2	I1						
I3	I3	I2	I1					
I4	I4	I3	I2	I1				
MCCNDD	MCCNDD	I4	I3	I2	I1			
I5	I5	MCCNDD	I4	I3	I2	I1		
I6	I6	I5	MCCNDD	I4	I3	I2	I1	
I7 (1)	I7	I6	I5	MCCNDD	I4	I3	I2	
d1	d1	I7	I6	I5	-	I4	I3	
d2	d2	d1	I7	I6	I5	-	I4	
d3	d3	d2	d1	I7	I6	I5	-	
etc		d3	d2	d1	I7	I6	I5	
....			d3	d2	d1	I7	I6	
....				d3	d2	d1	I7	
....					d3	d2	d1	
						d3	d2	
							d3	

(1) The RPC value in the MSTF register points to the instruction following I7 (instruction I8).

See also

MBCNDD #16BitDest, CNDF
MMOV32 mem32, MSTF
MMOV32 MSTF, mem32
MRCNDD CNDF

MCLRC BGINTM

Clear Background Task Interrupt Mask

Operands

None	This instruction does not have any operands

Opcode

```
LSW: 0000 0000 0000 0000
MSW: 0111 1111 0111 0000
```

Description

This instruction clears the background task interrupt mask (BGINTM) bit in the MSTSBGRND register, allowing any code thereafter to be interrupted by a higher priority task. This instruction clears the BGINTM bit at the end of the D2 phase.

Note

This instruction does not require the MEALLOW bit to be asserted before or deasserted after clearing BGINTM.

Flags

This instruction does not modify flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MCLRC BGINTM        ; Allow the background task to be
                    ; interrupted by clearing the
                    ; MSTSBGRND.BGINTM bit
```

See also

MSETC BGINTM

MCMP32 MRa, MRb

32-Bit Integer Compare for Equal, Less Than or Greater Than

Operands

MRa	CLA floating-point source register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1111 0010 0000
```

Description

Set ZF and NF flags on the result of MRa - MRb where MRa and MRb are 32-bit integers. For a floating-point compare, refer to MCMPF32.

Note

A known hardware issue exists in the MCMP32 instruction. Signed-integer comparisons using MCMP32 alone set the status bits in a way that is not useful for comparison when the difference between the two operands is too large, such as when the inputs have opposite sign and are near the extreme 32-bit signed values. This affects both signed and unsigned integer comparisons.

The compiler (version 18.1.5.LTS or higher) has implemented a workaround for this issue. The compiler checks the upper bits of the operands by performing a floating point comparison before proceeding to do the integer comparison or subtraction.

The compiler flag --cla_signed_compare_workaround enables this workaround.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified based on the integer results of the operation.

```
If(MRa ==    MRb) {ZF=1; NF=0;}
If(MRa >  MRb) {ZF=0; NF=0;}
If(MRa <  MRb) {ZF=0; NF=1;}
```

Pipeline

This is a single-cycle instruction.

Example

```
; Behavior of ZF and NF flags for different comparisons
;
; Given A = (int32)1
;       B = (int32)2
;       C = (int32)-7
;
    MMOV32 MR0, @_A ; MR0 = 1 (0x00000001)
    MMOV32 MR1, @_B ; MR1 = 2 (0x00000002)
    MMOV32 MR2, @_C ; MR2 = -7 (0xFFFFFFF9)
    MCMP32 MR2, MR2 ; NF = 0, ZF = 1
    MCMP32 MR0, MR1 ; NF = 1, ZF = 0
    MCMP32 MR1, MR0 ; NF = 0, ZF = 0
```

See also

MADD32 MRa, MRb, MRc
MSUB32 MRa, MRb, MRc

MCMPF32 MRa, MRb

32-Bit Floating-Point Compare for Equal, Less Than or Greater Than

Operands

MRa	CLA floating-point source register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1101 0000 0000
```

Description

Set ZF and NF flags on the result of MRa - MRb. The MCMPF32 instruction is performed as a logical compare operation. This is possible because of the IEEE format offsetting the exponent. Basically the bigger the binary number, the bigger the floating-point value.

Special cases for inputs:

- Negative zero is treated as positive zero.
- A denormalized value is treated as positive zero.
- Not-a-Number (NaN) is treated as infinity.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified as follows:

```
If(MRa == MRb)   {ZF=1; NF=0;}
If(MRa > MRb) {ZF=0; NF=0;}
If(MRa < MRb) {ZF=0; NF=1;}
```

Pipeline

This is a single-cycle instruction.

Example

```
; Behavior of ZF and NF flags for different comparisons
    MMOVIZ    MR1, #-2.0  ; MR1 = -2.0 (0xC0000000)
    MMOVIZ    MR0, #5.0   ; MR0 = 5.0 (0x40A00000)
    MCMPF32   MR1, MR0    ; ZF = 0, NF = 1
    MCMPF32   MR0, MR1    ; ZF = 0, NF = 0
    MCMPF32   MR0, MR0    ; ZF = 1, NF = 0
```

See also

MCMPF32 MRa, #16FHi
MMAXF32 MRa, #16FHi
MMAXF32 MRa, MRb
MMINF32 MRa, #16FHi
MMINF32 MRa, MRb

MCMPF32 MRa, #16FHi

32-Bit Floating-Point Compare for Equal, Less Than or Greater Than

Operands

MRa	CLA floating-point source register (MR0 to MR3)
#16FHi	A 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0.

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 1000 1100 00aa
```

Description

Compare the value in MRa with the floating-point value represented by the immediate operand. Set the ZF and NF flags on (MRa - #16FHi:0).

#16FHi is a 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0. This addressing mode is most useful for constants where the lowest 16-bits of the mantissa are 0. Some examples are 2.0 (0x40000000), 4.0 (0x40800000), 0.5 (0x3F000000), and -1.5 (0xBFC00000). The assembler accepts either a hex or float as the immediate value. That is, -1.5 can be represented as #-1.5 or #0xBFC0.

The MCMPF32 instruction is performed as a logical compare operation. This is possible because of the IEEE floating-point format offsets the exponent. Basically the bigger the binary number, the bigger the floating-point value.

Special cases for inputs:

- Negative zero is treated as positive zero.
- Denormalized value is treated as positive zero.
- Not-a-Number (NaN) is treated as infinity.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified as follows:

```
If(MRa == #16FHi:0) {ZF=1, NF=0;}
If(MRa > #16FHi:0) {ZF=0, NF=0;}
If(MRa < #16FHi:0) {ZF=0, NF=1;}
```

Pipeline

This is a single-cycle instruction

Example 1

```
; Behavior of ZF and NF flags for different comparisons
  MMOVIZ      MR1, #-2.0  ; MR1 = -2.0 (0xC0000000)
  MMOVIZ      MR0, #5.0   ; MR0 = 5.0 (0x40A00000)
  MCMPF32     MR1, #-2.2  ; ZF = 0, NF = 0
  MCMPF32     MR0, #6.5   ; ZF = 0, NF = 1
  MCMPF32     MR0, #5.0   ; ZF = 1, NF = 0
```

MCMPF32 MRa, #16FHi (continued)

32-Bit Floating-Point Compare for Equal, Less Than or Greater Than

Example 2

```
; X is an array of 32-bit floating-point values
; and has length elements. Find the maximum value in
; the array and store the value in Result
;
; Note: MCMPF32 and MSWAPF can be replaced with MMAXF32
;
_Cla1Task1:
  MMOVI16 MAR1,#_X         ; Start address
  MUI16TOF32 MR0, @_len    ; Length of the array
  MNOP                     ; delay for MAR1 load
  MNOP                     ; delay for MAR1 load
  MMOV32 MR1, *MAR1[2]++   ; MR1 = X0
LOOP
  MMOV32 MR2, *MAR1[2]++   ; MR2 = next element
  MCMPF32 MR2, MR1         ; Compare MR2 with MR1
  MSWAPF MR1, MR2, GT      ; MR1 = MAX(MR1, MR2)
  MADDF32 MR0, MR0, #-1.0 ; Decrememt the counter
  MCMPF32 MR0 #0.0         ; Set/clear flags for MBCNDD
  MNOP
  MNOP
  MNOP
  MBCNDD LOOP, NEQ         ; Branch if not equal to zero
  MMOV32 @_Result, MR1     ; Always executed
  MNOP                     ; Always executed
  MNOP                     ; Always executed
  MSTOP                    ; End of task
```

See also

MCMPF32 MRa, MRb
MMAXF32 MRa, #16FHi
MMAXF32 MRa, MRb
MMINF32 MRa, #16FHi
MMINF32 MRa, MRb

MDEBUGSTOP

Debug Stop Task

Operands

none	This instruction does not have any operands

Opcode

```
LSW: 0000 0000 0000 0000
MSW: 0111 1111 0110 0000
```

Description

When CLA breakpoints are enabled, the MDEBUGSTOP instruction is used to halt a task so that the task can be debugged. That is, MDEBUGSTOP is the CLA breakpoint. If CLA breakpoints are not enabled, the MDEBUGSTOP instruction behaves like a MNOP. Unlike the MSTOP, the MIRUN flag is not cleared and an interrupt is not issued. A single-step or run operation continues execution of the task.

Restrictions

The MDEBUGSTOP instruction cannot be placed 3 instructions before or after a MBCNDD, MCCNDD, or MRCNDD instruction.

Flags

This instruction does not modify flags in the MSTF register.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

See also

MSTOP , MDEBUGSTOP1

MDEBUGSTOP1

Software Breakpoint

Operands

none	This instruction does not have any operands

Opcode

```
LSW: 0000 0000 0000 0000
MSW: 0111 1111 0011 0000
```

Description

The instruction at which a software breakpoint is placed is replaced by the MDEBUGSTOP1 instruction. The instruction halts execution once the instruction reaches the D2 phase in the pipeline; at that point, the subsequent instructions that were fetched, after the halt, are flushed from the pipeline. The replace instruction is re-fetched after this and execution continues normally (either in run or step mode).

See Section 8.4.3 for a detailed explanation of the operation.

Restrictions

The MDEBUGSTOP1 instruction cannot be placed 3 instructions before or after a MBCNDD, MCCNDD, or MRCNDD instruction.

Flags

This instruction does not modify flags in the MSTF register.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

See also

MSTOP, MDEBUGSTOP

MEALLOW

Enable CLA Write Access to EALLOW Protected Registers

Operands

none	This instruction does not have any operands

Opcode

```
LSW: 0000 0000 0000 0000
MSW: 0111 1111 1001 0000
```

Description

This instruction sets the MEALLOW bit in the CLA status register MSTF. When this bit is set, the CLA is allowed write access to EALLOW protected registers. To again protect against CLA writes to protected registers, use the MEDIS instruction.

MEALLOW and MEDIS only control CLA write access; reads are allowed even if MEALLOW has not been executed. MEALLOW and MEDIS are also independent from the main CPU's EALLOW/EDIS. This instruction does not modify the EALLOW bit in the main CPU's status register. The MEALLOW bit in MSTF only controls access for the CLA while the EALLOW bit in the ST1 register only controls access for the main CPU.

As with EALLOW, the MEALLOW bit is overridden using the JTAG port, allowing full control of register accesses during debug from Code Composer Studio.

Flags

This instruction does not modify flags in the MSTF register.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
; C header file including definition of
; the EPwm1Regs structure
;
; The ePWM TZSEL register is EALLOW protected
;
   .cdecls C,LIST,"CLAShared.h"
   ...
_Cla1Task1:
   ...
   MEALLOW                            ; Allow CLA write access
   MMOV16 @_EPwm1Regs.TZSEL.all, MR3 ; Write to TZSEL
   MEDIS                              ; Disallow CLA write access
   ...
   ...
   MSTOP
```

See also

MEDIS

MEDIS

Disable CLA Write Access to EALLOW Protected Registers

Operands

none	This instruction does not have any operands

Opcode

```
LSW: 0000 0000 0000 0000
MSW: 0111 1111 1011 0000
```

Description

This instruction clears the MEALLOW bit in the CLA status register MSTF. When this bit is clear, the CLA is not allowed write access to EALLOW-protected registers. To enable CLA writes to protected registers, use the MEALLOW instruction.

MEALLOW and MEDIS only control CLA write access; reads are allowed even if MEALLOW has not been executed. MEALLOW and MEDIS are also independent from the main CPU's EALLOW/EDIS. This instruction does not modify the EALLOW bit in the main CPU's status register. The MEALLOW bit in MSTF only controls access for the CLA while the EALLOW bit in the ST1 register only controls access for the main CPU.

As with EALLOW, the MEALLOW bit is overridden using the JTAG port, allowing full control of register accesses during debug from the Code Composer Studio™ IDE.

Flags

This instruction does not modify flags in the MSTF register.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
; C header file including definition of
; the EPwm1Regs structure
;
; The ePWM TZSEL register is EALLOW protected
;
   .cdecls C,LIST,"CLAShared.h"
   ...
_Cla1Task1:
   ...
   MEALLOW                            ; Allow CLA write access
   MMOV16 @_EPwm1Regs.TZSEL.all, MR3 ; Write to TZSEL
   MEDIS                              ; Disallow CLA write access
   ...
   ...
   MSTOP
```

See also

MEALLOW

MEINVF32 MRa, MRb

32-Bit Floating-Point Reciprocal Approximation

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1111 0000 0000
```

Description

This operation generates an estimate of 1/X in 32-bit floating-point format accurate to approximately 8 bits. This value can be used in a Newton-Raphson algorithm to get a more accurate answer. That is:

```
Ye = Estimate(1/X);
Ye = Ye*(2.0 - Ye*X);
Ye = Ye*(2.0 - Ye*X);
```

After two iterations of the Newton-Raphson algorithm, you get an exact answer accurate to the 32-bit floating-point format. On each iteration, the mantissa bit accuracy approximately doubles. The MEINVF32 operation does not generate a negative zero, DeNorm, or NaN value.

```
MRa = Estimate of 1/MRb;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MEINVF32 generates an underflow condition.
- LVF = 1 if MEINVF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

Example

```
; Calculate Num/Den using a Newton-Raphson algorithum for 1/Den
; Ye = Estimate(1/X)
; Ye = Ye*(2.0 - Ye*X)
; Ye = Ye*(2.0 - Ye*X)
;
_Cla1Task1:
    MMOV32 MR1, @_Den      ; MR1 = Den
    MEINVF32 MR2, MR1      ; MR2 = Ye = Estimate(1/Den)
    MMPYF32 MR3, MR2, MR1  ; MR3 = Ye*Den
    MSUBF32 MR3, #2.0, MR3 ; MR3 = 2.0 - Ye*Den
    MMPYF32 MR2, MR2, MR3  ; MR2 = Ye = Ye*(2.0 - Ye*Den)
    MMPYF32 MR3, MR2, MR1  ; MR3 = Ye*Den
 || MMOV32 MR0, @_Num      ; MR0 = Num
    MSUBF32 MR3, #2.0, MR3 ; MR3 = 2.0 - Ye*Den
    MMPYF32 MR2, MR2, MR3  ; MR2 = Ye = Ye*(2.0 - Ye*Den)
 || MMOV32 MR1, @_Den      ; Reload Den To Set Sign
    MNEGF32 MR0, MR0, EQ   ; if(Den == 0.0) Change Sign Of Num
    MMPYF32 MR0, MR2, MR0  ; MR0 = Y = Ye*Num
    MMOV32 @_Dest, MR0     ; Store result
    MSTOP                  ; end of task
```

See also

MEISQRTF32 MRa, MRb

MEISQRTF32 MRa, MRb

32-Bit Floating-Point Square-Root Reciprocal Approximation

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1110 0100 0000
```

Description

This operation generates an estimate of 1/sqrt(X) in 32-bit floating-point format accurate to approximately 8 bits. This value can be used in a Newton-Raphson algorithm to get a more accurate answer. That is:

```
Ye = Estimate(1/sqrt(X));
Ye = Ye*(1.5 - Ye*Ye*X/2.0);
Ye = Ye*(1.5 - Ye*Ye*X/2.0);
```

After 2 iterations of the Newton-Raphson algorithm, you get an exact answer accurate to the 32-bit floating-point format. On each iteration, the mantissa bit accuracy approximately doubles. The MEISQRTF32 operation does not generate a negative zero, DeNorm, or NaN value.

```
MRa = Estimate of 1/sqrt (MRb);
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MEISQRTF32 generates an underflow condition.
- LVF = 1 if MEISQRTF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

Example

```
; Y = sqrt(X)
; Ye = Estimate(1/sqrt(X));
; Ye = Ye*(1.5 - Ye*Ye*X*0.5)
; Ye = Ye*(1.5 - Ye*Ye*X*0.5)
; Y = X*Ye
;
_Cla1Task3:
    MMOV32 MR0, @_x          ; MR0 = X
    MEISQRTF32 MR1, MR0      ; MR1 = Ye = Estimate(1/sqrt(X))
    MMOV32 MR1, @_x, EQ      ; if(X == 0.0) Ye = 0.0
    MMPYF32 MR3, MR0, #0.5   ; MR3 = X*0.5
    MMPYF32 MR2, MR1, MR3    ; MR2 = Ye*X*0.5
    MMPYF32 MR2, MR1, MR2    ; MR2 = Ye*Ye*X*0.5
    MSUBF32 MR2, #1.5, MR2   ; MR2 = 1.5 - Ye*Ye*X*0.5
    MMPYF32 MR1, MR1, MR2    ; MR1 = Ye = Ye*(1.5 - Ye*Ye*X*0.5)
    MMPYF32 MR2, MR1, MR3    ; MR2 = Ye*X*0.5
    MMPYF32 MR2, MR1, MR2    ; MR2 = Ye*Ye*X*0.5
    MSUBF32 MR2, #1.5, MR2   ; MR2 = 1.5 - Ye*Ye*X*0.5
    MMPYF32 MR1, MR1, MR2    ; MR1 = Ye = Ye*(1.5 - Ye*Ye*X*0.5)
    MMPYF32 MR0, MR1, MR0    ; MR0 = Y = Ye*X
    MMOV32 @_y, MR0          ; Store Y = sqrt(X)
    MSTOP                    ; end of task
```

MEISQRTF32 MRa, MRb (continued)

32-Bit Floating-Point Square-Root Reciprocal Approximation

See also MEINVF32 MRa, MRb

MF32TOI16 MRa, MRb

Convert 32-Bit Floating-Point Value to 16-Bit Integer

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1101 1110 0000
```

Description

Convert a 32-bit floating point value in MRb to a 16-bit integer and truncate. The result is stored in MRa.

```
MRa(15:0) = F32TOI16(MRb);
MRa(31:16) = sign extension of MRa(15);
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ      MR0, #5.0   ; MR0        = 5.0 (0x40A00000)
MF32TOI16   MR1, MR0    ; MR1(15:0)  = MF32TOI16(MR0) = 0x0005
                        ; MR1(31:16) = Sign extension of MR1(15) = 0x0000
MMOVIZ      MR2, #-5.0  ; MR2        = -5.0 (0xC0A00000)
MF32TOI16   MR3, MR2    ; MR3(15:0)  = MF32TOI16(MR2) = -5 (0xFFFB)
                        ; MR3(31:16) = Sign extension of MR3(15) = 0xFFFF
```

See also

MF32TOI16R MRa, MRb
MF32TOUI16 MRa, MRb
MF32TOUI16R MRa, MRb
MI16TOF32 MRa, MRb
MI16TOF32 MRa, mem16
MUI16TOF32 MRa, mem16
MUI16TOF32 MRa, MRb

MF32TOI16R MRa, MRb

Convert 32-Bit Floating-Point Value to 16-Bit Integer and Round

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1110 0110 0000
```

Description

Convert the 32-bit floating point value in MRb to a 16-bit integer and round to the nearest even value. The result is stored in MRa.

```
MRa(15:0) = F32TOI16round(MRb);
MRa(31:16) = sign extension of MRa(15);
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ MR0, #0x3FD9    ; MR0(31:16) = 0x3FD9
MMOVXI MR0, #0x999A    ; MR0(15:0) = 0x999A
                       ; MR0 = 1.7 (0x3FD9999A)
MF32TOI16R MR1, MR0    ; MR1(15:0) = MF32TOI16round (MR0) = 2 (0x0002)
                       ; MR1(31:16) = Sign extension of MR1(15) = 0x0000
MMOVF32 MR2, #-1.7     ; MR2 = -1.7 (0xBFD9999A)
MF32TOI16R MR3, MR2    ; MR3(15:0) = MF32TOI16round (MR2) = -2 (0xFFFE)
                       ; MR3(31:16) = Sign extension of MR2(15) = 0xFFFF
```

See also

MF32TOI16 MRa, MRb
MF32TOUI16 MRa, MRb
MF32TOUI16R MRa, MRb
MI16TOF32 MRa, MRb
MI16TOF32 MRa, mem16
MUI16TOF32 MRa, mem16
MUI16TOF32 MRa, MRb

MF32TOI32 MRa, MRb

Convert 32-Bit Floating-Point Value to 32-Bit Integer

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1101 0110 0000
```

Description

Convert the 32-bit floating-point value in MRb to a 32-bit integer value and truncate. Store the result in MRa.

```
MRa = F32TOI32(MRb);
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example 1

```
MMOVF32    MR2, #11204005.0   ; MR2 = 11204005.0 (0x4B2AF5A5)
MF32TOI32  MR3, MR2           ; MR3 = MF32TOI32(MR2) = 11204005 (0x00AAF5A5)
MMOVF32    MR0, #-11204005.0  ; MR0 = -11204005.0 (0xCB2AF5A5)
MF32TOI32  MR1, MR0           ; MR1 = MF32TOI32(MR0) = -11204005 (0xFF550A5B)
```

Example 2

```
; Given X, M and B are IQ24 numbers:
; X = IQ24(+2.5) = 0x02800000
; M = IQ24(+1.5) = 0x01800000
; B = IQ24(-0.5) = 0xFF800000
;
; Calculate Y = X * M + B
;
; Convert M, X, and B from IQ24 to float
;
_Cla1Task2:
   MI32TOF32 MR0, @_M          ; MR0 = 0x4BC00000
   MI32TOF32 MR1, @_X          ; MR1 = 0x4C200000
   MI32TOF32 MR2, @_B          ; MR2 = 0xCB000000
   MMPYF32   MR0, MR0, #0x3380 ; M = 1/(1*2^24) * iqm = 1.5 (0x3FC00000)
   MMPYF32   MR1, MR1, #0x3380 ; X = 1/(1*2^24) * iqx = 2.5 (0x40200000)
   MMPYF32   MR2, MR2, #0x3380 ; B = 1/(1*2^24) * iqb = -.5 (0xBF000000)
   MMPYF32   MR3, MR0, MR1     ; M*X
   MADDF32   MR2, MR2, MR3     ; Y=MX+B = 3.25 (0x40500000)
; Convert Y from float32 to IQ24
   MMPYF32 MR2, MR2, #0x4B80   ; Y * 1*2^24
   MF32TOI32 MR2, MR2          ; IQ24(Y) = 0x03400000
   MMOV32 @_Y, MR2             ; store result
   MSTOP                       ; end of task
```

See also

MF32TOUI32 MRa, MRb
MI32TOF32 MRa, MRb
MI32TOF32 MRa, mem32
MUI32TOF32 MRa, MRb
MUI32TOF32 MRa, mem32

MF32TOUI16 MRa, MRb

Convert 32-Bit Floating-Point Value to 16-bit Unsigned Integer

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1110 1010 0000
```

Description

Convert the 32-bit floating point value in MRb to an unsigned 16-bit integer value and truncate to zero. The result is stored in MRa. To instead round the integer to the nearest even value, use the MF32TOUI16R instruction.

```
MRa(15:0) = F32TOUI16(MRb);
MRa(31:16) = 0x0000;
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ      MR0, #9.0     ; MR0 = 9.0 (0x41100000)
MF32TOUI16  MR1, MR0      ; MR1(15:0) = MF32TOUI16(MR0) = 9 (0x0009)
                          ; MR1(31:16) = 0x0000
MMOVIZ      MR2, #-9.0    ; MR2 = -9.0 (0xC1100000)
MF32TOUI16  MR3, MR2      ; MR3(15:0) = MF32TOUI16(MR2) = 0 (0x0000)
                          ; MR3(31:16) = 0x0000
```

See also

MF32TOI16 MRa, MRb
MF32TOUI16 MRa, MRb
MF32TOUI16R MRa, MRb
MI16TOF32 MRa, MRb
MI16TOF32 MRa, mem16
MUI16TOF32 MRa, mem16
MUI16TOF32 MRa, MRb

MF32TOUI16R MRa, MRb

Convert 32-Bit Floating-Point Value to 16-bit Unsigned Integer and Round

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1110 1100 0000
```

Description

Convert the 32-bit floating-point value in MRb to an unsigned 16-bit integer and round to the closest even value. The result is stored in MRa. To instead truncate the converted value, use the MF32TOUI16 instruction.

```
MRa(15:0)  = MF32TOUI16round(MRb);
MRa(31:16) = 0x0000;
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ       MR0, #0x412C     ; MR0 = 0x412C
MMOVXI       MR0, #0xCCCD     ; MR0 = 0xCCCD ; MR0 = 10.8 (0x412CCCCD)
MF32TOUI16R  MR1, MR0         ; MR1(15:0) = MF32TOUI16round(MR0) = 11 (0x000B)
                              ; MR1(31:16) = 0x0000
MMOVF32      MR2, #-10.8      ; MR2 = -10.8 (0x0xC12CCCCD)
MF32TOUI16R  MR3, MR2         ; MR3(15:0) = MF32TOUI16round(MR2) = 0 (0x0000)
                              ; MR3(31:16) = 0x0000
```

See also

MF32TOI16 MRa, MRb
MF32TOI16R MRa, MRb
MF32TOUI16 MRa, MRb
MI16TOF32 MRa, MRb
MI16TOF32 MRa, mem16
MUI16TOF32 MRa, mem16
MUI16TOF32 MRa, MRb

MF32TOUI32 MRa, MRb

Convert 32-Bit Floating-Point Value to 32-Bit Unsigned Integer

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1101 1010 0000
```

Description

Convert the 32-bit floating-point value in MRb to an unsigned 32-bit integer and store the result in MRa.

```
MRa = F32TOUI32(MRb);
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ      MR0, #12.5   ; MR0 = 12.5 (0x41480000)
MF32TOUI32  MR0, MR0     ; MR0 = MF32TOUI32 (MR0) = 12 (0x0000000C)
MMOVIZ      MR1, #-6.5   ; MR1 = -6.5 (0xC0D00000)
MF32TOUI32  MR2, MR1     ; MR2 = MF32TOUI32 (MR1) = 0.0 (0x00000000)
```

See also

MF32TOI32 MRa, MRb
MI32TOF32 MRa, MRb
MI32TOF32 MRa, mem32
MUI32TOF32 MRa, MRb
MUI32TOF32 MRa, mem32

MFRACF32 MRa, MRb

Fractional Portion of a 32-Bit Floating-Point Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1110 0000 0000
```

Description Returns in MRa the fractional portion of the 32-bit floating-point value in MRb

Flags This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline This is a single-cycle instruction.

Example

```
MMOVIZ    MR2, #19.625 ; MR2 = 19.625 (0x419D0000)
MFRACF32  MR3, MR2     ; MR3 = MFRACF32(MR2) = 0.625 (0x3F200000)0)
```

MI16TOF32 MRa, MRb

Convert 16-Bit Integer to 32-Bit Floating-Point Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1110 1000 0000
```

Description

Convert the 16-bit signed integer in MRb to a 32-bit floating-point value and store the result in MRa.

```
MRa = MI16TOF32(MRb);
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ      MR0, #0x0000    ; MR0(31:16) = 0.0 (0x0000)
MMOVXI      MR0, #0x0004    ; MR0(15:0) = 4.0 (0x0004)
MI16TOF32   MR1, MR0        ; MR1 = MI16TOF32 (MR0) = 4.0 (0x40800000)
MMOVIZ      MR2, #0x0000    ; MR2(31:16) = 0.0 (0x0000)
MMOVXI      MR2, #0xFFFC    ; MR2(15:0) = -4.0 (0xFFFC)
MI16TOF32   MR3, MR2        ; MR3 = MI16TOF32 (MR2) = -4.0 (0xC0800000)
MSTOP
```

See also

MF32TOI16 MRa, MRb
MF32TOI16R MRa, MRb
MF32TOUI16 MRa, MRb
MF32TOUI16R MRa, MRb
MI16TOF32 MRa, mem16
MUI16TOF32 MRa, mem16
MUI16TOF32 MRa, MRb

MI16TOF32 MRa, mem16

Convert 16-Bit Integer to 32-Bit Floating-Point Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
mem16	16-bit source memory location to be converted

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0101 00aa addr
```

Description

Convert the 16-bit signed integer indicated by the mem16 pointer to a 32-bit floating-point value and store the result in MRa.

```
MRa = MI16TOF32[mem16];
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction:

Example

```
; Assume A = 4 (0x0004)
;        B = -4 (0xFFFC)
   MI16TOF32 MR0, @_A ; MR0 = MI16TOF32(A) = 4.0 (0x40800000)
   MI16TOF32 MR1, @_B ; MR1 = MI16TOF32(B) = -4.0 (0xC0800000
```

See also

MF32TOI16 MRa, MRb
MF32TOI16R MRa, MRb
MF32TOUI16 MRa, MRb
MF32TOUI16R MRa, MRb
MI16TOF32 MRa, MRb
MUI16TOF32 MRa, mem16
MUI16TOF32 MRa, MRb

MI32TOF32 MRa, mem32

Convert 32-Bit Integer to 32-Bit Floating-Point Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
mem32	32-bit memory source for the MMOV32 operation.

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0100 01aa addr
```

Description

Convert the 32-bit signed integer indicated by mem32 to a 32-bit floating-point value and store the result in MRa.

```
MRa = MI32TOF32[mem32];
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
; Given X, M and B are IQ24 numbers:
; X = IQ24(+2.5) = 0x02800000
; M = IQ24(+1.5) = 0x01800000
; B = IQ24(-0.5) = 0xFF800000
;
; Calculate Y = X * M + B
;
; Convert M, X, and B from IQ24 to float
;
_Cla1Task3:
   MI32TOF32 MR0, @_M          ; MR0 = 0x4BC00000
   MI32TOF32 MR1, @_X          ; MR1 = 0x4C200000
   MI32TOF32 MR2, @_B          ; MR2 = 0xCB000000
   MMPYF32 MR0, MR0, #0x3380   ; M = 1/(1*2^24) * iqm = 1.5 (0x3FC00000)
   MMPYF32 MR1, MR1, #0x3380   ; X = 1/(1*2^24) * iqx = 2.5 (0x40200000)
   MMPYF32 MR2, MR2, #0x3380   ; B = 1/(1*2^24) * iqb = -.5 (0xBF000000)
   MMPYF32 MR3, MR0, MR1       ; M*X
   MADDF32 MR2, MR2, MR3       ; Y=MX+B = 3.25 (0x40500000)
; Convert Y from float32 to IQ24
   MMPYF32 MR2, MR2, #0x4B80   ; Y * 1*2^24
   MF32TOI32 MR2, MR2          ; IQ24(Y) = 0x03400000
   MMOV32 @_Y, MR2             ; store result
   MSTOP                       ; end of task
```

See also

MF32TOI32 MRa, MRb
MF32TOUI32 MRa, MRb
MI32TOF32 MRa, MRb
MUI32TOF32 MRa, MRb
MUI32TOF32 MRa, mem32

MI32TOF32 MRa, MRb

Convert 32-Bit Integer to 32-Bit Floating-Point Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1101 1000 0000
```

Description

Convert the signed 32-bit integer in MRb to a 32-bit floating-point value and store the result in MRa.

```
MRa = MI32TOF32(MRb);
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
   MMOVIZ    MR2, #0x1111 ; MR2(31:16) = 4369 (0x1111)
   MMOVXI    MR2, #0x1111 ; MR2(15:0) = 4369 (0x1111)
                          ; MR2 = +286331153 (0x11111111)
   MI32TOF32 MR3, MR2     ; MR3 = MI32TOF32 (MR2) = 286331153.0 (0x4D888888)
```

See also

MF32TOI32 MRa, MRb
MF32TOUI32 MRa, MRb
MI32TOF32 MRa, mem32
MUI32TOF32 MRa, MRb
MUI32TOF32 MRa, mem32

MLSL32 MRa, #SHIFT

Logical Shift Left

Operands

MRa	CLA floating-point source/destination register (MR0 to MR3)
#SHIFT	Number of bits to shift (1 to 32)

Opcode

```
LSW: 0000 0000 0shi ftaa
MSW: 0111 1011 1100 0000
```

Description

Logical shift-left of MRa by the number of bits indicated. The number of bits can be 1 to 32.

```
MARa(31:0) = Logical Shift Left(MARa(31:0) by #SHIFT bits);
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified based on the integer results of the operation.

```
NF = MRa(31);
ZF = 0;
if(MRa(31:0) == 0) { ZF = 1; }
```

Pipeline

This is a single-cycle instruction.

Example

```
; Given m2 = (int32)32
;       x2 = (int32)64
;       b2 = (int32)-128
;
; Calculate:
;       m2 = m2*2
;       x2 = x2*4
;       b2 = b2*8
;
_Cla1Task3:
   MMOV32 MR0, @_m2    ; MR0 = 32 (0x00000020)
   MMOV32 MR1, @_x2    ; MR1 = 64 (0x00000040)
   MMOV32 MR2, @_b2    ; MR2 = -128 (0xFFFFFF80)
   MLSL32 MR0, #1      ; MR0 = 64 (0x00000040)
   MLSL32 MR1, #2      ; MR1 = 256 (0x00000100)
   MLSL32 MR2, #3      ; MR2 = -1024 (0xFFFFFC00)
   MMOV32 @_m2, MR0    ; Store results
   MMOV32 @_x2, MR1
   MMOV32 @_b2, MR2
   MSTOP               ; end of task
```

See also

MADD32 MRa, MRb, MRc
MASR32 MRa, #SHIFT
MAND32 MRa, MRb, MRc
MLSR32 MRa, #SHIFT
MOR32 MRa, MRb, MRc
MXOR32 MRa, MRb, MRc
MSUB32 MRa, MRb, MRc

MLSR32 MRa, #SHIFT

Logical Shift Right

Operands

MRa	CLA floating-point source/destination register (MR0 to MR3)
#SHIFT	Number of bits to shift (1 to 32)

Opcode

```
LSW: 0000 0000 0shi ftaa
MSW: 0111 1011 1000 0000
```

Description

Logical shift-right of MRa by the number of bits indicated. The number of bits can be 1 to 32. Unlike the arithmetic shift (MASR32), the logical shift does not preserve the number's sign bit. Every bit in the operand is moved the specified number of bit positions, and the vacant bit positions are filled in with zeros.

```
MARa(31:0) = Logical Shift Right(MARa(31:0) by #SHIFT bits);
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified based on the integer results of the operation.

```
NF = MRa(31);
ZF = 0;
if(MRa(31:0) == 0) { ZF = 1;}
```

Pipeline

This is a single-cycle instruction.

Example

```
; Illustrate the difference between MASR32 and MLSR32
MMOVIZ MR0, #0xAAAA    ; MR0 = 0xAAAA5555
MMOVXI MR0, #0x5555
MMOV32 MR1, MR0        ; MR1 = 0xAAAA5555
MMOV32 MR2, MR0        ; MR2 = 0xAAAA5555
MASR32 MR1, #1         ; MR1 = 0xD5552AAA
MLSR32 MR2, #1         ; MR2 = 0x55552AAA
MASR32 MR1, #1         ; MR1 = 0xEAAA9555
MLSR32 MR2, #1         ; MR2 = 0x2AAA9555
MASR32 MR1, #6         ; MR1 = 0xFFAAAA55
MLSR32 MR2, #6         ; MR2 = 0x00AAAA55
```

See also

MADD32 MRa, MRb, MRc
MASR32 MRa, #SHIFT
MAND32 MRa, MRb, MRc
MLSL32 MRa, #SHIFT
MOR32 MRa, MRb, MRc
MXOR32 MRa, MRb, MRc
MSUB32 MRa, MRb, MRc

MMACF32 MR3, MR2, MRd, MRe, MRf ||MMOV32 MRa, mem32

32-Bit Floating-Point Multiply and Accumulate with Parallel Move

Operands

MR3	floating-point destination/source register MR3 for the add operation
MR2	CLA floating-point source register MR2 for the add operation
MRd	CLA floating-point destination register (MR0 to MR3) for the multiply operation MRd cannot be the same register as MRa
MRe	CLA floating-point source register (MR0 to MR3) for the multiply operation
MRf	CLA floating-point source register (MR0 to MR3) for the multiply operation
MRa	CLA floating-point destination register for the MMOV32 operation (MR0 to MR3). MRa cannot be MR3 or the same register as MRd.
mem32	32-bit source for the MMOV32 operation

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0011 ffee ddaa addr
```

Description

Multiply and accumulate the contents of floating-point registers and move from register to memory. The destination register for the MMOV32 cannot be the same as the destination registers for the MMACF32.

```
MR3 = MR3 + MR2;
MRd = MRe * MRf;
MRa = [mem32];
```

Restrictions

The destination registers for the MMACF32 and the MMOV32 must be unique. That is, MRa cannot be MR3 and MRa cannot be the same register as MRd.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MMACF32 (add or multiply) generates an underflow condition.
- LVF = 1 if MMACF32 (add or multiply) generates an overflow condition.

MMOV32 sets the NF and ZF flags as follows:

```
NF = MRa(31);
ZF = 0;
if(MRa(30:23) == 0) { ZF = 1; NF = 0; }
```

Pipeline

MMACF32 and MMOV32 complete in a single cycle.

MMACF32 MR3, MR2, MRd, MRe, MRf ||MMOV32 MRa, mem32 (continued)

32-Bit Floating-Point Multiply and Accumulate with Parallel Move

Example 1

```
; Perform 5 multiply and accumulate operations:
;
; X and Y are 32-bit floating-point arrays
;
; 1st multiply: A = X0 * Y0
; 2nd multiply: B = X1 * Y1
; 3rd multiply: C = X2 * Y2
; 4th multiply: D = X3 * Y3
; 5th multiply: E = X3 * Y3
;
; Result = A + B + C + D + E
;
_Cla1Task1:
   MMOVI16 MAR0, #_X               ; MAR0 points to X array
   MMOVI16 MAR1, #_Y               ; MAR1 points to Y array
   MNOP                            ; Delay for MAR0, MAR1 load
   MNOP                            ; Delay for MAR0, MAR1 load
                                   ; <-- MAR0 valid
   MMOV32 MR0, *MAR0[2]++          ; MR0 = X0, MAR0 += 2
                                   ; <-- MAR1 valid
   MMOV32 MR1, *MAR1[2]++          ; MR1 = Y0, MAR1 += 2
   MMPYF32 MR2, MR0, MR1           ; MR2 = A = X0 * Y0
|| MMOV32 MR0, *MAR0[2]++          ; In parallel MR0 = X1, MAR0 += 2
   MMOV32 MR1, *MAR1[2]++          ; MR1 = Y1, MAR1 += 2
   MMPYF32 MR3, MR0, MR1           ; MR3 = B = X1 * Y1
|| MMOV32 MR0, *MAR0[2]++          ; In parallel MR0 = X2, MAR0 += 2
   MMOV32 MR1, *MAR1[2]++          ; MR1 = Y2, MAR2 += 2
   MMACF32 MR3, MR2, MR2, MR0, MR1 ; MR3 = A + B, MR2 = C = X2 * Y2
|| MMOV32 MR0, *MAR0[2]++          ; In parallel MR0 = X3
   MMOV32 MR1, *MAR1[2]++          ; MR1 = Y3 M
   MACF32 MR3, MR2, MR2, MR0, MR1  ; MR3 = (A + B) + C, MR2 = D = X3 * Y3
|| MMOV32 MR0, *MAR0               ; In parallel MR0 = X4
   MMOV32 MR1, *MAR1               ; MR1 = Y4
   MMPYF32 MR2, MR0, MR1           ; MR2 = E = X4 * Y4
|| MADDF32 MR3, MR3, MR2           ; in parallel MR3 = (A + B + C) + D
   MADDF32 MR3, MR3, MR2           ; MR3 = (A + B + C + D) + E
   MMOV32 @_Result, MR3            ; Store the result
   MSTOP                           ; end of task
```

MMACF32 MR3, MR2, MRd, MRe, MRf ||MMOV32 MRa, mem32 (continued)

32-Bit Floating-Point Multiply and Accumulate with Parallel Move

Example 2

```
; sum = X0*B0 + X1*B1 + X2*B2 + Y1*A1 + Y2*B2
;
;       X2 = X1
;       X1 = X0
;       Y2 = Y1 ; Y1 = sum
;
_ClaTask2:
     MMOV32     MR0, @_B2       ; MR0 = B2
     MMOV32     MR1, @_X2       ; MR1 = X2
     MMPYF32    MR2, MR1, MR0   ; MR2 = X2*B2
  || MMOV32     MR0, @_B1       ; MR0 = B1
     MMOVD32    MR1, @_X1       ; MR1 = X1, X2 = X1
     MMPYF32    MR3, MR1, MR0   ; MR3 = X1*B1
  || MMOV32     MR0, @_B0       ; MR0 = B0
     MMOVD32    MR1, @_X0       ; MR1 = X0, X1 = X0
;  MR3 = X1*B1 + X2*B2, MR2 = X0*B0
;  MR0 = A2
     MMACF32 MR3, MR2, MR2, MR1, MR0
  || MMOV32 MR0, @_A2 M

     MOV32 MR1, @_Y2            ; MR1 = Y2
;  MR3 = X0*B0 + X1*B1 + X2*B2, MR2 = Y2*A2
;  MR0 = A1
     MMACF32 MR3, MR2, MR2, MR1, MR0
  || MMOV32 MR0, @_A1
     MMOVD32 MR1,@_Y1           ; MR1 = Y1, Y2 = Y1
     MADDF32 MR3, MR3, MR2      ; MR3 = Y2*A2 + X0*B0 + X1*B1 + X2*B2
  || MMPYF32 MR2, MR1, MR0      ; MR2 = Y1*A1
     MADDF32 MR3, MR3, MR2      ; MR3 = Y1*A1 + Y2*A2 + X0*B0 + X1*B1 + X2*B2
     MMOV32 @_Y1, MR3           ; Y1 = MR3
     MSTOP                      ; end of task
```

See also MMPYF32 MRa, MRb, MRc || MADDF32 MRd, MRe, MRf

MMAXF32 MRa, MRb

32-Bit Floating-Point Maximum

Operands

MRa	CLA floating-point source/destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1101 0010 0000
```

Description

```
if(MRa < MRb) MRa = MRb;
```

Special cases for the output from the MMAXF32 operation:

- NaN output is converted to infinity
- A denormalized output is converted to positive zero.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The ZF and NF flags are configured on the result of the operation, not the result stored in the destination register.

```
if(MRa == MRb)  {ZF=1; NF=0;}
if(MRa > MRb) {ZF=0; NF=0;}
if(MRa < MRb) {ZF=0; NF=1;}
```

Pipeline

This is a single-cycle instruction.

Example 1

```
MMOVIZ   MR0, #5.0  ; MR0 =  5.0  (0x40A00000)
MMOVIZ   MR1, #-2.0 ; MR1 = -2.0  (0xC0000000)
MMOVIZ   MR2, #-1.5 ; MR2 = -1.5  (0xBFC00000)
MMAXF32   MR2, MR1  ; MR2 = -1.5, ZF = NF = 0
MMAXF32   MR1, MR2  ; MR1 = -1.5, ZF = 0, NF = 1
MMAXF32   MR2, MR0  ; MR2 =  5.0, ZF = 0, NF = 1
MAXF32   MR0, MR2   ; MR2 =  5.0, ZF = 1, NF = 0
```

MMAXF32 MRa, MRb (continued)

32-Bit Floating-Point Maximum

Example 2

```
; X is an array of 32-bit floating-point values
; Find the maximum value in an array X
; and store the value in Result
;
_Cla1Task1:
  MMOVI16    MAR1,#_X          ; Start address
  MUI16TOF32 MR0, @_len        ; Length of the array
  MNOP                         ; delay for MAR1 load
  MNOP                         ; delay for MAR1 load
  MMOV32     MR1, *MAR1[2]++   ; MR1 = X0
LOOP
  MMOV32     MR2, *MAR1[2]++   ; MR2 = next element
  MMAXF32    MR1, MR2          ; MR1 = MAX(MR1, MR2)
  MADDF32    MR0, MR0, #-1.0   ; Decrememt the counter
  MCMPF32    MR0 #0.0          ; Set/clear flags for MBCNDD
  MNOP
  MNOP
  MNOP
  MBCNDD     LOOP, NEQ         ; Branch if not equal to zero
  MMOV32     @_Result, MR1     ; Always executed
  MNOP                         ; Always executed
  MNOP                         ; Always executed
  MSTOP                        ; End of task
```

See also

MCMPF32 MRa, MRb
MCMPF32 MRa, #16FHi
MMAXF32 MRa, #16FHi
MMINF32 MRa, MRb
MMINF32 MRa, #16FHi

MMAXF32 MRa, #16FHi

32-Bit Floating-Point Maximum

Operands

MRa	CLA floating-point source/destination register (MR0 to MR3)
#16FHi	A 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0.

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 1001 0000 00aa
```

Description

Compare MRa with the floating-point value represented by the immediate operand. If the immediate value is larger, then load the value into MRa.

```
if(MRa < #16FHi:0) MRa = #16FHi:0;
```

#16FHi is a 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0. This addressing mode is most useful for constants where the lowest 16-bits of the mantissa are 0. Some examples are 2.0 (0x40000000), 4.0 (0x40800000), 0.5 (0x3F000000), and -1.5 (0xBFC00000). The assembler accepts either a hex or float as the immediate value. That is, -1.5 can be represented as #-1.5 or #0xBFC0.

Special cases for the output from the MMAXF32 operation:

- NaN output is converted to infinity
- A denormalized output is converted to positive zero.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The ZF and NF flags are configured on the result of the operation, not the result stored in the destination register.

```
if(MRa == #16FHi:0)  {ZF=1; NF=0;}
if(MRa > #16FHi:0) {ZF=0; NF=0;}
if(MRa < #16FHi:0) {ZF=0; NF=1;}
```

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ    MR0, #5.0  ; MR0 =  5.0  (0x40A00000)
MMOVIZ    MR1, #4.0  ; MR1 =  4.0  (0x40800000)
MMOVIZ    MR2, #-1.5 ; MR2 = -1.5  (0xBFC00000)
MMAXF32   MR0, #5.5  ; MR0 =  5.5, ZF = 0, NF = 1
MMAXF32   MR1, #2.5  ; MR1 =  4.0, ZF = 0, NF = 0
MMAXF32   MR2, #-1.0 ; MR2 = -1.0, ZF = 0, NF = 1
MMAXF32   MR2, #-1.0 ; MR2 = -1.5, ZF = 1, NF = 0
```

See also

MMAXF32 MRa, MRb
MMINF32 MRa, MRb
MMINF32 MRa, #16FHi

MMINF32 MRa, MRb

32-Bit Floating-Point Minimum

Operands

MRa	CLA floating-point source/destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1101 0100 0000
```

Description

```
if(MRa > MRb) MRa = MRb;
```

Special cases for the output from the MMINF32 operation:

- NaN output is converted to infinity
- A denormalized output is converted to positive zero.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The ZF and NF flags are configured on the result of the operation, not the result stored in the destination register.

```
if(MRa == MRb)  {ZF=1; NF=0;}
if(MRa > MRb) {ZF=0; NF=0;}
if(MRa < MRb) {ZF=0; NF=1;}
```

Pipeline

This is a single-cycle instruction.

Example 1

```
MMOVIZ MR0, #5.0  ; MR0 = 5.0 (0x40A00000)
MMOVIZ MR1, #4.0  ; MR1 = 4.0 (0x40800000)
MMOVIZ MR2, #-1.5 ; MR2 = -1.5 (0xBFC00000)
MMINF32 MR0, MR1  ; MR0 = 4.0, ZF = 0, NF = 0
MMINF32 MR1, MR2  ; MR1 = -1.5, ZF = 0, NF = 0
MMINF32 MR2, MR1  ; MR2 = -1.5, ZF = 1, NF = 0
MMINF32 MR1, MR0  ; MR2 = -1.5, ZF = 0, NF = 1
```

MMINF32 MRa, MRb (continued)

32-Bit Floating-Point Minimum

Example 2

```
;
; X is an array of 32-bit floating-point values
; Find the minimum value in an array X
; and store the value in Result
;
_Cla1Task1:
MMOVI16    MAR1,#_X          ; Start address
MUI16TOF32 MR0, @_len        ; Length of the array
MNOP                         ; delay for MAR1 load
MNOP                         ; delay for MAR1 load
MMOV32     MR1, *MAR1[2]++   ; MR1 = X0
LOOP
MMOV32     MR2, *MAR1[2]++   ; MR2 = next element
MMINF32    MR1, MR2          ; MR1 = MAX(MR1, MR2)
MADDF32    MR0, MR0, #-1.0   ; Decrememt the counter
MCMPF32    MR0 #0.0          ; Set/clear flags for MBCNDD
MNOP
MNOP
MNOP
MBCNDD     LOOP, NEQ         ; Branch if not equal to zero
MMOV32     @_Result, MR1     ; Always executed
MNOP                         ; Always executed
MNOP                         ; Always executed
MSTOP                        ; End of task
```

See also

MMAXF32 MRa, MRb
MMAXF32 MRa, #16FHi
MMINF32 MRa, #16FHi

MMINF32 MRa, #16FHi

32-Bit Floating-Point Minimum

Operands

MRa	floating-point source/destination register (MR0 to MR3)
#16FHi	A 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0.

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 1001 0100 00aa
```

Description

Compare MRa with the floating-point value represented by the immediate operand. If the immediate value is smaller, then load the value into MRa.

```
if(MRa > #16FHi:0) MRa = #16FHi:0;
```

#16FHi is a 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0. This addressing mode is most useful for constants where the lowest 16-bits of the mantissa are 0. Some examples are 2.0 (0x40000000), 4.0 (0x40800000), 0.5 (0x3F000000), and -1.5 (0xBFC00000). The assembler accepts either a hex or float as the immediate value. That is, -1.5 can be represented as #-1.5 or #0xBFC0.

Special cases for the output from the MMINF32 operation:

- NaN output is converted to infinity
- A denormalized output is converted to positive zero.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The ZF and NF flags are configured on the result of the operation, not the result stored in the destination register.

```
if(MRa == #16FHi:0)   {ZF=1; NF=0;}
if(MRa > #16FHi:0) {ZF=0; NF=0;}
if(MRa < #16FHi:0) {ZF=0; NF=1;}
```

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ   MR0, #5.0   ; MR0 =  5.0  (0x40A00000)
MMOVIZ   MR1, #4.0   ; MR1 =  4.0  (0x40800000)
MMOVIZ   MR2, #-1.5  ; MR2 = -1.5  (0xBFC00000)
MMINF32  MR0, #5.5   ; MR0 =  5.0, ZF = 0, NF = 1
MMINF32  MR1, #2.5   ; MR1 =  2.5, ZF = 0, NF = 0
MMINF32  MR2, #-1.0  ; MR2 = -1.5, ZF = 0, NF = 1
MMINF32  MR2, #-1.5  ; MR2 = -1.5, ZF = 1, NF = 0
```

See also

MMAXF32 MRa, #16FHi
MMAXF32 MRa, MRb
MMINF32 MRa, MRb

MMOV16 MARx, MRa, #16I

Load the Auxiliary Register with MRa + 16-bit Immediate Value

Operands

MARx	Auxiliary register MAR0 or MAR1
MRa	CLA Floating-point register (MR0 to MR3)
#16I	16-bit immediate value

Opcode

```
LSW: IIII IIII IIII IIII (opcode of MMOV16 MAR0, MRa, #16I)
MSW: 0111 1111 1101 00AA
LSW: IIII IIII IIII IIII (opcode of MMOV16 MAR1, MRa, #16I)
MSW: 0111 1111 1111 00AA
```

Description

Load the auxiliary register, MAR0 or MAR1, with MRa(15:0) + 16-bit immediate value. Refer to the Pipeline section for important information regarding this instruction.

```
MARx = MRa(15:0) + #16I;
```

Flags

This instruction does not modify flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction. The load of MAR0 or MAR1 occurs in the EXE phase of the pipeline. Any post increment of MAR0 or MAR1 using indirect addressing occurs in the D2 phase of the pipeline. Therefore, the following applies when loading the auxiliary registers:

- **I1 and I2**

  The two instructions following MMOV16 use MAR0 or MAR1 before the update occurs. Thus, these two instructions use the old value of MAR0 or MAR1.
- **I3**

  Loading of an auxiliary register occurs in the EXE phase while updates due to post-increment addressing occur in the D2 phase. Thus, I3 cannot use the auxiliary register or there is a conflict. In the case of a conflict, the update due to address-mode post increment wins and the auxiliary register is not updated with #_X.
- **I4**

  Starting with the 4th instruction, MAR0 or MAR1 is the new value loaded with MMOVI16.

```
; Assume MAR0 is 50, MR0 is 10, and #_X is 20
  MMOV16 MAR0, MR0, #_X        ; Load MAR0 with address of X (20) + MR0 (10)
  <Instruction 1>         ; I1 Uses the old value of MAR0 (50)
  <Instruction 2>         ; I2 Uses the old value of MAR0 (50)
  <Instruction 3>         ; I3 Cannot use MAR0
  <Instruction 4>         ; I4 Uses the new value of MAR0 (30)
  <Instruction 5>         ; I5
```

MMOV16 MARx, MRa, #16I (continued)

Load the Auxiliary Register with MRa + 16-bit Immediate Value

Table 8-16. Pipeline Activity for MMOV16 MARx, MRa , #16I

Instruction	F1	F2	D1	D2	R1	R2	E	W
MMOV16 MAR0, MR0, #_X	MMOV16							
I1	I1	MMOV16						
I2	I2	I1	MMOV16					
I3	I3	I2	I1	MMOV16				
I4	I4	I3	I2	I1	MMOV16			
I5	I5	I4	I3	I2	I1	MMOV16		
I6	I6	I5	I4	I3	I2	I1	MMOV16	

Example 1

```
; Calculate an offset into a sin/cos table
;
_Cla1Task1:
    MMOV32 MR0,@_rad                ; MR0 = rad
    MMOV32 MR1,@_TABLE_SIZEDivTwoPi ; MR1 = TABLE_SIZE/(2*Pi)
    MMPYF32 MR1,MR0,MR1             ; MR1 = rad* TABLE_SIZE/(2*Pi)
||  MMOV32 MR2,@_TABLE_MASK         ; MR2 = TABLE_MASK
    MF32TOI32 MR3,MR1               ; MR3 = K=int(rad*TABLE_SIZE/(2*Pi))
    MAND32 MR3,MR3,MR2              ; MR3 = K & TABLE_MASK
    MLSL32 MR3,#1                   ; MR3 = K * 2
    MMOV16 MAR0,MR3,#_Cos0          ; MAR0 K*2+addr of table.Cos0
    MFRACF32 MR1,MR1                ; I1
    MMOV32 MR0,@_TwoPiDivTABLE_SIZE ; I2
    MMPYF32 MR1,MR1,MR0             ; I3
||  MMOV32 MR0,@_Coef3
    MMOV32 MR2,*MAR0[#-64]++        ; MR2 = *MAR0, MAR0 += (-64)
    ...
    ...
    MSTOP ; end of task
```

MMOV16 MARx, MRa, #16I (continued)

Load the Auxiliary Register with MRa + 16-bit Immediate Value

Example 2

```
; This task logs the last NUM_DATA_POINTS
; ADCRESULT1 values in the array VoltageCLA
;
; When the last element in the array has been
; filled, the task goes back to the
; the first element.
;
; Before starting the ADC conversions, force
; Task 8 to initialize the ConversionCount to zero
;
; The ADC is set to sample (acquire) for 15 SYSCLK cycles
; or 75ns. After the capacitor has captured the analog
; value, the ADC triggers this task early.
; It takes 10.5 ADCCLKs to complete a conversion,
; the ADCCLK being SYSCLK/4
;   T_sys = 1/200MHz = 5ns
;   T_adc = 4*T_sys  = 20ns
; The ADC takes 10.5 * 4 or 42 SYSCLK cycles to complete
; a conversion. The ADC result register can be read on the
; 36th instruction after the task begins.
;
_Cla1Task2:
    .asg       0, N
    .loop
    MNOP                                    ;I1 - I28 Wait till I36 to read
result
    .eval      N + 1, N
    .break     N = 28
    .endloop
    MMOVZ16    MR0, @_ConversionCount           ;I29 Current Conversion
    MMOV16     MAR1, MR0, #_VoltageCLA          ;I30 Next array location
    MUI16TOF32 MR0, MR0                         ;I31 Convert count to float32
    MADDF32    MR0, MR0, #1.0                   ;I32 Add 1 to conversion count
    MCMPF32    MR0, #NUM_DATA_POINTS.0          ;I33 Compare count to max
    MF32TOUI16 MR0, MR0                         ;I34 Convert count to Uint16
    MNOP                                        ;I35 Wait till I36 to read
result
    MMOVZ16    MR2, @_AdcaResultRegs.ADCRESULT1 ;I36 Read ADCRESULT1
    MMOV16     *MAR1, MR2                       ; Store ADCRESULT1
    MBCNDD     _RestartCount, GEQ               ; If count >= NUM_DATA_POINTS
    MMOVIZ     MR1, #0.0                        ; Always executed: MR1=0
    MNOP
    MNOP
    MMOV16     @_ConversionCount, MR0           ; If branch not taken
    MSTOP                                       ; store current count
_RestartCount
    MMOV16     @_ConversionCount, MR1           ; If branch taken, restart
count
    MSTOP                                       ; end of task
; This task initializes the ConversionCount
; to zero
;
_Cla1Task8:
    MMOVIZ     MR0, #0.0
    MMOV16     @_ConversionCount, MR0
    MSTOP
_ClaT8End:
```

MMOV16 MARx, mem16

Load MAR1 with 16-bit Value

Operands

MARx	CLA auxiliary register MAR0 or MAR1
mem16	16-bit destination memory accessed using one of the available addressing modes

Opcode

```
LSW: mmmm mmmm mmmm mmmm (Opcode for MMOV16 MAR0, mem16)
MSW: 0111 0110 0000 addr
LSW: mmmm mmmm mmmm mmmm (Opcode for MMOV16 MAR1, mem16)
MSW: 0111 0110 0100 addr
```

Description

Load MAR0 or MAR1 with the 16-bit value pointed to by mem16. Refer to the Pipeline section for important information regarding this instruction.

```
MAR1 = [mem16];
```

Flags

No flags MSTF flags are affected.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction. The load of MAR0 or MAR1 occurs in the EXE phase of the pipeline. Any post increment of MAR0 or MAR1 using indirect addressing occurs in the D2 phase of the pipeline. Therefore, the following applies when loading the auxiliary registers:

- **I1 and I2**

  The two instructions following MMOV16 use MAR0 or MAR1 before the update occurs. Thus, these two instructions use the old value of MAR0 or MAR1.
- **I3**

  Loading of an auxiliary register occurs in the EXE phase while updates due to post-increment addressing occur in the D2 phase. Thus, I3 cannot use the auxiliary register or there is a conflict. In the case of a conflict, the update due to address-mode post increment, the auxiliary register is not updated with #_X.
- **I4**

  Starting with the 4th instruction, MAR0 or MAR1 is the new value loaded with MMOV16.

```
; Assume MAR0 is 50 and @_X is 20
MMOV16 MAR0, @_X    ; Load MAR0 with the contents of X (20)
<Instruction 1>     ; I1 Uses the old value of MAR0 (50)
<Instruction 2>     ; I2 Uses the old value of MAR0 (50)
<Instruction 3>     ; I3 Cannot use MAR0
<Instruction 4>     ; I4 Uses the new value of MAR0 (20)
<Instruction 5>     ; I5
....
```

MMOV16 MARx, mem16 (continued)

Load MAR1 with 16-bit Value

Table 8-17. Pipeline Activity for MMOV16 MAR0/MAR1, mem16

Instruction	F1	F2	D1	D2	R1	R2	E	W
MMOV16 MAR0, @_X	MMOV16							
I1	I1	MMOV16						
I2	I2	I1	MMOV16					
I3	I3	I2	I1	MMOV16				
I4	I4	I3	I2	I1	MMOV16			
I5	I5	I4	I3	I2	I1	MMOV16		
I6	I6	I5	I4	I3	I2	I1	MMOV16	

MMOV16 MARx, mem16 (continued)

Load MAR1 with 16-bit Value

Example

```
; This task logs the last NUM_DATA_POINTS
; ADCRESULT1 values in the array VoltageCLA
;
; When the last element in the array has been
; filled, the task goes back to the
; the first element.
;
; Before starting the ADC conversions, force
; Task 8 to initialize the ConversionCount to zero
;
; The ADC is set to sample (acquire) for 15 SYSCLK cycles
; or 75ns. After the capacitor has captured the analog
; value, the ADC triggers this task early.
; It takes 10.5 ADCCLKs to complete a conversion,
; the ADCCLK being SYSCLK/4
;   T_sys = 1/200MHz = 5ns
;   T_adc = 4*T_sys  = 20ns
; The ADC takes 10.5 * 4 or 42 SYSCLK cycles to complete
; a conversion. The ADC result register can be read on the
; 36th instruction after the task begins.
;
_Cla1Task2:
    .asg       0, N
    .loop
    MNOP
;I1 - I28 Wait till I36 to read result
    .eval      N + 1, N
    .break     N = 28
    .endloop
    MMOVZ16    MR0, @_ConversionCount              ;I29 Current Conversion
    MMOV16     MAR1, MR0, #_VoltageCLA             ;I30 Next array location
    MUI16TOF32 MR0, MR0                            ;I31 Convert count to float32
    MADDF32    MR0, MR0, #1.0                      ;I32 Add 1 to conversion count
    MCMPF32    MR0, #NUM_DATA_POINTS.0             ;I33 Compare count to max
    MF32TOUI16 MR0, MR0                            ;I34 Convert count to Uint16
    MNOP                                           ;I35 Wait until I36 to read
result
    MMOVZ16    MR2, @_AdcaResultRegs.ADCRESULT1    ;I36 Read ADCRESULT1
    MMOV16     *MAR1, MR2                          ; Store ADCRESULT1
    MBCNDD     _RestartCount, GEQ                  ; If count >= NUM_DATA_POINTS
    MMOVIZ     MR1, #0.0                           ; Always executed: MR1=0
    MNOP
    MNOP
    MMOV16     @_ConversionCount, MR0              ; If branch not taken
    MSTOP                                          ; store current count
_RestartCount
    MMOV16     @_ConversionCount, MR1              ; If branch taken, restart
count
    MSTOP                                          ; end of task
; This task initializes the ConversionCount
; to zero
;
_Cla1Task8:
    MMOVIZ     MR0, #0.0
    MMOV16     @_ConversionCount, MR0
    MSTOP
_ClaT8End:
```

MMOV16 mem16, MARx

Move 16-Bit Auxiliary Register Contents to Memory

Operands

mem16	16-bit destination memory accessed using one of the available addressing modes
MARx	CLA auxiliary register MAR0 or MAR1

Opcode

```
LSW: mmmm mmmm mmmm mmmm (Opcode for MMOV16 mem16, MAR0)
MSW: 0111 0110 1000 addr
LSW: mmmm mmmm mmmm mmmm (Opcode for MMOV16 mem16, MAR1)
MSW: 0111 0110 1100 addr
```

Description

Store the contents of MAR0 or MAR1 in the 16-bit memory location pointed to by mem16.

```
[mem16] = MAR0;
```

Flags

No flags MSTF flags are affected.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

MMOV16 mem16, MRa

Move 16-Bit Floating-Point Register Contents to Memory

Operands

mem16	16-bit destination memory accessed using one of the available addressing modes
MRa	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0101 11aa addr
```

Description

Move 16-bit value from the lower 16-bits of the floating-point register (MRa(15:0)) to the location pointed to by mem16.

```
[mem16] = MRa(15:0);
```

Flags

No flags MSTF flags are affected.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

MMOV16 mem16, MRa (continued)

Move 16-Bit Floating-Point Register Contents to Memory

Example

```
; This task logs the last NUM_DATA_POINTS
; ADCRESULT1 values in the array VoltageCLA
;
; When the last element in the array has been
; filled, the task goes back to the
; the first element.
;
; Before starting the ADC conversions, force
; Task 8 to initialize the ConversionCount to zero
;
; The ADC is set to sample (acquire) for 15 SYSCLK cycles
; or 75ns. After the capacitor has captured the analog
; value, the ADC triggers this task early.
; It takes 10.5 ADCCLKs to complete a conversion,
; the ADCCLK being SYSCLK/4
;   T_sys = 1/200MHz = 5ns
;   T_adc = 4*T_sys  = 20ns
; The ADC takes 10.5 * 4 or 42 SYSCLK cycles to complete
; a conversion. The ADC result register can be read on the
; 36th instruction after the task begins.
;
_Cla1Task2:
    .asg       0, N
    .loop
    MNOP
;I1 - I28 Wait till I36 to read result
    .eval      N + 1, N
    .break     N = 28
    .endloop
    MMOVZ16    MR0, @_ConversionCount              ;I29 Current Conversion
    MMOV16     MAR1, MR0, #_VoltageCLA             ;I30 Next array location
    MUI16TOF32 MR0, MR0                            ;I31 Convert count to float32
    MADDF32    MR0, MR0, #1.0                      ;I32 Add 1 to conversion count
    MCMPF32    MR0, #NUM_DATA_POINTS.0             ;I33 Compare count to max
    MF32TOUI16 MR0, MR0                            ;I34 Convert count to Uint16
    MNOP                                           ;I35 Wait till I36 to read
result
    MMOVZ16    MR2, @_AdcaResultRegs.ADCRESULT1    ;I36 Read ADCRESULT1
    MMOV16     *MAR1, MR2                          ; Store ADCRESULT1
    MBCNDD     _RestartCount, GEQ                  ; If count >= NUM_DATA_POINTS
    MMOVIZ     MR1, #0.0                           ; Always executed: MR1=0
    MNOP
    MNOP
    MMOV16     @_ConversionCount, MR0              ; If branch not taken
    MSTOP                                          ; store current count
_RestartCount
    MMOV16     @_ConversionCount, MR1              ; If branch taken, restart
count
    MSTOP                                          ; end of task
; This task initializes the ConversionCount
; to zero
;
_Cla1Task8:
    MMOVIZ     MR0, #0.0
    MMOV16     @_ConversionCount, MR0
    MSTOP
_ClaT8End:
```

See also

MMOVIZ MRa, #16FHi
MMOVXI MRa, #16FLoHex

MMOV32 mem32, MRa

Move 32-Bit Floating-Point Register Contents to Memory

Operands

MRa	floating-point register (MR0 to MR3)
mem32	32-bit destination memory accessed using one of the available addressing modes

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0100 11aa addr
```

Description

Move from MRa to 32-bit memory location indicated by mem32.

```
[mem32] = MRa;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

No flags affected.

Pipeline

This is a single-cycle instruction.

Example

```
; Perform 5 multiply and accumulate operations:
;
; X and Y are 32-bit floating-point arrays;
; 1st multiply: A = X0 * Y0
; 2nd multiply: B = X1 * Y1
; 3rd multiply: C = X2 * Y2
; 4th multiply: D = X3 * Y3
; 5th multiply: E = X3 * Y3;
; Result = A + B + C + D + E
;
_Cla1Task1:
   MMOVI16       MAR0, #_X               ; MAR0 points to X array
   MMOVI16       MAR1, #_Y               ; MAR1 points to Y array
   MNOP                                  ; Delay for MAR0, MAR1 load
   MNOP                                  ; Delay for MAR0, MAR1 load
                                         ; <-- MAR0 valid
   MMOV32        MR0, *MAR0[2]++         ; MR0 = X0, MAR0 += 2
                                         ; <-- MAR1 valid
   MMOV32        MR1, *MAR1[2]++         ; MR1 = Y0, MAR1 += 2
   MMPYF32       MR2,  MR0, MR1          ; MR2 = A = X0 * Y0
|| MMOV32        MR0, *MAR0[2]++         ; In parallel MR0 = X1, MAR0 += 2
   MMOV32        MR1, *MAR1[2]++         ; MR1 = Y1, MAR1 += 2
   MMPYF32       MR3, MR0, MR1           ; MR3 = B = X1 * Y1
|| MMOV32        MR0, *MAR0[2]++         ; In parallel MR0 = X2, MAR0 += 2
   MMOV32        MR1, *MAR1[2]++         ; MR1 = Y2, MAR2 += 2
   MMACF32       MR3, MR2, MR2, MR0, MR1 ; MR3 = A + B, MR2 = C = X2 * Y2
|| MMOV32        MR0, *MAR0[2]++         ; In parallel MR0 = X3
   MMOV32        MR1, *MAR1[2]++         ; MR1 = Y3
   MMACF32       MR3, MR2, MR2, MR0, MR1 ; MR3 = (A + B) + C, MR2 = D = X3 *
Y3
|| MMOV32        MR0, *MAR0              ; In parallel MR0 = X4
   MMOV32        MR1, *MAR1              ; MR1 = Y4
   MMPYF32       MR2, MR0, MR1           ; MR2 = E = X4 * Y4
|| MADDF32       MR3, MR3, MR2           ; in parallel MR3 = (A + B + C) + D
   MADDF32       MR3, MR3, MR2           ; MR3 = (A + B + C + D) + E
   MMOV32        @_Result, MR3           ; Store the result MSTOP ; end of task
```

See also

MMOV32 mem32, MSTF

MMOV32 mem32, MSTF

Move 32-Bit MSTF Register to Memory

Operands

MSTF	Floating-point status register
mem32	32-bit destination memory

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0111 0100 addr
```

Description

Copy the CLA floating-point status register, MSTF, to memory.

```
[mem32] = MSTF;
```

Flags

This instruction does not modify flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

One of the uses of this instruction is to save off the return PC (RPC) prior to calling a function. The decision to jump to a function is made when the MCCNDD is in the decode2 (D2) phase of the pipeline; the RPC is also updated in this phase. The actual jump occurs 3 cycles later when MCCNDD enters the execution (E) phase. You must save the old RPC before MCCNDD updates in the D2 phase; that is, save MSTF 3 instructions prior to the function call.

Example

The following example illustrates the pipeline flow for the context save (of the flags and RPC) prior to a function call. The first column in the comments shows the pipeline stages for the MMOV32 instruction while the second column pertains to the MCCNDD instruction.

```
 MMOV32 @_temp, MSTF ; D2|   |
MNOP                 ; R1|F1| MCCNDD is fetched
MNOP                 ; R2|F2|
MNOP                 ; E |D1|
MCCNDD _bar, UNC     ; W |D2| old RPC written to memory,
                     ;   |  | RPC updated with MPC+1
MNOP                 ;   |R1|
MNOP                 ;   |R2|
MNOP                 ;   |E | execution branches to _bar
```

See also

MMOV32 mem32, MRa

MMOV32 MRa, mem32 {, CNDF}

Conditional 32-Bit Move

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
mem32	32-bit memory location accessed using one of the available addressing modes
CNDF	Optional condition

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 00cn dfaa addr
```

Description

If the condition is true, then move the 32-bit value referenced by mem32 to the floating-point register indicated by MRa.

```
if (CNDF == TRUE) MRa = [mem32];
```

CNDF is one of the following conditions:

Encode(1)	CNDF	Description	MSTF Flags Tested
0000	NEQ	Not equal to zero	ZF == 0
0001	EQ	Equal to zero	ZF == 1
0010	GT	Greater than zero	ZF == 0 AND NF == 0
0011	GEQ	Greater than or equal to zero	NF == 0
0100	LT	Less than zero	NF == 1
0101	LEQ	Less than or equal to zero	ZF == 1 OR NF == 1
1010	TF	Test flag set	TF == 1
1011	NTF	Test flag not set	TF == 0
1100	LU	Latched underflow	LUF == 1
1101	LV	Latched overflow	LVF == 1
1110	UNC	Unconditional	None
1111	UNCF(2)	Unconditional with flag modification	None

(1) Values not shown are reserved.
(2) This is the default operation, if no CNDF field is specified. This condition allows the ZF and NF flags to be modified when a conditional operation is executed. All other conditions do not modify these flags.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

```
if(CNDF == UNCF)
{
  NF = MRa(31);
  ZF = 0;
  if(MRa(30:23) == 0) { ZF = 1; NF = 0; }
}
  else No flags modified;
```

Pipeline

This is a single-cycle instruction.

MMOV32 MRa, mem32 {, CNDF} (continued)

Conditional 32-Bit Move

Example

```
; Given A, B, X, M1 and M2 are 32-bit floating-point numbers
;
; if(A == B) calculate Y = X*M1
; if(A! = B) calculate Y = X*M2
;
_Cla1Task5:
    MMOV32    MR0, @_A
    MMOV32    MR1, @_B
    MCMPF32   MR0, MR1
    MMOV32    MR2, @_M1, EQ  ; if A == B, MR2 = M1
                             ;     Y = M1*X
    MMOV32    MR2, @_M2, NEQ ; if A! = B, MR2 = M2
                             ;     Y = M2*X
    MMOV32    MR3, @_X
    MMPYF32   MR3, MR2, MR3  ; Calculate Y
    MMOV32    @_Y, MR3       ; Store Y
    MSTOP                    ; end of task
```

See also

MMOV32 MRa, MRb {, CNDF}
MMOVD32 MRa, mem32

MMOV32 MRa, MRb {, CNDF}

Conditional 32-Bit Move

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)
CNDF	Optional condition

Opcode

```
LSW: 0000 0000 cndf bbaa
MSW: 0111 1010 1100 0000
```

Description

If the condition is true, then move the 32-bit value in MRb to the floating-point register indicated by MRa.

```
if (CNDF == TRUE) MRa = MRb;
```

CNDF is one of the following conditions:

Encode(1)	CNDF	Description	MSTF Flags Tested
0000	NEQ	Not equal to zero	ZF == 0
0001	EQ	Equal to zero	ZF == 1
0010	GT	Greater than zero	ZF == 0 AND NF == 0
0011	GEQ	Greater than or equal to zero	NF == 0
0100	LT	Less than zero	NF == 1
0101	LEQ	Less than or equal to zero	ZF == 1 OR NF == 1
1010	TF	Test flag set	TF == 1
1011	NTF	Test flag not set	TF == 0
1100	LU	Latched underflow	LUF == 1
1101	LV	Latched overflow	LVF == 1
1110	UNC	Unconditional	None
1111	UNCF(2)	Unconditional with flag modification	None

(1) Values not shown are reserved.
(2) This is the default operation, if no CNDF field is specified. This condition allows the ZF, and NF flags to be modified when a conditional operation is executed. All other conditions do not modify these flags.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

```
if(CNDF == UNCF)
 {
   NF = MRa(31); ZF = 0;
   if(MRa(30:23) == 0) {ZF = 1; NF = 0;}
}
else No flags modified;
```

Pipeline

This is a single-cycle instruction.

MMOV32 MRa, MRb {, CNDF} (continued)

Conditional 32-Bit Move

Example

```
; Given: X = 8.0
;        Y = 7.0
;        A = 2.0
;        B = 5.0
; _ClaTask1
    MMOV32   MR3, @_X      ; MR3 = X = 8.0
    MMOV32   MR0, @_Y      ; MR0 = Y = 7.0
    MMAXF32  MR3, MR0      ; ZF = 0, NF = 0, MR3 = 8.0
    MMOV32   MR1, @_A, GT  ; true, MR1 = A = 2.0
    MMOV32   MR1, @_B, LT  ; false, does not load MR1
    MMOV32   MR2, MR1, GT  ; true, MR2 = MR1 = 2.0
    MMOV32   MR2, MR0, LT  ; false, does not load MR2
    MSTOP
```

See also MMOV32 MRa, mem32 {,CNDF}

MMOV32 MSTF, mem32

Move 32-Bit Value from Memory to the MSTF Register

Operands

MSTF	CLA status register
mem32	32-bit source memory location

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0111 0000 addr
```

Description

Move from memory to the CLA's status register MSTF. This instruction is most useful when nesting function calls (using MCCNDD).

```
MSTF = [mem32];
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	Yes	Yes	Yes	Yes	Yes

Loading the status register can overwrite all flags and the RPC field. The MEALLOW field is not affected.

Pipeline

This is a single-cycle instruction.

See also

MMOV32 mem32, MSTF

MMOVD32 MRa, mem32

Move 32-Bit Value from Memory with Data Copy

Operands

MRa	CLA floating-point register (MR0 to MR3)
mem32	32-bit memory location accessed using one of the available addressing modes

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0100 00aa addr
```

Description

Move the 32-bit value referenced by mem32 to the floating-point register indicated by MRa.

```
MRa = [mem32];
[mem32+2] = [mem32];
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

```
NF = MRa(31);
ZF = 0;
if(MRa(30:23) == 0){ ZF = 1; NF = 0; }
```

Pipeline

This is a single-cycle instruction.

Example

```
; sum = X0*B0 + X1*B1 + X2*B2 + Y1*A1 + Y2*B2
;
;       X2 = X1
;       X1 = X0
;       Y2 = Y1
;       Y1 = sum
;
_Cla1Task2:
    MMOV32 MR0, @_B2        ; MR0 = B2
    MMOV32 MR1, @_X2        ; MR1 = X2
    MMPYF32 MR2, MR1, MR0   ; MR2 = X2*B2
||  MMOV32 MR0, @_B1        ; MR0 = B1
    MMOVD32 MR1, @_X1       ; MR1 = X1, X2 = X1
    MMPYF32 MR3, MR1, MR0   ; MR3 = X1*B1
||  MMOV32 MR0, @_B0        ; MR0 = B0
    MMOVD32 MR1, @_X0       ; MR1 = X0, X1 = X0
; MR3 = X1*B1 + X2*B2, MR2 = X0*B0
; MR0 = A2
    MMACF32 MR3, MR2, MR2, MR1, MR0
||  MMOV32 MR0, @_A2

    MMOV32 MR1, @_Y2        ; MR1 = Y2
; MR3 = X0*B0 + X1*B1 + X2*B2, MR2 = Y2*A2
; MR0 = A1
    MMACF32 MR3, MR2, MR2, MR1, MR0
||  MMOV32 MR0, @_A1
    MMOVD32 MR1,@_Y1        ; MR1 = Y1, Y2 = Y1
    MADDF32 MR3, MR3, MR2   ; MR3 = Y2*A2 + X0*B0 + X1*B1 + X2*B2
||  MMPYF32 MR2, MR1, MR0   ; MR2 = Y1*A1
    MADDF32 MR3, MR3, MR2   ; MR3 = Y1*A1 + Y2*A2 + X0*B0 + X1*B1 + X2*B2
    MMOV32 @_Y1, MR3        ; Y1 = MR3
    MSTOP                   ; end of task
```

MMOVD32 MRa, mem32 (continued)

Move 32-Bit Value from Memory with Data Copy

See also MMOV32 MRa, mem32 {,CNDF}

MMOVF32 MRa, #32F

Load the 32-Bits of a 32-Bit Floating-Point Register

Operands

This instruction is an alias for the MMOVIZ and MMOVXI instructions. The second operand is translated by the assembler such that the instruction becomes:

```
MMOVIZ MRa, #16FHiHex MMOVXI MRa, #16FLoHex
```

MRa	CLA floating-point destination register (MR0 to MR3)
#32F	Immediate float value represented in floating-point representation

Opcode

```
LSW: IIII IIII IIII IIII (opcode of MMOVIZ MRa, #16FHiHex)
MSW: 0111 1000 0100 00aa
LSW: IIII IIII IIII IIII (opcode of MMOVXI MRa, #16FLoHex)
MSW: 0111 1000 1000 00aa
```

Description

This instruction accepts the immediate operand only in floating-point representation. To specify the immediate value as a hex value (IEEE 32-bit floating- point format), use the MOVI32 MRa, #32FHex instruction.

Load the 32-bits of MRa with the immediate float value represented by #32F.

#32F is a float value represented in floating-point representation. The assembler only accepts a float value represented in floating-point representation. That is, 3.0 can only be represented as #3.0 (#0x40400000 results in an error).

```
MRa = #32F;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

Depending on #32F, this instruction takes one or two cycles. If all of the lower 16-bits of the IEEE 32-bit floating-point format of #32F are zeros, then the assembler converts MMOVF32 into only an MMOVIZ instruction. If the lower 16-bits of the IEEE 32-bit floating-point format of #32F are not zeros, then the assembler converts MMOVF32 into MMOVIZ and MMOVXI instructions.

Example

```
MMOVF32 MR1, #3.0     ; MR1 = 3.0 (0x40400000)
                      ; Assembler converts this instruction as
                      ; MMOVIZ MR1, #0x4040
MMOVF32 MR2, #0.0     ; MR2 = 0.0 (0x00000000)
                      ; Assembler converts this instruction as
                      ; MMOVIZ MR2, #0x0
MMOVF32 MR3, #12.265  ; MR3 = 12.625 (0x41443D71)
                      ; Assembler converts this instruction as
                      ; MMOVIZ MR3, #0x4144
                      ; MMOVXI MR3, #0x3D71
```

See also

MMOVIZ MRa, #16FHi
MMOVXI MRa, #16FLoHex
MMOVI32 MRa, #32FHex

MMOVI16 MARx, #16I

Load the Auxiliary Register with the 16-Bit Immediate Value

Operands

MARx	Auxiliary register MAR0 or MAR1
#16I	16-bit immediate value

Opcode

```
LSW: IIII IIII IIII IIII (opcode of MMOVI16 MAR0, #16I)
MSW: 0111 1111 1100 0000
LSW: IIII IIII IIII IIII (opcode of MMOVI16 MAR1, #16I)
MSW: 0111 1111 1110 0000
```

Description

Load the auxiliary register, MAR0 or MAR1, with a 16-bit immediate value. Refer to the Pipeline section for important information regarding this instruction.

```
MARx = #16I;
```

Flags

This instruction does not modify flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction. The immediate load of MAR0 or MAR1 occurs in the EXE phase of the pipeline. Any post increment of MAR0 or MAR1 using indirect addressing occurs in the D2 phase of the pipeline. Therefore, the following applies when loading the auxiliary registers:

- **I1 and I2**

  The two instructions following MMOVI16 use MAR0 or MAR1 before the update occurs. Thus, these two instructions use the old value of MAR0 or MAR1.
- **I3**

  Loading of an auxiliary register occurs in the EXE phase while updates due to post-increment addressing occur in the D2 phase. Thus, I3 cannot use the auxiliary register or there is a conflict. In the case of a conflict, the update due to address-mode post increment, the auxiliary register is not updated with #_X.
- **I4**

  Starting with the 4th instruction, MAR0 or MAR1 is the new value loaded with MMOVI16.

```
;  Assume MAR0 is 50 and #_X is 20
MMOVI16 MAR0, #_X             ; Load MAR0 with address of X (20)
<Instruction 1>          ; I1 Uses the old value of MAR0 (50)
<Instruction 2>          ; I2 Uses the old value of MAR0 (50)
<Instruction 3>          ; I3 Cannot use MAR0
<Instruction 4>          ; I4 Uses the new value of MAR0 (20)
<Instruction 5>          ; I5
....
```

MMOVI16 MARx, #16I (continued)

Load the Auxiliary Register with the 16-Bit Immediate Value

Table 8-18. Pipeline Activity for MMOVI16 MAR0/MAR1, #16I

Instruction	F1	F2	D1	D2	R1	R2	E	W
MMOVI16 MAR0, #_X	MMOVI16							
I1	I1	MMOVI16						
I2	I2	I1	MMOVI16					
I3	I3	I2	I1	MMOVI16				
I4	I4	I3	I2	I1	MMOVI16			
I5	I5	I4	I3	I2	I1	MMOVI16		
I6	I6	I5	I4	I3	I2	I1	MMOVI16	

MMOVI32 MRa, #32FHex

Load the 32-Bits of a 32-Bit Floating-Point Register with the Immediate

Operands

MRa	Floating-point register (MR0 to MR3)
#32FHex	A 32-bit immediate value that represents an IEEE 32-bit floating-point value.

This instruction is an alias for the MMOVIZ and MMOVXI instructions. The second operand is translated by the assembler such that the instruction becomes:

```
MMOVIZ MRa, #16FHiHex
MMOVXI MRa, #16FLoHex
```

Opcode

```
LSW: IIII IIII IIII IIII (opcode of MMOVIZ MRa, #16FHiHex)
MSW: 0111 1000 0100 00aa
LSW: IIII IIII IIII IIII (opcode of MMOVXI MRa, #16FLoHex)
MSW: 0111 1000 1000 00aa
```

Description

This instruction only accepts a hex value as the immediate operand. To specify the immediate value with a floating-point representation, use the MMOVF32 MRa, #32F instruction.

Load the 32-bits of MRa with the immediate 32-bit hex value represented by #32FHex.

#32FHex is a 32-bit immediate hex value that represents the IEEE 32-bit floating-point value of a floating-point number. The assembler only accepts a hex immediate value. That is, 3.0 can only be represented as #0x40400000 (#3.0 results in an error).

```
MRa = #32FHex;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

Depending on #32FHex, this instruction takes one or two cycles. If all of the lower 16-bits of #32FHex are zeros, then the assembler converts MOVI32 to an MMOVIZ instruction. If the lower 16-bits of #32FHex are not zeros, then the assembler converts MOVI32 to MMOVIZ and MMOVXI instructions.

Example

```
MOVI32   MR1, #0x40400000 ; MR1 = 0x40400000
                          ; Assembler converts this instruction as
                          ; MMOVIZ MR1, #0x4040
MOVI32  MR2, #0x00000000  ; MR2 = 0x00000000
                          ; Assembler converts this instruction as
                          ; MMOVIZ MR2, #0x0
MOVI32  MR3, #0x40004001  ; MR3 = 0x40004001
                          ; Assembler converts this instruction as
                          ; MMOVIZ MR3, #0x4000
                          ; MMOVXI MR3, #0x4001
MOVI32  MR0, #0x00004040  ; MR0 = 0x00004040
                          ; Assembler converts this instruction as
                          ; MMOVIZ MR0, #0x0000
                          ; MMOVXI MR0, #0x4040
```

MMOVI32 MRa, #32FHex (continued)

Load the 32-Bits of a 32-Bit Floating-Point Register with the Immediate

See also

MMOVIZ MRa, #16FHi
MMOVXI MRa, #16FLoHex
MMOVF32 MRa, #32F

MMOVIZ MRa, #16FHi

Load the Upper 16-Bits of a 32-Bit Floating-Point Register

Operands

MRa	Floating-point register (MR0 to MR3)
#16FHi	A 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0.

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 1000 0100 00aa
```

Description

Load the upper 16-bits of MRa with the immediate value #16FHi and clear the low 16-bits of MRa.

#16FHi is a 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0. The assembler only accepts a decimal or hex immediate value. That is, -1.5 can be represented as #-1.5 or #0xBFC0.

By itself, MMOVIZ is useful for loading a floating-point register with a constant in which the lowest 16-bits of the mantissa are 0. Some examples are 2.0 (0x40000000), 4.0 (0x40800000), 0.5 (0x3F000000), and -1.5 (0xBFC00000). If a constant requires all 32-bits of a floating-point register to be initialized, then use MMOVIZ along with the MMOVXI instruction.

```
MRa(31:16) = #16FHi;
MRa(15:0)  = 0;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
; Load MR0 and MR1 with -1.5 (0xBFC00000)
     MMOVIZ    MR0, #0xBFC0      ; MR0 = 0xBFC00000 (1.5)
     MMOVIZ    MR1, #-1.5        ; MR1 = -1.5 (0xBFC00000)
; Load MR2 with pi = 3.141593 (0x40490FDB)
     MMOVIZ    MR2, #0x4049      ; MR2 = 0x40490000
     MMOVXI    MR2, #0x0FDB      ; MR2 = 0x40490FDB
```

See also

MMOVF32 MRa, #32F
MMOVI32 MRa, #32FHex
MMOVXI MRa, #16FLoHex

MMOVZ16 MRa, mem16

Load MRx with 16-Bit Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
mem16	16-bit source memory location

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0101 10aa addr
```

Description

Move the 16-bit value referenced by mem16 to the floating-point register indicated by MRa.

```
MRa(31:16) = 0;
MRa(15:0) = [mem16];
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified based on the integer results of the operation.

```
NF = 0;
if (MRa(31:0)== 0) { ZF = 1; }
```

Pipeline

This is a single-cycle instruction.

MMOVXI MRa, #16FLoHex

Move Immediate Value to the Lower 16-Bits of a Floating-Point Register

Operands

MRa	CLA floating-point register (MR0 to MR3)
#16FLoHex	A 16-bit immediate hex value that represents the lower 16-bits of an IEEE 32-bit floating-point value. The upper 16-bits are not modified.

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 1000 1000 00aa
```

Description

Load the lower 16-bits of MRa with the immediate value #16FLoHex. #16FLoHex represents the lower 16-bits of an IEEE 32-bit floating-point value. The upper 16-bits of MRa are not modified. MMOVXI can be combined with the MMOVIZ instruction to initialize all 32-bits of a MRa register.

```
MRa(15:0) = #16FLoHex;
MRa(31:16) = Unchanged;
```

Flags

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
; Load MR0 with pi = 3.141593 (0x40490FDB)
    MMOVIZ      MR0,#0x4049   ; MR0 = 0x40490000
    MMOVXI      MR0,#0x0FDB   ; MR0 = 0x40490FDB
```

See also

MMOVIZ MRa, #16FHi

MMPYF32 MRa, MRb, MRc

32-Bit Floating-Point Multiply

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)
MRc	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 00cc bbaa
MSW: 0111 1100 0000 0000
```

Description

Multiply the contents of two floating-point registers.

```
MRa = MRb * MRc;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MMPYF32 generates an underflow condition.
- LVF = 1 if MMPYF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

Example

```
; Calculate Num/Den using a Newton-Raphson algorithum for 1/Den
; Ye = Estimate(1/X)
; Ye = Ye*(2.0 - Ye*X)
; Ye = Ye*(2.0 - Ye*X)
;
_Cla1Task1:
    MMOV32     MR1, @_Den       ; MR1 = Den
    MEINVF32   MR2, MR1         ; MR2 = Ye = Estimate(1/Den)
    MMPYF32    MR3, MR2, MR1    ; MR3 = Ye*Den
    MSUBF32    MR3, #2.0, MR3   ; MR3 = 2.0 - Ye*Den
    MMPYF32    MR2, MR2, MR3    ; MR2 = Ye = Ye*(2.0 - Ye*Den)
    MMPYF32    MR3, MR2, MR1    ; MR3 = Ye*Den
||  MMOV32     MR0, @_Num       ; MR0 = Num
    MSUBF32    MR3, #2.0, MR3   ; MR3 = 2.0 - Ye*Den
    MMPYF32    MR2, MR2, MR3    ; MR2 = Ye = Ye*(2.0 - Ye*Den)
||  MMOV32     MR1, @_Den       ; Reload Den To Set Sign
    MNEGF32    MR0, MR0, EQ     ; if(Den == 0.0) Change Sign Of Num
    MMPYF32    MR0, MR2, MR0    ; MR0 = Y = Ye*Num
    MMOV32     @_Dest, MR0      ; Store result
    MSTOP                       ; end of task
```

See also

MMPYF32 MRa, #16FHi, MRb
MMPYF32 MRa, MRb, MRc || MADDF32 MRd, MRe, MRf
MMPYF32 MRd, MRe, MRf || MMOV32 MRa, mem32
MMPYF32 MRd, MRe, MRf || MMOV32 mem32, MRa
MMPYF32 MRa, MRb, MRc || MSUBF32 MRd, MRe, MRf
MMACF32 MR3, MR2, MRd, MRe, MRf || MMOV32 MRa, mem32

MMPYF32 MRa, #16FHi, MRb

32-Bit Floating-Point Multiply

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
#16FHi	A 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The lower 16-bits of the mantissa are assumed to be all 0.
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 0111 1000 bbaa
```

Description

Multiply MRb with the floating-point value represented by the immediate operand. Store the result of the addition in MRa.

#16FHi is a 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The lower 16-bits of the mantissa are assumed to be all 0. #16FHi is most useful for representing constants where the lowest 16-bits of the mantissa are 0. Some examples are 2.0 (0x40000000), 4.0 (0x40800000), 0.5 (0x3F000000), and -1.5 (0xBFC00000). The assembler accepts either a hex or float as the immediate value. That is, the value -1.5 can be represented as #-1.5 or #0xBFC0.

```
MRa = MRb * #16FHi:0;
```

This instruction can also be written as MMPYF32 MRa, MRb, #16FHi.

Flags

This instruction modifies the following flags in the MSTF register:.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MMPYF32 generates an underflow condition.
- LVF = 1 if MMPYF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

Example 1

```
; Same as example 2 but #16FHi is represented in float
    MMOVIZ    MR3, #2.0       ; MR3 = 2.0 (0x40000000)
    MMPYF32   MR0, #3.0, MR3 ; MR0 = 3.0 * MR3 = 6.0 (0x40C00000)
    MMOV32    @_X, MR0        ; Save the result in variable X
```

Example 2

```
; Same as example 1 but #16FHi is represented in Hex
    MMOVIZ    MR3, #2.0          ; MR3 = 2.0 (0x40000000)
    MMPYF32   MR0, #0x4040, MR3 ; MR0 = 0x4040 * MR3 = 6.0 (0x40C00000)
    MMOV32    @_X, MR0           ; Save the result in variable X
```

MMPYF32 MRa, #16FHi, MRb (continued)

32-Bit Floating-Point Multiply

Example 3

```
; Given X, M, and B are IQ24 numbers:
; X = IQ24(+2.5) = 0x02800000
; M = IQ24(+1.5) = 0x01800000
; B = IQ24(-0.5) = 0xFF800000
;
; Calculate Y = X * M + B
;
;
_Cla1Task2:
;
; Convert M, X, and B from IQ24 to float
     MI32TOF32   MR0, @_M          ; MR0 = 0x4BC00000
     MI32TOF32   MR1, @_X          ; MR1 = 0x4C200000
     MI32TOF32   MR2, @_B          ; MR2 = 0xCB000000
     MMPYF32     MR0, MR0, #0x3380 ; M = 1/(1*2^24) * iqm = 1.5 (0x3FC00000)
     MMPYF32     MR1, MR1, #0x3380 ; X = 1/(1*2^24) * iqx = 2.5 (0x40200000)
     MMPYF32     MR2, MR2, #0x3380 ; B = 1/(1*2^24) * iqb = -.5 (0xBF000000)
     MMPYF32     MR3, MR0, MR1     ; M*X
     MADDF32     MR2, MR2, MR3     ; Y=MX+B = 3.25 (0x40500000)
; Convert Y from float32 to IQ24
     MMPYF32     MR2, MR2, #0x4B80 ; Y * 1*2^24
     MF32TOI32   MR2, MR2          ; IQ24(Y) = 0x03400000
     MMOV32 @_Y, MR2               ; store result
     MSTOP                         ; end of task
```

See also

MMPYF32 MRa, MRb, #16FHi
MMPYF32 MRa, MRb, MRc
MMPYF32 MRa, MRb, MRc || MADDF32 MRd, MRe, MRf

MMPYF32 MRa, MRb, #16FHi

32-Bit Floating-Point Multiply

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)
#16FHi	A 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The lower 16-bits of the mantissa are assumed to be all 0.

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 0111 1000 bbaa
```

Description

Multiply MRb with the floating-point value represented by the immediate operand. Store the result of the addition in MRa.

#16FHi is a 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0. #16FHi is most useful for representing constants where the lowest 16-bits of the mantissa are 0. Some examples are 2.0 (0x40000000), 4.0 (0x40800000), 0.5 (0x3F000000), and -1.5 (0xBFC00000). The assembler accepts either a hex or float as the immediate value. That is, the value -1.5 can be represented as #-1.5 or #0xBFC0.

```
MRa = MRb * #16FHi:0;
```

This instruction can also be written as MMPYF32 MRa, #16FHi, MRb.

Flags

This instruction modifies the following flags in the MSTF register:.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MMPYF32 generates an underflow condition.
- LVF = 1 if MMPYF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

Example 1

```
;Same as example 2 but #16FHi is represented in float
    MMOVIZ    MR3, #2.0        ; MR3 = 2.0 (0x40000000)
    MMPYF32   MR0, MR3, #3.0   ; MR0 = MR3 * 3.0 = 6.0 (0x40C00000)
    MMOV32    @_X, MR0         ; Save the result in variable X
```

Example 2

```
;Same as example 1 but #16FHi is represented in Hex
    MMOVIZ    MR3, #2.0           ; MR3 = 2.0 (0x40000000)
    MMPYF32   MR0, MR3, #0x4040   ; MR0 = MR3 * 0x4040 = 6.0 (0x40C00000)
    MMOV32    @_X, MR0            ; Save the result in variable X
```

MMPYF32 MRa, MRb, #16FHi (continued)

32-Bit Floating-Point Multiply

Example 3

```
; Given X, M, and B are IQ24 numbers:
; X = IQ24(+2.5) = 0x02800000
; M = IQ24(+1.5) = 0x01800000
; B = IQ24(-0.5) = 0xFF800000
;
; Calculate Y = X * M + B
;
_Cla1Task2:
;
; Convert M, X, and B from IQ24 to float
    MI32TOF32   MR0, @_M          ; MR0 = 0x4BC00000
    MI32TOF32   MR1, @_X          ; MR1 = 0x4C200000
    MI32TOF32   MR2, @_B          ; MR2 = 0xCB000000
    MMPYF32     MR0, #0x3380, MR0 ; M = 1/(1*2^24) * iqm = 1.5 (0x3FC00000)
    MMPYF32     MR1, #0x3380, MR1 ; X = 1/(1*2^24) * iqx = 2.5 (0x40200000)
    MMPYF32     MR2, #0x3380, MR2 ; B = 1/(1*2^24) * iqb = -.5 (0xBF000000)
    MMPYF32     MR3, MR0, MR1     ; M*X
    MADDF32     MR2, MR2, MR3     ; Y=MX+B = 3.25 (0x40500000)
; Convert Y from float32 to IQ24
    MMPYF32     MR2, #0x4B80, MR2 ; Y * 1*2^24
    MF32TOI32   MR2, MR2          ; IQ24(Y) = 0x03400000
    MMOV32      @_Y, MR2          ; store result
    MSTOP                         ; end of task
```

See also

MMPYF32 MRa, #16FHi, MRb
MMPYF32 MRa, MRb, MRc

MMPYF32 MRa, MRb, MRc||MADDF32 MRd, MRe, MRf

32-Bit Floating-Point Multiply with Parallel Add

Operands

MRa	CLA floating-point destination register for MMPYF32 (MR0 to MR3) MRa cannot be the same register as MRd
MRb	CLA floating-point source register for MMPYF32 (MR0 to MR3)
MRc	CLA floating-point source register for MMPYF32 (MR0 to MR3)
MRd	CLA floating-point destination register for MADDF32 (MR0 to MR3) MRd cannot be the same register as MRa
MRe	CLA floating-point source register for MADDF32 (MR0 to MR3)
MRf	CLA floating-point source register for MADDF32 (MR0 to MR3)

Opcode

```
LSW: 0000 ffee ddcc bbaa
MSW: 0111 1010 0000 0000
```

Description

Multiply the contents of two floating-point registers with parallel addition of two registers.

```
MRa = MRb * MRc;
MRd = MRe + MRf;
```

Restrictions

The destination register for the MMPYF32 and the MADDF32 must be unique. That is, MRa cannot be the same register as MRd.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MMPYF32 or MADDF32 generates an underflow condition.
- LVF = 1 if MMPYF32 or MADDF32 generates an overflow condition.

Pipeline

Both MMPYF32 and MADDF32 complete in a single cycle.

MMPYF32 MRa, MRb, MRc||MADDF32 MRd, MRe, MRf (continued)

32-Bit Floating-Point Multiply with Parallel Add

Example

```
; Perform 5 multiply and accumulate operations:
;
; X and Y are 32-bit floating-point arrays
;
; 1st multiply: A = X0 * Y0
; 2nd multiply: B = X1 * Y1
; 3rd multiply: C = X2 * Y2
; 4th multiply: D = X3 * Y3
; 5th multiply: E = X3 * Y3
;
; Result = A + B + C + D + E
;
_Cla1Task1:
    MMOVI16    MAR0, #_X               ; MAR0 points to X array
    MMOVI16    MAR1, #_Y               ; MAR1 points to Y array
    MNOP                               ; Delay for MAR0, MAR1 load
    MNOP                               ; Delay for MAR0, MAR1 load
                                       ; <-- MAR0 valid
    MMOV32     MR0, *MAR0[2]++         ; MR0 = X0, MAR0 += 2
                                       ; <-- MAR1 valid
    MMOV32     MR1, *MAR1[2]++         ; MR1 = Y0, MAR1 += 2
    MMPYF32    MR2, MR0, MR1           ; MR2 = A = X0 * Y0
||  MMOV32     MR0, *MAR0[2]++         ; In parallel MR0 = X1, MAR0 += 2
    MMOV32     MR1, *MAR1[2]++         ; MR1 = Y1, MAR1 += 2
    MMPYF32    MR3, MR0, MR1           ; MR3 = B = X1 * Y1
||  MMOV32     MR0, *MAR0[2]++         ; In parallel MR0 = X2, MAR0 += 2
    MMOV32     MR1, *MAR1[2]++         ; MR1 = Y2, MAR2 += 2
    MMACF32    MR3, MR2, MR2, MR0, MR1 ; MR3 = A + B, MR2 = C = X2 * Y2
||  MMOV32     MR0, *MAR0[2]++         ; In parallel MR0 = X3
    MMOV32     MR1, *MAR1[2]++         ; MR1 = Y3
    MMACF32    MR3, MR2, MR2, MR0, MR1 ; MR3 = (A + B) + C, MR2 = D = X3 * Y3
||  MMOV32     MR0, *MAR0              ; In parallel MR0 = X4
    MMOV32     MR1, *MAR1              ; MR1 = Y4
    MMPYF32    MR2, MR0, MR1           ; MR2 = E = X4 * Y4
||  MADDF32    MR3, MR3, MR2           ; in parallel MR3 = (A + B + C) + D

    MADDF32    MR3, MR3, MR2           ; MR3 = (A + B + C + D) + E
    MMOV32     @_Result, MR3           ; Store the result
    MSTOP                              ; end of task
```

See also MMACF32 MR3, MR2, MRd, MRe, MRf || MMOV32 MRa, mem32

MMPYF32 MRd, MRe, MRf ||MMOV32 MRa, mem32

32-Bit Floating-Point Multiply with Parallel Move

Operands

MRd	CLA floating-point destination register for MMPYF32 (MR0 to MR3) MRd cannot be the same register as MRa
MRe	CLA floating-point source register for MMPYF32 (MR0 to MR3)
MRf	CLA floating-point source register for MMPYF32 (MR0 to MR3)
MRa	CLA floating-point destination register for MMOV32 (MR0 to MR3) MRa cannot be the same register as MRd
mem32	32-bit memory location accessed using one of the available addressing modes. This is the source of MMOV32.

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0000 ffee ddaa addr
```

Description

Multiply the contents of two floating-point registers and load another.

```
MRd = MRe * MRf;
MRa = [mem32];
```

Restrictions

The destination register for the MMPYF32 and the MMOV32 must be unique. That is, MRa cannot be the same register as MRd.

Flags

This instruction modifies the following flags in the MSTF register:.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MMPYF32 generates an underflow condition.
- LVF = 1 if MMPYF32 generates an overflow condition.

The MMOV32 instruction sets the NF and ZF flags as follows:

```
NF = MRa(31);
ZF = 0;
if(MRa(30:23) == 0) { ZF = 1; NF = 0; }
```

Pipeline

Both MMPYF32 and MMOV32 complete in a single cycle.

Example 1

```
; Given M1, X1, and B1 are 32-bit floating point
; Calculate Y1 = M1*X1+B1
;
_Cla1Task1:
    MMOV32     MR0, @M1         ; Load MR0 with M1
    MMOV32     MR1, @X1         ; Load MR1 with X1
    MMPYF32    MR1, MR1, MR0    ; Multiply M1*X1
||  MMOV32     MR0, @B1         ; and in parallel load MR0 with B1
    MADDF32    MR1, MR1, MR0    ; Add M*X1 to B1 and store in MR1
    MMOV32     @Y1, MR1         ; Store the result
    MSTOP                       ; end of task
```

MMPYF32 MRd, MRe, MRf ||MMOV32 MRa, mem32 (continued)

32-Bit Floating-Point Multiply with Parallel Move

Example 2

```
; Given A, B, and C are 32-bit floating-point numbers
; Calculate Y2 = (A * B)
;           Y3 = (A * B) * C
;
_Cla1Task2:
    MMOV32    MR0, @A         ; Load MR0 with A
    MMOV32    MR1, @B         ; Load MR1 with B
    MMPYF32   MR1, MR1, MR0   ; Multiply A*B
||  MMOV32    MR0, @C         ; and in parallel load MR0 with C
    MMPYF32   MR1, MR1, MR0   ; Multiply (A*B) by C
||  MMOV32    @Y2, MR1        ; and in parallel store A*B
    MMOV32    @Y3, MR1        ; Store the result
    MSTOP                     ; end of task
```

See also

MMPYF32 MRd, MRe, MRf || MMOV32 mem32, MRa
MMACF32 MR3, MR2, MRd, MRe, MRf || MMOV32 MRa, mem32

MMPYF32 MRd, MRe, MRf ||MMOV32 mem32, MRa

32-Bit Floating-Point Multiply with Parallel Move

Operands

MRd	CLA floating-point destination register for MMPYF32 (MR0 to MR3)
MRe	CLA floating-point source register for MMPYF32 (MR0 to MR3)
MRf	CLA floating-point source register for MMPYF32 (MR0 to MR3)
mem32	32-bit memory location accessed using one of the available addressing modes. This is the destination of MMOV32.
MRa	CLA floating-point source register for MMOV32 (MR0 to MR3)

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0100 ffee ddaa addr
```

Description

Multiply the contents of two floating-point registers and move from memory to register.

```
MRd = MRe * MRf;
[mem32] = MRa;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MMPYF32 generates an underflow condition.
- LVF = 1 if MMPYF32 generates an overflow condition.

Pipeline

MMPYF32 and MMOV32 both complete in a single cycle.

Example

```
; Given A, B, and C are 32-bit floating-point numbers
; Calculate Y2 = (A * B)
;           Y3 = (A * B) * C
;
_Cla1Task2:
     MMOV32    MR0, @A         ; Load MR0 with A
     MMOV32    MR1, @B         ; Load MR1 with B
     MMPYF32   MR1, MR1, MR0   ; Multiply A*B
||   MMOV32    MR0, @C         ; and in parallel load MR0 with C
     MMPYF32   MR1, MR1, MR0   ; Multiply (A*B) by C
||   MMOV32    @Y2, MR1        ; and in parallel store A*B
     MMOV32    @Y3, MR1        ; Store the result
     MSTOP                     ; end of task
```

See also

MMPYF32 MRd, MRe, MRf || MMOV32 MRa, mem32
MMACF32 MR3, MR2, MRd, MRe, MRf || MMOV32 MRa, mem32

MMPYF32 MRa, MRb, MRc ||MSUBF32 MRd, MRe, MRf

32-Bit Floating-Point Multiply with Parallel Subtract

Operands

MRa	CLA floating-point destination register for MMPYF32 (MR0 to MR3) MRa cannot be the same register as MRd
MRb	CLA floating-point source register for MMPYF32 (MR0 to MR3)
MRc	CLA floating-point source register for MMPYF32 (MR0 to MR3)
MRd	CLA floating-point destination register for MSUBF32 (MR0 to MR3) MRd cannot be the same register as MRa
MRe	CLA floating-point source register for MSUBF32 (MR0 to MR3)
MRf	CLA floating-point source register for MSUBF32 (MR0 to MR3)

Opcode

```
LSW: 0000 ffee ddcc bbaa
MSW: 0111 1010 0100 0000
```

Description

Multiply the contents of two floating-point registers with parallel subtraction of two registers.

```
MRa = MRb * MRc;
MRd = MRe - MRf;
```

Restrictions

The destination register for the MMPYF32 and the MSUBF32 must be unique. That is, MRa cannot be the same register as MRd.

Flags

This instruction modifies the following flags in the MSTF register:.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MMPYF32 or MSUBF32 generates an underflow condition.
- LVF = 1 if MMPYF32 or MSUBF32 generates an overflow condition.

Pipeline

MMPYF32 and MSUBF32 both complete in a single cycle.

Example

```
; Given A, B, and C are 32-bit floating-point numbers
; Calculate Y2 = (A * B)
;           Y3 = (A - B)
;
_Cla1Task2:
     MMOV32   MR0, @A          ; Load MR0 with A
     MMOV32   MR1, @B          ; Load MR1 with B
     MMPYF32  MR2, MR0, MR1    ; Multiply (A*B)
||   MSUBF32  MR3, MR0, MR1    ; and in parallel Sub (A-B)
     MMOV32   @Y2, MR2         ; Store A*B
     MMOV32   @Y3, MR3         ; Store A-B
     MSTOP                     ; end of task
```

See also

MSUBF32 MRa, MRb, MRc
MSUBF32 MRd, MRe, MRf || MMOV32 MRa, mem32
MSUBF32 MRd, MRe, MRf || MMOV32 mem32, MRa

MNEGF32 MRa, MRb{, CNDF}

Conditional Negation

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)
CNDF	Condition tested

Opcode

```
LSW: 0000 0000 cndf bbaa
MSW: 0111 1010 1000 0000
```

Description

```
if (CNDF == true) {MRa = - MRb; }
else {MRa = MRb; }
```

CNDF is one of the following conditions:

Encode[1]	CNDF	Description	MSTF Flags Tested
0000	NEQ	Not equal to zero	ZF == 0
0001	EQ	Equal to zero	ZF == 1
0010	GT	Greater than zero	ZF == 0 AND NF == 0
0011	GEQ	Greater than or equal to zero	NF == 0
0100	LT	Less than zero	NF == 1
0101	LEQ	Less than or equal to zero	ZF == 1 OR NF == 1
1010	TF	Test flag set	TF == 1
1011	NTF	Test flag not set	TF == 0
1100	LU	Latched underflow	LUF == 1
1101	LV	Latched overflow	LVF == 1
1110	UNC	Unconditional	None
1111	UNCF[2]	Unconditional with flag modification	None

(1) Values not shown are reserved.
(2) This is the default operation, if no CNDF field is specified. This condition allows the ZF, and NF flags to be modified when a conditional operation is executed. All other conditions do not modify these flags.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

Pipeline

This is a single-cycle instruction.

MNEGF32 MRa, MRb{, CNDF} (continued)

Conditional Negation

Example 1

```
; Show the basic operation of MNEGF32
;
     MMOVIZ    MR0, #5.0       ; MR0 = 5.0 (0x40A00000)
     MMOVIZ    MR1, #4.0       ; MR1 = 4.0 (0x40800000)
     MMOVIZ    MR2, #-1.5      ; MR2 = -1.5 (0xBFC00000)
     MMPYF32   MR3, MR1, MR2   ; MR3 = -6.0
     MMPYF32   MR0, MR0, MR1   ; MR0 = 20.0
     MMOVIZ    MR1, #0.0
     MCMPF32   MR3, MR1        ; NF = 1
     MNEGF32   MR3, MR3, LT    ; if NF = 1, MR3 = 6.0
     MCMPF32   MR0, MR1        ; NF = 0
     MNEGF32   MR0, MR0, GEQ   ; if NF = 0, MR0 = -20.0
```

Example 2

```
; Calculate Num/Den using a Newton-Raphson algorithum for 1/Den
; Ye = Estimate(1/X)
; Ye = Ye*(2.0 - Ye*X)
; Ye = Ye*(2.0 - Ye*X)
;
_Cla1Task1:
    MMOV32    MR1, @_Den      ; MR1 = Den
    MEINVF32  MR2, MR1        ; MR2 = Ye = Estimate(1/Den)
    MMPYF32   MR3, MR2, MR1   ; MR3 = Ye*Den
    MSUBF32   MR3, #2.0, MR3  ; MR3 = 2.0 - Ye*Den
    MMPYF32   MR2, MR2, MR3   ; MR2 = Ye = Ye*(2.0 - Ye*Den)
    MMPYF32   MR3, MR2, MR1   ; MR3 = Ye*Den
||  MMOV32    MR0, @_Num      ; MR0 = Num
    MSUBF32   MR3, #2.0, MR3  ; MR3 = 2.0 - Ye*Den
    MMPYF32   MR2, MR2, MR3   ; MR2 = Ye = Ye*(2.0 - Ye*Den)
||  MMOV32    MR1, @_Den      ; Reload Den To Set Sign
    MNEGF32   MR0, MR0, EQ    ; if(Den == 0.0) Change Sign Of Num
    MMPYF32   MR0, MR2, MR0   ; MR0 = Y = Ye*Num
    MMOV32    @_Dest, MR0     ; Store result
    MSTOP                     ; end of task
```

See also MABSF32 MRa, MRb

MNOP

No Operation

Operands

none	This instruction does not have any operands

Opcode

```
LSW: 0000 0000 0000 0000
MSW: 0111 1111 1010 0000
```

Description

Do nothing. This instruction is used to fill required pipeline delay slots when other instructions are not available to fill the slots.

Flags

This instruction does not modify flags in the MSTF register.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
; X is an array of 32-bit floating-point values
; Find the maximum value in an array X
; and store the value in Result
;
_Cla1Task1:
    MMOVI16    MAR1,#_X          ; Start address
    MUI16TOF32 MR0, @_len        ; Length of the array
    MNOP                         ; delay for MAR1 load
    MNOP                         ; delay for MAR1 load
    MMOV32     MR1, *MAR1[2]++   ; MR1 = X0
LOOP
    MMOV32     MR2, *MAR1[2]++   ; MR2 = next element
    MMAXF32    MR1, MR2          ; MR1 = MAX(MR1, MR2)
    MADDF32    MR0, MR0, #-1.0   ; Decrememt the counter
    MCMPF32    MR0 #0.0          ; Set/clear flags for MBCNDD
    MNOP                         ; Too late to affect MBCNDD
    MNOP                         ; Too late to affect MBCNDD
    MNOP                         ; Too late to affect MBCNDD
    MBCNDD    LOOP, NEQ          ; Branch if not equal to zero
    MMOV32    @_Result, MR1      ; Always executed
    MNOP                         ; Pad to seperate MBCNDD and MSTOP
    MNOP                         ; Pad to seperate MBCNDD and MSTOP
    MSTOP                        ; End of task
```

MOR32 MRa, MRb, MRc

Bitwise OR

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)
MRc	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 00cc bbaa
MSW: 0111 1100 1000 0000
```

Description

Bitwise OR of MRb with MRc.

```
MARa(31:0) = MARb(31:0) OR MRc(31:0);
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified based on the integer results of the operation.

```
NF = MRa(31);
ZF = 0;
if(MRa(31:0) == 0) { ZF = 1; }
```

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ    MR0, #0x5555 ; MR0 = 0x5555AAAA
MMOVXI    MR0, #0xAAAA
MMOVIZ    MR1, #0x5432 ; MR1 = 0x5432FEDC
MMOVXI    MR1, #0xFEDC
                       ; 0101 OR 0101 = 0101 (5)
                       ; 0101 OR 0100 = 0101 (5)
                       ; 0101 OR 0011 = 0111 (7)
                       ; 0101 OR 0010 = 0111 (7)
                       ; 1010 OR 1111 = 1111 (F)
                       ; 1010 OR 1110 = 1110 (E)
                       ; 1010 OR 1101 = 1111 (F)
                       ; 1010 OR 1100 = 1110 (E)
MOR32 MR2, MR1, MR0    ; MR3 = 0x5555FEFE
```

See also

MAND32 MRa, MRb, MRc
MXOR32 MRa, MRb, MRc

MRCNDD {CNDF}

Return Conditional Delayed

Operands

CNDF	Optional condition

Opcode

```
LSW: 0000 0000 0000 0000
MSW: 0111 1001 1010 cndf
```

Description

If the specified condition is true, then the RPC field of MSTF is loaded into MPC and fetching continues from that location. Otherwise, program fetches continue without the return.

Refer to the Pipeline section for important information regarding this instruction.

```
if (CNDF == TRUE) MPC = RPC;
```

CNDF is one of the following conditions:

Encode(1)	CNDF	Description	MSTF Flags Tested
0000	NEQ	Not equal to zero	ZF == 0
0001	EQ	Equal to zero	ZF == 1
0010	GT	Greater than zero	ZF == 0 AND NF == 0
0011	GEQ	Greater than or equal to zero	NF == 0
0100	LT	Less than zero	NF == 1
0101	LEQ	Less than or equal to zero	ZF == 1 OR NF == 1
1010	TF	Test flag set	TF == 1
1011	NTF	Test flag not set	TF == 0
1100	LU	Latched underflow	LUF == 1
1101	LV	Latched overflow	LVF == 1
1110	UNC	Unconditional	None
1111	UNCF(2)	Unconditional with flag modification	None

(1) Values not shown are reserved.
(2) This is the default operation, if no CNDF field is specified. This condition allows the ZF and NF flags to be modified when a conditional operation is executed. All other conditions do not modify these flags.

Flags

This instruction does not modify flags in the MSTF register.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

The MRCNDD instruction by itself is a single-cycle instruction. As shown in Table 8-19, 6 instruction slots are executed for each return; 3 slots before the return instruction (d5-d7) and 3 slots after the return instruction (d8-d10). The total number of cycles for a return taken or not taken depends on the usage of these slots. That is, the number of cycles depends on how many slots are filled with a MNOP as well as which slots are filled. The effective number of cycles for a return can, therefore, range from 1 to 7 cycles. The number of cycles for a return taken cannot be the same as for a return not taken.

MRCNDD {CNDF} (continued)

Return Conditional Delayed

Referring to the following code fragment and the pipeline diagrams in Table 8-19 and Table 8-20, the instructions before and after MRCNDD have the following properties:

```
;
;
<Instruction 1>     ; I1 Last instruction that can affect flags for
                    ; the MCCNDD operation
<Instruction 2>     ; I2 Cannot be stop, branch, call or return
<Instruction 3>     ; I3 Cannot be stop, branch, call or return
<Instruction 4>     ; I4 Cannot be stop, branch, call or return
MCCNDD _func, NEQ   ; Call to func if not eqal to zero
                    ; Three instructions after MCCNDD are always
                    ; executed whether the call is taken or not
<Instruction 5>     ; I5 Cannot be stop, branch, call or return
<Instruction 6>     ; I6 Cannot be stop, branch, call or return
<Instruction 7>     ; I7 Cannot be stop, branch, call or return
<Instruction 8>     ; I8 The address of this instruction is saved
                    ; in the RPC field of the MSTF register.
                    ; Upon return this value is loaded into MPC
                    ; and fetching continues from this point.
<Instruction 9>     ; I9
<Instruction 10>    ; I10
....
....
_func:
<Destination 1>     ; d1 Can be any instruction
<Destination 2>     ; d2
<Destination 3>     ; d3
<Destination 4>     ; d4 Last instruction that can affect flags for
                    ; the MRCNDD operation
<Destination 5>     ; d5 Cannot be stop, branch, call or return
<Destination 6>     ; d6 Cannot be stop, branch, call or return
<Destination 7>     ; d7 Cannot be stop, branch, call or return
MRCNDD NEQ          ; Return to <Instruction 8> if not equal to zero
                    ; Three instructions after MRCNDD are always
                    ; executed whether the return is taken or not
<Destination 8>     ; d8 Cannot be stop, branch, call or return
<Destination 9>     ; d9 Cannot be stop, branch, call or return
<Destination 10>    ; d10 Cannot be stop, branch, call or return
<Destination 11>    ; d11
<Destination 12>    ; d12
....
....
MSTOP
....
```

- **d4**
  - d4 is the last instruction that can effect the CNDF flags for the MRCNDD instruction. The CNDF flags are tested in the D2 phase of the pipeline. That is, a decision is made whether to return or not when MRCNDD is in the D2 phase.
  - There are no restrictions on the type of instruction for d4.
- **d5, d6, and d7**
  - The three instructions proceeding MRCNDD can change MSTF flags but have no effect on whether the MRCNDD instruction makes the return or not. This is because the flag modification occurs after the D2 phase of the MRCNDD instruction.
  - These instructions must not be the following: MSTOP, MDEBUGSTOP, MBCNDD, MCCNDD, or MRCNDD.
- **d8, d9, and d10**
  - The three instructions following MRCNDD are always executed irrespective of whether the return is taken or not.
  - These instructions must not be the following: MSTOP, MDEBUGSTOP, MBCNDD, MCCNDD, or MRCNDD.

MRCNDD {CNDF} (continued)

Return Conditional Delayed

Table 8-19. Pipeline Activity for MRCNDD, Return Not Taken

Instruction	F1	F2	D1	D2	R1	R2	E	W
d4	d4	d3	d2	d1	I7	I6	I5	
d5	d5	d4	d3	d2	d1	I7	I6	
d6	d6	d5	d4	d3	d2	d1	i7	
d7	d7	d6	d5	d4	d3	d2	d1	
MRCNDD	MRCNDD	d7	d6	d5	d4	d3	d2	
d8	d8	MRCNDD	d7	d6	d5	d4	d3	
d9	d9	d8	MRCNDD	d7	d6	d5	d4	
d10	d10	d9	d8	MRCNDD	d7	d6	d5	
d11	d11	d10	d9	d8	-	d7	d6	
d12	d12	d11	d10	d9	d8	-	d7	
etc....	....	d12	d11	d10	d9	d8	-	
....	....	....	d12	d11	d10	d9	d8	
....	....	....	....	d12	d11	d10	d9	
					d12	d11	d10	
						d12	d11	
							d12	

Table 8-20. Pipeline Activity for MRCNDD, Return Taken

Instruction	F1	F2	D1	D2	R1	R2	E	W
d4	d4	d3	d2	d1	I7	I6	I5	
d5	d5	d4	d3	d2	d1	I7	I6	
d6	d6	d5	d4	d3	d2	d1	i7	
d7	d7	d6	d5	d4	d3	d2	d1	
MRCNDD	MRCNDD	d7	d6	d5	d4	d3	d2	
d8	d8	MRCNDD	d7	d6	d5	d4	d3	
d9	d9	d8	MRCNDD	d7	d6	d5	d4	
d10	d10	d9	d8	MRCNDD	d7	d6	d5	
I8	I8	d10	d9	d8	-	d7	d6	
I9	I9	I8	d10	d9	d8	-	d7	
I10	I10	I9	I8	d10	d9	d8	-	
etc....	....	I10	I9	I8	d10	d9	d8	
....	....		I10	I9	I8	d10	d9	
....	....			I10	I9	I8	d10	
					I10	I9	I8	
						I10	I9	
							I10	

See also

MBCNDD #16BitDest, CNDF
MCCNDD 16BitDest, CNDF
MMOV32 mem32, MSTF
MMOV32 MSTF, mem32

MSETC BGINTM

Set Background Task Interrupt Mask

Operands

none	This instruction does not have any operands

Opcode

```
LSW: 0000 0000 0000 0000
MSW: 0111 1111 0101 0000
```

Description

This instruction sets the background task interrupt mask (BGINTM) bit in the MSTSBGRND register, making any code thereafter uninterruptible. No other higher priority task is able to interrupt the background task until the BGINTM is cleared. This instruction sets the BGINTM bit at the end of the D2 phase.

This instruction does not require the MEALLOW bit to be asserted before, or de-asserted after, setting BGINTM.

Flags

This instruction does not modify the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MSETC BGINTM          ; Set the MSTSBGRND.BGINTM bit
                      ; to prevent any other tasks from
                      ; interrupting the background task
```

See also

MCLRC BGINTM

MSETFLG FLAG, VALUE

Set or Clear Selected Floating-Point Status Flags

Operands

FLAG	8-bit mask indicating which floating-point status flags to change.
VALUE	8-bit mask indicating the flag value: 0 or 1.

Opcode

```
LSW: FFFF FFFF VVVV VVVV
MSW: 0111 1001 1100 0000
```

Description

The MSETFLG instruction is used to set or clear selected floating-point status flags in the MSTF register. The FLAG field is an 11-bit value that indicates which flags are changed. That is, if a FLAG bit is set to 1, that flag is changed; all other flags are not modified. The bit mapping of the FLAG field is:

9	8	7	6	5	4	3	2	1	0
RNDF32	Reserved		TF	Reserved		ZF	NF	LUF	LVF

The VALUE field indicates the value the flag can be set to: 0 or 1.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	Yes	Yes	Yes	Yes	Yes

Any flag can be modified by this instruction. The MEALLOW and RPC fields cannot be modified with this instruction.

Pipeline

This is a single-cycle instruction.

Example

To make it easier and legible, the assembler accepts a FLAG=VALUE syntax for the MSTFLG operation as:

```
MSETFLG RNDF32=0, TF=0, NF=1; FLAG = 11000100; VALUE = 00XXX1XX;
```

See also

MMOV32 mem32, MSTF
MMOV32 MSTF, mem32

MSTOP

Stop Task

Operands

none	This instruction does not have any operands

Opcode

```
LSW: 0000 0000 0000 0000
MSW: 0111 1111 1000 0000
```

Description

The MSTOP instruction must be placed to indicate the end of each task. In addition, placing MSTOP in unused memory locations within the CLA program RAM can be useful for debugging and preventing run away CLA code. When MSTOP enters the D2 phase of the pipeline, the MIRUN flag for the task is cleared and the associated interrupt is flagged in the PIE vector table.

There are three special cases that can occur when single-stepping a task such that the MPC reaches the MSTOP instruction.

1. If you are single-stepping or halted in "task A" and "task B" comes in before the MPC reaches the MSTOP, then "task B" starts if you continue to step through the MSTOP instruction. Basically, if "task B" is pending before the MPC reaches MSTOP in "task A" then there is no issue in "task B" starting and no special action is required.
2. In this case, you have single-stepped or halted in "task A" and the MPC has reached the MSTOP with no tasks pending. If "task B" comes in at this point, "task B" is flagged in the MIFR register but "task B" can or cannot start if you continue to single-step through the MSTOP instruction of "task A". It depends on exactly when the new task comes in. To reliably start "task B", perform a soft reset and reconfigure the MIER bits. Once this is done, you can start single-stepping "task B".
3. Case 2 can be handled slightly differently if there is control over when "task B" comes in (for example using the IACK instruction to start the task). In this case you have single-stepped or halted in "task A" and the MPC has reached the MSTOP with no tasks pending. Before forcing "task B", run free to force the CLA out of the debug state. Once this is done you can force "task B" and continue debugging.

Restrictions

The MSTOP instruction cannot be placed 3 instructions before or after a MBCNDD, MCCNDD, or MRCNDD instruction.

Flags

This instruction does not modify flags in the MSTF register.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

MSTOP (continued)

Stop Task

Pipeline

This is a single-cycle instruction. Table 8-21 shows the pipeline behavior of the MSTOP instruction. The MSTOP instruction cannot be placed with 3 instructions of a MBCNDD, MCCNDD, or MRCNDD instruction.

Table 8-21. Pipeline Activity for MSTOP

Instruction	F1	F2	D1	D2	R1	R2	E	W
I1	I1							
I2	I2	I1						
I3	I3	I2	I1					
MSTOP	MSTOP	I3	I2	I1				
I4	I4	MSTOP	I3	I2	I1			
I5	I5	I4	MSTOP	I3	I2	I1		
I6	I6	I5	I4	MSTOP	I3	I2	I1	
New Task Arbitrated and Prioritized	-	-	-	-	-	I3	I2	
New Task Arbitrated and Prioritized	-	-	-	-	-	-	I3	
I1	I1	-	-	-	-	-	-	
I2	I2	I1	-	-	-	-	-	
I3	I3	I2	I1	-	-	-	-	
I4	I4	I3	I2	I1	-	-	-	
I5	I5	I4	I3	I2	I1	-	-	
I6	I6	I5	I4	I3	I2	I1	-	
I7	I7	I6	I5	I4	I3	I2	I1	
....								

Example

```
; Given A = (int32)1
;       B = (int32)2
;       C = (int32)-7
;
; Calculate Y2 = A - B - C
_Cla1Task3:
   MMOV32   MR0, @_A       ; MR0 = 1 (0x00000001)
   MMOV32   MR1, @_B       ; MR1 = 2 (0x00000002)
   MMOV32   MR2, @_C       ; MR2 = -7 (0xFFFFFFF9)
   MSUB32   MR3, MR0, MR1 ; A + B
   MSUB32   MR3, MR3, MR2 ; A + B + C = 6 (0x0000006)
   MMOV32   @_y2, MR3      ; Store y2
   MSTOP                   ; End of task
```

See also

MDEBUGSTOP , MDEBUGSTOP1

MSUB32 MRa, MRb, MRc

32-Bit Integer Subtraction

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point destination register (MR0 to MR3)
MRc	CLA floating-point destination register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 00cc bbaa
MSW: 0111 1100 1110 0000
```

Description

32-bit integer addition of MRb and MRc.

```
MARa(31:0) = MARb(31:0) - MRc(31:0);
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified as follows:

```
NF = MRa(31);
ZF = 0;
if(MRa(31:0) == 0) { ZF = 1; }
```

Pipeline

This is a single-cycle instruction.

Example

```
; Given A = (int32)1
;       B = (int32)2
;       C = (int32)-7
;
;
Calculate Y2 = A - B - C
;
_Cla1Task3:
   MMOV32   MR0, @_A          ; MR0 = 1 (0x00000001)
   MMOV32   MR1, @_B          ; MR1 = 2 (0x00000002)
   MMOV32   MR2, @_C          ; MR2 = -7 (0xFFFFFFF9)
   MSUB32   MR3, MR0, MR1     ; A + B
   MSUB32   MR3, MR3, MR2     ; A + B + C = 6 (0x0000006)
   MMOV32   @_y2, MR3         ; Store y2
   MSTOP                      ; End of task
```

See also

MADD32 MRa, MRb, MRc
MAND32 MRa, MRb, MRc
MASR32 MRa, #SHIFT
MLSL32 MRa, #SHIFT
MLSR32 MRa, #SHIFT
MOR32 MRa, MRb, MRc
MXOR32 MRa, MRb, MRc

MSUBF32 MRa, MRb, MRc

32-Bit Floating-Point Subtraction

Operands

MRa	CLA floating-point destination register (MR0 to R1)
MRb	CLA floating-point source register (MR0 to R1)
MRc	CLA floating-point source register (MR0 to R1)

Opcode

```
LSW: 0000 0000 00cc bbaa
MSW: 0111 1100 0100 0000
```

Description

Subtract the contents of two floating-point registers

```
MRa = MRb - MRc;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MSUBF32 generates an underflow condition.
- LVF = 1 if MSUBF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

Example

```
; Given A, B, and C are 32-bit floating-point numbers
; Calculate Y2 = A + B - C
;
_Cla1Task5:
   MMOV32    MR0, @_A      ; Load MR0 with A
   MMOV32    MR1, @_B      ; Load MR1 with B
   MADDF32   MR0, MR1, MR0 ; Add A + B
|| MMOV32    MR1, @_C      ;   and in parallel load C
   MSUBF32   MR0, MR0, MR1 ; Subtract C from (A + B)
   MMOV32    @Y,  MR0      ; (A+B) - C
   MSTOP                   ; end of task
```

See also

MSUBF32 MRa, #16FHi, MRb
MSUBF32 MRd, MRe, MRf || MMOV32 MRa, mem32
MSUBF32 MRd, MRe, MRf || MMOV32 mem32, MRa
MMPYF32 MRa, MRb, MRc || MSUBF32 MRd, MRe, MRf

MSUBF32 MRa, #16FHi, MRb

32-Bit Floating-Point Subtraction

Operands

MRa	CLA floating-point destination register (MR0 to R1)
#16FHi	A 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The lower 16-bits of the mantissa are assumed to be all 0.
MRb	CLA floating-point source register (MR0 to R1)

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 1000 0000 baaa
```

Description

Subtract MRb from the floating-point value represented by the immediate operand. Store the result of the addition in MRa.

#16FHi is a 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The lower 16-bits of the mantissa are assumed to be all 0. #16FHi is most useful for representing constants where the lowest 16-bits of the mantissa are 0. Some examples are 2.0 (0x40000000), 4.0 (0x40800000), 0.5 (0x3F000000), and -1.5 (0xBFC00000). The assembler accepts either a hex or float as the immediate value. That is, the value -1.5 can be represented as #-1.5 or #0xBFC0.

```
MRa = #16FHi:0 - MRb;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MSUBF32 generates an underflow condition.
- LVF = 1 if MSUBF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

Example

```
; Y = sqrt(X)
; Ye = Estimate(1/sqrt(X));
; Ye = Ye*(1.5 - Ye*Ye*X*0.5)
; Ye = Ye*(1.5 - Ye*Ye*X*0.5)
; Y = X*Ye
;
_cla1Task3:
   MMOV32     MR0, @_x           ; MR0 = X
   MEISQRTF32 MR1, MR0           ; MR1 = Ye = Estimate(1/sqrt(X))
   MMOV32     MR1, @_x, EQ       ; if(X == 0.0) Ye = 0.0
   MMPYF32    MR3, MR0, #0.5     ; MR3 = X*0.5
   MMPYF32    MR2, MR1, MR3      ; MR2 = Ye*X*0.5
   MMPYF32    MR2, MR1, MR2      ; MR2 = Ye*Ye*X*0.5
   MSUBF32    MR2, #1.5, MR2     ; MR2 = 1.5 - Ye*Ye*X*0.5
   MMPYF32    MR1, MR1, MR2      ; MR1 = Ye = Ye*(1.5 - Ye*Ye*X*0.5)
   MMPYF32    MR2, MR1, MR3      ; MR2 = Ye*X*0.5
   MMPYF32    MR2, MR1, MR2      ; MR2 = Ye*Ye*X*0.5
   MSUBF32    MR2, #1.5, MR2     ; MR2 = 1.5 - Ye*Ye*X*0.5
   MMPYF32    MR1, MR1, MR2      ; MR1 = Ye = Ye*(1.5 - Ye*Ye*X*0.5)
   MMPYF32    MR0, MR1, MR0      ; MR0 = Y = Ye*X
   MMOV32     @_y, MR0           ; Store Y = sqrt(X)
   MSTOP                         ; end of task
```

MSUBF32 MRa, #16FHi, MRb (continued)

32-Bit Floating-Point Subtraction

See also

MSUBF32 MRa, MRb, MRc
MSUBF32 MRd, MRe, MRf || MMOV32 MRa, mem32
MSUBF32 MRd, MRe, MRf || MMOV32 mem32, MRa
MMPYF32 MRa, MRb, MRc || MSUBF32 MRd, MRe, MRf

MSUBF32 MRd, MRe, MRf ||MMOV32 MRa, mem32

32-Bit Floating-Point Subtraction with Parallel Move

Operands

MRd	CLA floating-point destination register (MR0 to MR3) for the MSUBF32 operation MRd cannot be the same register as MRa
MRe	CLA floating-point source register (MR0 to MR3) for the MSUBF32 operation
MRf	CLA floating-point source register (MR0 to MR3) for the MSUBF32 operation
MRa	CLA floating-point destination register (MR0 to MR3) for the MMOV32 operation MRa cannot be the same register as MRd
mem32	32-bit memory location accessed using one of the available addressing modes. Source for the MMOV32 operation.

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0010 ffee ddaa addr
```

Description

Subtract the contents of two floating-point registers and move from memory to a floating-point register.

```
MRd = MRe - MRf;
MRa = [mem32];
```

Restrictions

The destination register for the MSUBF32 and the MMOV32 must be unique. That is, MRa cannot be the same register as MRd.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MSUBF32 generates an underflow condition.
- LVF = 1 if MSUBF32 generates an overflow condition.

The MMOV32 instruction sets the NF and ZF flags.

Pipeline

Both MSUBF32 and MMOV32 complete in a single cycle.

Example

```
NF = MRa(31);
ZF = 0;
if(MRa(30:23) == 0) { ZF = 1; NF = 0; }
```

See also

MSUBF32 MRa, MRb, MRc
MSUBF32 MRa, #16FHi, MRb
MMPYF32 MRa, MRb, MRc || MSUBF32 MRd, MRe, MRf

MSUBF32 MRd, MRe, MRf ||MMOV32 mem32, MRa

32-Bit Floating-Point Subtraction with Parallel Move

Operands

MRd	CLA floating-point destination register (MR0 to MR3) for the MSUBF32 operation
MRe	CLA floating-point source register (MR0 to MR3) for the MSUBF32 operation
MRf	CLA floating-point source register (MR0 to MR3) for the MSUBF32 operation
mem32	32-bit destination memory location for the MMOV32 operation
MRa	CLA floating-point source register (MR0 to MR3) for the MMOV32 operation

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0110 ffee ddaa addr
```

Description

Subtract the contents of two floating-point registers and move from a floating-point register to memory.

```
MRd = MRe - MRf;
[mem32] = MRa;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MSUBF32 generates an underflow condition.
- LVF = 1 if MSUBF32 generates an overflow condition.

Pipeline

Both MSUBF32 and MMOV32 complete in a single cycle.

See also

MSUBF32 MRa, MRb, MRc
MSUBF32 MRa, #16FHi, MRb
MSUBF32 MRd, MRe, MRf || MMOV32 MRa, mem32
MMPYF32 MRa, MRb, MRc || MSUBF32 MRd, MRe, MRf

MSWAPF MRa, MRb {, CNDF}

Conditional Swap

Operands

MRa	CLA floating-point register (MR0 to MR3)
MRb	CLA floating-point register (MR0 to MR3)
CNDF	Optional condition tested based on the MSTF flags

Opcode

```
LSW: 0000 0000 CNDF bbaa
MSW: 0111 1011 0000 0000
```

Description

Conditional swap of MRa and MRb.

```
if (CNDF == true) swap MRa and MRb;
```

CNDF is one of the following conditions:

Encode[1]	CNDF	Description	MSTF Flags Tested
0000	NEQ	Not equal to zero	ZF == 0
0001	EQ	Equal to zero	ZF == 1
0010	GT	Greater than zero	ZF == 0 AND NF == 0
0011	GEQ	Greater than or equal to zero	NF == 0
0100	LT	Less than zero	NF == 1
0101	LEQ	Less than or equal to zero	ZF == 1 OR NF == 1
1010	TF	Test flag set	TF == 1
1011	NTF	Test flag not set	TF == 0
1100	LU	Latched underflow	LUF == 1
1101	LV	Latched overflow	LVF == 1
1110	UNC	Unconditional	None
1111	UNCF[2]	Unconditional with flag modification	None

(1) Values not shown are reserved.
(2) This is the default operation, if no CNDF field is specified. This condition allows the ZF and NF flags to be modified when a conditional operation is executed. All other conditions do not modify these flags.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

No flags affected

Pipeline

This is a single-cycle instruction.

MSWAPF MRa, MRb {, CNDF} (continued)

Conditional Swap

Example

```
; X is an array of 32-bit floating-point values
; and has length elements. Find the maximum value in
; the array and store the value in Result
;
; Note: MCMPF32 and MSWAPF can be replaced by MMAXF32
;
_Cla1Task1:
    MMOVI16    MAR1,#_X          ; Start address
    MUI16TOF32 MR0, @_len        ; Length of the array
    MNOP                         ; delay for MAR1 load
    MNOP                         ; delay for MAR1 load
    MMOV32     MR1, *MAR1[2]++   ; MR1 = X0
LOOP
    MMOV32     MR2, *MAR1[2]++   ; MR2 = next element
    MCMPF32    MR2, MR1          ; Compare MR2 with MR1
    MSWAPF     MR1, MR2, GT      ; MR1 = MAX(MR1, MR2)
    MADDF32    MR0, MR0, #-1.0   ; Decrememt the counter
    MCMPF32    MR0 #0.0          ; Set/clear flags for MBCNDD
    MNOP
    MNOP
    MNOP
    MBCNDD     LOOP, NEQ         ; Branch if not equal to zero
    MMOV32     @_Result, MR1     ; Always executed
    MNOP                         ; Always executed
    MNOP                         ; Always executed
    MSTOP                        ; End of task
```

MTESTTF CNDF

Test MSTF Register Flag Condition

Operands

CNDF	Condition to test based on MSTF flags

Opcode

```
LSW: 0000 0000 0000 cndf
MSW: 0111 1111 0100 0000
```

Description

Test the CLA floating-point condition and if true, set the MSTF[TF] flag. If the condition is false, clear the MSTF[TF] flag. This is useful for temporarily storing a condition for later use.

```
if (CNDF == true) TF = 1;
else TF = 0;
```

CNDF is one of the following conditions:

Encode[1]	CNDF	Description	MSTF Flags Tested
0000	NEQ	Not equal to zero	ZF == 0
0001	EQ	Equal to zero	ZF == 1
0010	GT	Greater than zero	ZF == 0 AND NF == 0
0011	GEQ	Greater than or equal to zero	NF == 0
0100	LT	Less than zero	NF == 1
0101	LEQ	Less than or equal to zero	ZF == 1 OR NF == 1
1010	TF	Test flag set	TF == 1
1011	NTF	Test flag not set	TF == 0
1100	LU	Latched underflow	LUF == 1
1101	LV	Latched overflow	LVF == 1
1110	UNC	Unconditional	None
1111	UNCF[2]	Unconditional with flag modification	None

(1) Values not shown are reserved.
(2) This is the default operation, if no CNDF field is specified. This condition allows the ZF and NF flags to be modified when a conditional operation is executed. All other conditions do not modify these flags.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	Yes	No	No	No	No

```
TF = 0;
if (CNDF == true) TF = 1;
```

Note: If (CNDF == UNC or UNCF), the TF flag is set to 1.

Pipeline

This is a single-cycle instruction.

MTESTTF CNDF (continued)

Test MSTF Register Flag Condition

Example

```
; if (State == 0.1)
;    RampState = RampState || RAMPMASK
; else if (State == 0.01)
;    CoastState = CoastState || COASTMASK
; else
;    SteadyState = SteadyState || STEADYMASK
;
_Cla1Task2:
   MMOV32    MR0, @_State
   MCMPF32   MR0, #0.1         ; Affects flags for 1st MBCNDD (A)
   MCMPF32   MR0, #0.01        ; Check used by 2nd MBCNDD (B)
   MTESTTF   EQ                ; Store EQ flag in TF for 2nd MBCNDD (B)
   MNOP
   MBCNDD    _Skip1, NEQ       ; (A) If State != 0.1, go to Skip1
   MMOV32    MR1, @_RampState  ; Always executed
   MMOVXI    MR2, #RAMPMASK    ; Always executed
   MOR32     MR1, MR2          ; Always executed
   MMOV32    @_RampState, MR1  ; Execute if (A) branch not taken
   MSTOP                       ; end of task if (A) branch not taken
_Skip1:
   MMOV32    MR3, @_SteadyState
   MMOVXI    MR2, #STEADYMASK
   MOR32     MR3, MR2
   MBCNDD    _Skip2, NTF       ; (B) if State != .01, go to Skip2
   MMOV32    MR1, @_CoastState ; Always executed
   MMOVXI    MR2, #COASTMASK   ; Always executed
   MOR32     MR1, MR2          ; Always executed
   MMOV32    @_CoastState, MR1 ; Execute if (B) branch not taken
   MSTOP                       ; end of task if (B) branch not taken
_Skip2:
   MMOV32 @_SteadyState, MR3   ; Executed if (B) branch taken
   MSTOP
```

MUI16TOF32 MRa, mem16

Convert Unsigned 16-Bit Integer to 32-Bit Floating-Point Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
mem16	16-bit source memory location

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0101 01aa addr
```

Description

When converting F32 to I16/UI16 data format, the MF32TOI16/UI16 operation truncates to zero while the MF32TOI16R/UI16R operation rounds to the nearest (even) value.

```
MRa = UI16TOF32[mem16];
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

See also

MF32TOI16 MRa, MRb
MF32TOI16R MRa, MRb
MF32TOUI16 MRa, MRb
MF32TOUI16R MRa, MRb
MI16TOF32 MRa, MRb
MI16TOF32 MRa, mem16
MUI16TOF32 MRa, MRb

MUI16TOF32 MRa, MRb

Convert Unsigned 16-Bit Integer to 32-Bit Floating-Point Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1110 1110 0000
```

Description

Convert an unsigned 16-bit integer to a 32-bit floating-point value. When converting float32 to I16/UI16 data format, the MF32TOI16/UI16 operation truncates to zero while the MF32TOI16R/UI16R operation rounds to the nearest (even) value.

```
MRa = UI16TOF32[MRb];
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MMOVXI MR1, #0x800F ; MR1(15:0) = 32783 (0x800F)
MUI16TOF32 MR0, MR1 ; MR0 = UI16TOF32 (MR1(15:0))
                    ; = 32783.0 (0x47000F00)
```

See also

MF32TOI16 MRa, MRb
MF32TOI16R MRa, MRb
MF32TOUI16 MRa, MRb
MF32TOUI16R MRa, MRb
MI16TOF32 MRa, MRb
MI16TOF32 MRa, mem16
MUI16TOF32 MRa, mem16

MUI32TOF32 MRa, mem32

Convert Unsigned 32-Bit Integer to 32-Bit Floating-Point Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
mem32	32-bit memory location accessed using one of the available addressing modes

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0100 10aa addr
```

Description

```
MRa = UI32TOF32[mem32];
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
; Given x2, m2, and b2 are Uint32 numbers:
;
; x2 = Uint32(2) = 0x00000002
; m2 = Uint32(1) = 0x00000001
; b2 = Uint32(3) = 0x00000003
;
; Calculate y2 = x2 * m2 + b2
;
_Cla1Task1:
   MUI32TOF32  MR0, @_m2      ; MR0 = 1.0 (0x3F800000)
   MUI32TOF32  MR1, @_x2      ; MR1 = 2.0 (0x40000000)
   MUI32TOF32  MR2, @_b2      ; MR2 = 3.0 (0x40400000)
   MMPYF32     MR3, MR0, MR1  ; M*X
   MADDF32     MR3, MR2, MR3  ; Y=MX+B = 5.0 (0x40A00000)
   MF32TOUI32  MR3, MR3       ; Y = Uint32(5.0) = 0x00000005
   MMOV32      @_y2, MR3      ; store result
   MSTOP                      ; end of task
```

See also

MF32TOI32 MRa, MRb
MF32TOUI32 MRa, MRb
MI32TOF32 MRa, mem32
MI32TOF32 MRa, MRb
MUI32TOF32 MRa, MRb

MUI32TOF32 MRa, MRb

Convert Unsigned 32-Bit Integer to 32-Bit Floating-Point Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1101 1100 0000
```

Description

```
MRa = UI32TOF32 [MRb];
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ     MR3, #0x8000 ; MR3(31:16) = 0x8000
MMOVXI     MR3, #0x1111 ; MR3(15:0) = 0x1111
                        ; MR3 = 2147488017
MUI32TOF32 MR3, MR3     ; MR3 = MUI32TOF32 (MR3) = 2147488017.0 (0x4F000011)
```

See also

MF32TOI32 MRa, MRb
MF32TOUI32 MRa, MRb
MI32TOF32 MRa, mem32
MI32TOF32 MRa, MRb
MUI32TOF32 MRa, mem32

MXOR32 MRa, MRb, MRc

Bitwise Exclusive Or

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)
MRc	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 00cc bbaa
MSW: 0111 1100 1010 0000
```

Description Bitwise XOR of MRb with MRc.

```
MARa(31:0) = MARb(31:0) XOR MRc(31:0);
```

Flags This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified based on the integer results of the operation.

```
NF = MRa(31);
ZF = 0;
if(MRa(31:0) == 0) { ZF = 1; }
```

Pipeline This is a single-cycle instruction.

Example

```
MMOVIZ MR0, #0x5555   ; MR0 = 0x5555AAAA
  MMOVXI MR0, #0xAAAA
  MMOVIZ MR1, #0x5432   ; MR1 = 0x5432FEDC
  MMOVXI MR1, #0xFEDC
  ; 0101 XOR 0101 = 0000 (0)
  ; 0101 XOR 0100 = 0001 (1)
  ; 0101 XOR 0011 = 0110 (6)
  ; 0101 XOR 0010 = 0111 (7)
  ; 1010 XOR 1111 = 0101 (5)
  ; 1010 XOR 1110 = 0100 (4)
  ; 1010 XOR 1101 = 0111 (7)
  ; 1010 XOR 1100 = 0110 (6)
  MXOR32 MR2, MR1, MR0  ; MR3 = 0x01675476
```

See also MAND32 MRa, MRb, MRc
MOR32 MRa, MRb, MRc

8.8 CLA Registers

This section describes the Control Law Accelerator registers.

8.8.1 CLA Base Address Table (C28)

Table 8-22. CLA Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
Cla1Regs	CLA_REGS	CLA1_BASE	0x0000_1400	YES	YES	-	-	-

8.8.2 CLA_ONLY_REGS Registers

Table 8-23 lists the memory-mapped registers for the CLA_ONLY_REGS registers. All register offset addresses not listed in Table 8-23 should be considered as reserved locations and the register contents should not be modified.

Table 8-23. CLA_ONLY_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
80h	_MVECTBGRNDACTIVE	Active register for MVECTBGRND.	EALLOW	Go
C0h	_MPSACTL	CLA PSA Control Register	EALLOW	Go
C2h	_MPSA1	CLA PSA1 Register	EALLOW	Go
C4h	_MPSA2	CLA PSA2 Register	EALLOW	Go
E0h	SOFTINTEN	CLA Software Interrupt Enable Register		Go
E2h	SOFTINTFRC	CLA Software Interrupt Force Register		Go

Complex bit access types are encoded to fit into small table cells. Table 8-24 shows the codes that are used for access types in this section.

Table 8-24. CLA_ONLY_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

8.8.2.1 _MVECTBGRNDACTIVE Register (Offset = 80h) [Reset = 0h]

_MVECTBGRNDACTIVE is shown in Figure 8-2 and described in Table 8-25.

Return to the Summary Table.

Gives the current interrupted MPC value of the background task, if the background task was running and interrupted, or reflects the MVECTBGRND value, if MCTLBGRND.BGSTART bit is 0.

Figure 8-2. _MVECTBGRNDACTIVE Register

15	14	13	12	11	10	9	8
i16							
R-0h							
7	6	5	4	3	2	1	0
i16							
R-0h							

Table 8-25. _MVECTBGRNDACTIVE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	i16	R	0h	Gives the current interrupted MPC value of the background task, if the background task was running and interrupted, or reflects the MVECTBGRND value, if MCTLBGRND.BGSTART bit is 0. Reset type: SYSRSn

8.8.2.2 _MPSACTL Register (Offset = C0h) [Reset = 0h]

_MPSACTL is shown in Figure 8-3 and described in Table 8-26.

Return to the Summary Table.

PSA Control Register

Figure 8-3. _MPSACTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
MPSA2CFG		MPSA2CLEAR	MPSA1CLEAR	MDWDBCYC	MDWDBSTART	MPABCYC	MPABSTART
R/W-0h		R-0/W1S-0h	R-0/W1S-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 8-26. _MPSACTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-6	MPSA2CFG	R/W	0h	CLA PSA2 Polynomial Configuration Bits: These bits configure the type of polynomial used for PSA2. The polynomials chosen are commonly used in the industry: Mode Polynomial Type 0,0 PSA 0,1 CRC32 1,0 CRC16 1,1 CRC16-CCITT Note: [1] Polynomial configuration should be performed when PSA2 is stopped. Reset type: SYSRSn
5	MPSA2CLEAR	R-0/W1S	0h	CLA PSA2 Clear Bit: Writing of "1" will clear contents of PSA2 register. Writes of "0" are ignored. Always reads back a "0" Note: Clearing operation should be performed when PSA2 is stopped. Reset type: SYSRSn
4	MPSA1CLEAR	R-0/W1S	0h	CLA PSA1 Clear Bit: Writing of "1" will clear contents of PSA1 register. Writes of "0" are ignored. Always reads back a "0" Note: Clearing operation should be performed when PSA1 is stopped. Reset type: SYSRSn
3	MDWDBCYC	R/W	0h	CLA Data Write Data Bus PSA2 Cycle or Event Based Bit: 0 PSA2 calculated on every cycle 1 PSA2 calculated on every bus event Reset type: SYSRSn
2	MDWDBSTART	R/W	0h	CLA Data Write Data Bus PSA2 Start/Stop Bit: 0 PSA2 stopped 1 PSA2 start Reset type: SYSRSn
1	MPABCYC	R/W	0h	CLA Program Address Bus PSA1 Cycle/Event Based Bit: 0 PSA1 calculated on every cycle 1 PSA1 calculated on every bus event Reset type: SYSRSn
0	MPABSTART	R/W	0h	CLA Program Address Bus PSA1 Start/Stop Bit: 0 PSA1 stopped 1 PSA1 start Reset type: SYSRSn

8.8.2.3 _MPSA1 Register (Offset = C2h) [Reset = 0h]

_MPSA1 is shown in Figure 8-4 and described in Table 8-27.

Return to the Summary Table.

PSA1 Register

Figure 8-4. _MPSA1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
i32																															
R/W-0h																															

Table 8-27. _MPSA1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	i32	R/W	0h	PSA1 Value: Reading this register gives the current PSA1 value. The value can be read at any time. Writes to this register are allowed to initialize the PSA1 to a known value. Writes to this register should only be made when PSA1 is stopped. Register value is cleared to zero by reset or by writing to the MPSA1CLEAR bit in the MPSACTL register. Reset type: SYSRSn

8.8.2.4 _MPSA2 Register (Offset = C4h) [Reset = 0h]

_MPSA2 is shown in Figure 8-5 and described in Table 8-28.

Return to the Summary Table.

PSA2 Register

Figure 8-5. _MPSA2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
i32																															
R/W-0h																															

Table 8-28. _MPSA2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	i32	R/W	0h	PSA2 Value: Reading this register gives the current PSA2 value. The value can be read at any time. Writes to this register are allowed to initialize the PSA2 to a known value. Writes to this register should only be made when PSA2 is stopped. Register value is cleared to zero by reset or by writing to the MPSA2CLEAR bit in the MPSACTL register. Reset type: SYSRSn

8.8.2.5 SOFTINTEN Register (Offset = E0h) [Reset = 0h]

SOFTINTEN is shown in Figure 8-6 and described in Table 8-29.

Return to the Summary Table.

Enables the ability to generate CLA task interrupt from within the task, by writing to SOFTINTFRC register. SOFTINTFRC register can only be written from CLA.

Figure 8-6. SOFTINTEN Register

15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
TASK8	TASK7	TASK6	TASK5	TASK4	TASK3	TASK2	TASK1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 8-29. SOFTINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R/W	0h	Reserved
7	TASK8	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
6	TASK7	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
5	TASK6	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
4	TASK5	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
3	TASK4	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
2	TASK3	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
1	TASK2	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn

Table 8-29. SOFTINTEN Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	TASK1	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn

8.8.2.6 SOFTINTFRC Register (Offset = E2h) [Reset = 0h]

SOFTINTFRC is shown in Figure 8-7 and described in Table 8-30.

Return to the Summary Table.

Writing a value of 1 in a bit will generate the corresponding task interrupt.

Figure 8-7. SOFTINTFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
TASK8	TASK7	TASK6	TASK5	TASK4	TASK3	TASK2	TASK1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 8-30. SOFTINTFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R/W	0h	Reserved
7	TASK8	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
6	TASK7	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
5	TASK6	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
4	TASK5	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
3	TASK4	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
2	TASK3	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
1	TASK2	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
0	TASK1	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn

8.8.3 CLA_SOFTINT_REGS Registers

Table 8-31 lists the memory-mapped registers for the CLA_SOFTINT_REGS registers. All register offset addresses not listed in Table 8-31 should be considered as reserved locations and the register contents should not be modified.

Table 8-31. CLA_SOFTINT_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	SOFTINTEN	CLA Software Interrupt Enable Register		Go
2h	SOFTINTFRC	CLA Software Interrupt Force Register		Go

Complex bit access types are encoded to fit into small table cells. Table 8-32 shows the codes that are used for access types in this section.

Table 8-32. CLA_SOFTINT_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

8.8.3.1 SOFTINTEN Register (Offset = 0h) [Reset = 0h]

SOFTINTEN is shown in Figure 8-8 and described in Table 8-33.

Return to the Summary Table.

Enables the ability to generate CLA task interrupt from within the task, by writing to SOFTINTFRC register. SOFTINTFRC register can only be written from CLA.

Figure 8-8. SOFTINTEN Register

15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
TASK8	TASK7	TASK6	TASK5	TASK4	TASK3	TASK2	TASK1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 8-33. SOFTINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R/W	0h	Reserved
7	TASK8	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
6	TASK7	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
5	TASK6	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
4	TASK5	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
3	TASK4	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
2	TASK3	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
1	TASK2	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn

Table 8-33. SOFTINTEN Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	TASK1	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn

8.8.3.2 SOFTINTFRC Register (Offset = 2h) [Reset = 0h]

SOFTINTFRC is shown in Figure 8-9 and described in Table 8-34.

Return to the Summary Table.

Writing a value of 1 in a bit will generate the corresponding task interrupt.This register is only accessible by the CLA (not the CPU).

Figure 8-9. SOFTINTFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
TASK8	TASK7	TASK6	TASK5	TASK4	TASK3	TASK2	TASK1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 8-34. SOFTINTFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R/W	0h	Reserved
7	TASK8	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
6	TASK7	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
5	TASK6	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
4	TASK5	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
3	TASK4	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
2	TASK3	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
1	TASK2	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
0	TASK1	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn

8.8.4 CLA_REGS Registers

Table 8-35 lists the memory-mapped registers for the CLA_REGS registers. All register offset addresses not listed in Table 8-35 should be considered as reserved locations and the register contents should not be modified.

Table 8-35. CLA_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	MVECT1	Task Interrupt Vector	EALLOW	Go
1h	MVECT2	Task Interrupt Vector	EALLOW	Go
2h	MVECT3	Task Interrupt Vector	EALLOW	Go
3h	MVECT4	Task Interrupt Vector	EALLOW	Go
4h	MVECT5	Task Interrupt Vector	EALLOW	Go
5h	MVECT6	Task Interrupt Vector	EALLOW	Go
6h	MVECT7	Task Interrupt Vector	EALLOW	Go
7h	MVECT8	Task Interrupt Vector	EALLOW	Go
10h	MCTL	Control Register	EALLOW	Go
1Bh	_MVECTBGRNDACTIVE	Active register for MVECTBGRND.	EALLOW	Go
1Ch	SOFTINTEN	CLA Software Interrupt Enable Register		Go
1Dh	_MSTSBGRND	Status register for the back ground task.	EALLOW	Go
1Eh	_MCTLBGRND	Control register for the back ground task.	EALLOW	Go
1Fh	_MVECTBGRND	Vector for the back ground task.	EALLOW	Go
20h	MIFR	Interrupt Flag Register	EALLOW	Go
21h	MIOVF	Interrupt Overflow Flag Register	EALLOW	Go
22h	MIFRC	Interrupt Force Register	EALLOW	Go
23h	MICLR	Interrupt Flag Clear Register	EALLOW	Go
24h	MICLROVF	Interrupt Overflow Flag Clear Register	EALLOW	Go
25h	MIER	Interrupt Enable Register	EALLOW	Go
26h	MIRUN	Interrupt Run Status Register	EALLOW	Go
28h	_MPC	CLA Program Counter		Go
2Ah	_MAR0	CLA Auxiliary Register 0		Go
2Bh	_MAR1	CLA Auxiliary Register 1		Go
2Eh	_MSTF	CLA Floating-Point Status Register		Go
30h	_MR0	CLA Floating-Point Result Register 0		Go
34h	_MR1	CLA Floating-Point Result Register 1		Go
38h	_MR2	CLA Floating-Point Result Register 2		Go
3Ch	_MR3	CLA Floating-Point Result Register 3		Go
42h	_MPSACTL	CLA PSA Control Register	EALLOW	Go
44h	_MPSA1	CLA PSA1 Register	EALLOW	Go
46h	_MPSA2	CLA PSA2 Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 8-36 shows the codes that are used for access types in this section.

Table 8-36. CLA_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		

Table 8-36. CLA_REGS Access Type Codes (continued)

Access Type	Code	Description
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

8.8.4.1 MVECT1 Register (Offset = 0h) [Reset = 0h]

MVECT1 is shown in Figure 8-10 and described in Table 8-37.

Return to the Summary Table.

Each CLA interrupt has its own interrupt vector (MVECT1 to MVECT8). This interrupt vector points to the first instruction of the associated task. When a task begins, the CLA will start fetching instructions at the location indicated by the appropriate MVECT register .

Figure 8-10. MVECT1 Register

15	14	13	12	11	10	9	8
MVECT							
R/W-0h							
7	6	5	4	3	2	1	0
MVECT							
R/W-0h							

Table 8-37. MVECT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	MVECT	R/W	0h	MPC Start Address: These bits specify the start address for the given interrupt (task). The address range of the CLA with a 16-bit MVECT is 64Kx16 words or 32K CLA instructions. There is one MVECT register per interrupt (task). Interrupt 1 uses MVECT1, interrupt 2 uses MVECT2 and so forth. Note: While the CLA is running or executing a task, the CPU can change the MVECT values.. Reset type: SYSRSn

8.8.4.2 MVECT2 Register (Offset = 1h) [Reset = 0h]

MVECT2 is shown in Figure 8-11 and described in Table 8-38.

Return to the Summary Table.

Each CLA interrupt has its own interrupt vector (MVECT1 to MVECT8). This interrupt vector points to the first instruction of the associated task. When a task begins, the CLA will start fetching instructions at the location indicated by the appropriate MVECT register .

Figure 8-11. MVECT2 Register

15	14	13	12	11	10	9	8
MVECT							
R/W-0h							
7	6	5	4	3	2	1	0
MVECT							
R/W-0h							

Table 8-38. MVECT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	MVECT	R/W	0h	MPC Start Address: These bits specify the start address for the given interrupt (task). The address range of the CLA with a 16-bit MVECT is 64Kx16 words or 32K CLA instructions. There is one MVECT register per interrupt (task). Interrupt 1 uses MVECT1, interrupt 2 uses MVECT2 and so forth. Note: While the CLA is running or executing a task, the CPU can change the MVECT values.. Reset type: SYSRSn

8.8.4.3 MVECT3 Register (Offset = 2h) [Reset = 0h]

MVECT3 is shown in Figure 8-12 and described in Table 8-39.

Return to the Summary Table.

Each CLA interrupt has its own interrupt vector (MVECT1 to MVECT8). This interrupt vector points to the first instruction of the associated task. When a task begins, the CLA will start fetching instructions at the location indicated by the appropriate MVECT register .

Figure 8-12. MVECT3 Register

15	14	13	12	11	10	9	8
MVECT							
R/W-0h							
7	6	5	4	3	2	1	0
MVECT							
R/W-0h							

Table 8-39. MVECT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	MVECT	R/W	0h	MPC Start Address: These bits specify the start address for the given interrupt (task). The address range of the CLA with a 16-bit MVECT is 64Kx16 words or 32K CLA instructions. There is one MVECT register per interrupt (task). Interrupt 1 uses MVECT1, interrupt 2 uses MVECT2 and so forth. Note: While the CLA is running or executing a task, the CPU can change the MVECT values.. Reset type: SYSRSn

8.8.4.4 MVECT4 Register (Offset = 3h) [Reset = 0h]

MVECT4 is shown in Figure 8-13 and described in Table 8-40.

Return to the Summary Table.

Each CLA interrupt has its own interrupt vector (MVECT1 to MVECT8). This interrupt vector points to the first instruction of the associated task. When a task begins, the CLA will start fetching instructions at the location indicated by the appropriate MVECT register .

Figure 8-13. MVECT4 Register

15	14	13	12	11	10	9	8
MVECT							
R/W-0h							
7	6	5	4	3	2	1	0
MVECT							
R/W-0h							

Table 8-40. MVECT4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	MVECT	R/W	0h	MPC Start Address: These bits specify the start address for the given interrupt (task). The address range of the CLA with a 16-bit MVECT is 64Kx16 words or 32K CLA instructions. There is one MVECT register per interrupt (task). Interrupt 1 uses MVECT1, interrupt 2 uses MVECT2 and so forth. Note: While the CLA is running or executing a task, the CPU can change the MVECT values.. Reset type: SYSRSn

8.8.4.5 MVECT5 Register (Offset = 4h) [Reset = 0h]

MVECT5 is shown in Figure 8-14 and described in Table 8-41.

Return to the Summary Table.

Each CLA interrupt has its own interrupt vector (MVECT1 to MVECT8). This interrupt vector points to the first instruction of the associated task. When a task begins, the CLA will start fetching instructions at the location indicated by the appropriate MVECT register .

Figure 8-14. MVECT5 Register

15	14	13	12	11	10	9	8
MVECT							
R/W-0h							
7	6	5	4	3	2	1	0
MVECT							
R/W-0h							

Table 8-41. MVECT5 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	MVECT	R/W	0h	MPC Start Address: These bits specify the start address for the given interrupt (task). The address range of the CLA with a 16-bit MVECT is 64Kx16 words or 32K CLA instructions. There is one MVECT register per interrupt (task). Interrupt 1 uses MVECT1, interrupt 2 uses MVECT2 and so forth. Note: While the CLA is running or executing a task, the CPU can change the MVECT values.. Reset type: SYSRSn

8.8.4.6 MVECT6 Register (Offset = 5h) [Reset = 0h]

MVECT6 is shown in Figure 8-15 and described in Table 8-42.

Return to the Summary Table.

Each CLA interrupt has its own interrupt vector (MVECT1 to MVECT8). This interrupt vector points to the first instruction of the associated task. When a task begins, the CLA will start fetching instructions at the location indicated by the appropriate MVECT register .

Figure 8-15. MVECT6 Register

15	14	13	12	11	10	9	8
MVECT							
R/W-0h							
7	6	5	4	3	2	1	0
MVECT							
R/W-0h							

Table 8-42. MVECT6 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	MVECT	R/W	0h	MPC Start Address: These bits specify the start address for the given interrupt (task). The address range of the CLA with a 16-bit MVECT is 64Kx16 words or 32K CLA instructions. There is one MVECT register per interrupt (task). Interrupt 1 uses MVECT1, interrupt 2 uses MVECT2 and so forth. Note: While the CLA is running or executing a task, the CPU can change the MVECT values.. Reset type: SYSRSn

8.8.4.7 MVECT7 Register (Offset = 6h) [Reset = 0h]

MVECT7 is shown in Figure 8-16 and described in Table 8-43.

Return to the Summary Table.

Each CLA interrupt has its own interrupt vector (MVECT1 to MVECT8). This interrupt vector points to the first instruction of the associated task. When a task begins, the CLA will start fetching instructions at the location indicated by the appropriate MVECT register .

Figure 8-16. MVECT7 Register

15	14	13	12	11	10	9	8
MVECT							
R/W-0h							
7	6	5	4	3	2	1	0
MVECT							
R/W-0h							

Table 8-43. MVECT7 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	MVECT	R/W	0h	MPC Start Address: These bits specify the start address for the given interrupt (task). The address range of the CLA with a 16-bit MVECT is 64Kx16 words or 32K CLA instructions. There is one MVECT register per interrupt (task). Interrupt 1 uses MVECT1, interrupt 2 uses MVECT2 and so forth. Note: While the CLA is running or executing a task, the CPU can change the MVECT values.. Reset type: SYSRSn

8.8.4.8 MVECT8 Register (Offset = 7h) [Reset = 0h]

MVECT8 is shown in Figure 8-17 and described in Table 8-44.

Return to the Summary Table.

Each CLA interrupt has its own interrupt vector (MVECT1 to MVECT8). This interrupt vector points to the first instruction of the associated task. When a task begins, the CLA will start fetching instructions at the location indicated by the appropriate MVECT register .

Figure 8-17. MVECT8 Register

15	14	13	12	11	10	9	8
MVECT							
R/W-0h							
7	6	5	4	3	2	1	0
MVECT							
R/W-0h							

Table 8-44. MVECT8 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	MVECT	R/W	0h	MPC Start Address: These bits specify the start address for the given interrupt (task). The address range of the CLA with a 16-bit MVECT is 64Kx16 words or 32K CLA instructions. There is one MVECT register per interrupt (task). Interrupt 1 uses MVECT1, interrupt 2 uses MVECT2 and so forth. Note: While the CLA is running or executing a task, the CPU can change the MVECT values.. Reset type: SYSRSn

8.8.4.9 MCTL Register (Offset = 10h) [Reset = 0h]

MCTL is shown in Figure 8-18 and described in Table 8-45.

Return to the Summary Table.

Control Register

Figure 8-18. MCTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					IACKE	SOFTRESET	HARDRESET
R-0h					R/W-0h	R-0/W1S-0h	R-0/W1S-0h

Table 8-45. MCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	Reserved
2	IACKE	R/W	0h	IACK Operation Enable Bit: Writing a "1" to this bit will enable the IACK operation for setting the MIFR bits in the same manner as the MIFRC register (write of "1" will set respective MIFR bit). At reset, this feature is disabled. This feature enables the C28 CPU to efficiently trigger a task. Note: IACK operation should ignore EALLOW status of C28 core when accessing the MIFRC register. Reset type: SYSRSn 0h (R/W) = The CLA ignores the IACK instruction. (default) 1h (R/W) = Enable the main CPU to use the IACK #16bit instruction to set MIFR bits in the same manner as writing to the MIFRC register. Each bit in the operand, #16bit, corresponds to a bit in the MIFRC register. Using IACK has the advantage of not having to first set the EALLOW bit. This allows the main CPU to efficiently trigger a CLA task through software. Examples IACK #0x0001 Write a 1 to MIFRC bit 0 to force task 1 IACK #0x0003 Write a 1 to MIFRC bit 0 and 1 to force task 1 and task 2
1	SOFTRESET	R-0/W1S	0h	Soft Reset Bit: Writing a "1" to this bit will stop a current task, clear the RUN flag and also clear all bits in the MIER register. Writes of "0" are ignored and reads always return a "0". Note: After issuing SOFTRESET command, user should wait at least 1 clock cycle before attempting to write to MIER register. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 are ignored. 1h (R/W) = Writing a 1 will cause a soft reset of the CLA. This will stop the current task, clear the MIRUN flag and clear all bits in the MIER register. After a soft reset you must wait at least 1 SYSCLKOUT cycle before reconfiguring the MIER bits. If these two operations are done back-to-back then the MIER bits will not get set.
0	HARDRESET	R-0/W1S	0h	Hard Reset Bit: Writing a "1" to this bit will cause a HARD reset on the CLA. The behavior of a HARD reset is the same as a system reset SYSRSn on the CLA. Writes of "0" are ignored and reads always return a "0". Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 are ignored. 1h (R/W) = Writing a 1 will cause a hard reset of the CLA. This will set all CLA registers to their default state.

8.8.4.10 _MVECTBGRNDACTIVE Register (Offset = 1Bh) [Reset = 0h]

_MVECTBGRNDACTIVE is shown in Figure 8-19 and described in Table 8-46.

Return to the Summary Table.

Gives the current interrupted MPC value of the background task, if the background task was running and interrupted, or reflects the MVECTBGRND value, if MCTLBGRND.BGSTART bit is 0.

Figure 8-19. _MVECTBGRNDACTIVE Register

15	14	13	12	11	10	9	8
i16							
R-0h							
7	6	5	4	3	2	1	0
i16							
R-0h							

Table 8-46. _MVECTBGRNDACTIVE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	i16	R	0h	Gives the current interrupted MPC value of the background task, if the background task was running and interrupted, or reflects the MVECTBGRND value, if MCTLBGRND.BGSTART bit is 0. Reset type: SYSRSn

8.8.4.11 SOFTINTEN Register (Offset = 1Ch) [Reset = 0h]

SOFTINTEN is shown in Figure 8-20 and described in Table 8-47.

Return to the Summary Table.

Enables the ability to generate CLA task interrupt from within the task, by writing to SOFTINTFRC register. SOFTINTFRC register can only be written from CLA. Only reads are allowed from CPU. Writes are not allowed from CPU.

Figure 8-20. SOFTINTEN Register

15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
TASK8	TASK7	TASK6	TASK5	TASK4	TASK3	TASK2	TASK1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 8-47. SOFTINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R/W	0h	Reserved
7	TASK8	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
6	TASK7	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
5	TASK6	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
4	TASK5	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
3	TASK4	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
2	TASK3	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
1	TASK2	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn

Table 8-47. SOFTINTEN Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	TASK1	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn

8.8.4.12 _MSTSBGRND Register (Offset = 1Dh) [Reset = 0h]

_MSTSBGRND is shown in Figure 8-21 and described in Table 8-48.

Return to the Summary Table.

Status bits for the backgrondtask.

Figure 8-21. _MSTSBGRND Register

15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED					BGOVF	_BGINTM	RUN
R/W-0h					R/W1C-0h	R-0h	R-0h

Table 8-48. _MSTSBGRND Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R/W	0h	Reserved
2	BGOVF	R/W1C	0h	Value of 1 indicates a hardware trigger (which is enabled) occurred while the MCTLBGRND.BGSTART bit is set. Writing a value of 1 to this bit clears the BGOVF bit. Write of 0 has no effect, Value of 0 indicates the background task trigger did not result in a overflow. Reset type: SYSRSn
1	_BGINTM	R	0h	Value of 1 indicates that backgroiund task will not be interrupted. This bit is set when MSETC _BGINTM bit is executed. Value of 0 indicates that background task can be interrupted. Reset type: SYSRSn
0	RUN	R	0h	Value of 1 indicates that background task is running. Value of 0 indicates that background task is not running. Reset type: SYSRSn

8.8.4.13 _MCTLBGRND Register (Offset = 1Eh) [Reset = 0h]

_MCTLBGRND is shown in Figure 8-22 and described in Table 8-49.

Return to the Summary Table.

Holds the configuration bits to start the background task, enable hardware trigger.

Figure 8-22. _MCTLBGRND Register

15	14	13	12	11	10	9	8
BGEN	RESERVED						
R/W-0h	R/W-0h						
7	6	5	4	3	2	1	0
RESERVED						TRIGEN	BGSTART
R/W-0h						R/W-0h	R/W1S-0h

Table 8-49. _MCTLBGRND Register Field Descriptions

Bit	Field	Type	Reset	Description
15	BGEN	R/W	0h	0 Background task is disabled, BGSTART will not be set either in a hardware trigger or by writing 1 to BGSTART bit. 1 Background task is enabled and MIER[INT8] will be cleared, preventing task 8 from triggering. Reset type: SYSRSn
14-2	RESERVED	R/W	0h	Reserved
1	TRIGEN	R/W	0h	Hardware trigger enable for the background task. 1 Hardware trigger is enabled. 0 Hardware trigger is disabled. Note: Trigger source for the background task will be the same as that for task 8 Reset type: SYSRSn
0	BGSTART	R/W1S	0h	Value of 1 will start the background task, provided there are no other pending tasks. - Value of 0 has no effect if the background task has not started. - This bit is also set by hardware, if MCTLBGRND.TRIGEN = 1 and a hardware trigger occurs. - This bit is cleared by hardware when a MSTOP instruction occurs in the background task - If the background task is running and this bit is cleared, it will not have any effect on the task execution. Reset type: SYSRSn

8.8.4.14 _MVECTBGRND Register (Offset = 1Fh) [Reset = 0h]

_MVECTBGRND is shown in Figure 8-23 and described in Table 8-50.

Return to the Summary Table.

These bits specify the start address for the background task . The value in this register is forced into the MPC register when the background task starts.

Figure 8-23. _MVECTBGRND Register

15	14	13	12	11	10	9	8
i16							
R/W-0h							
7	6	5	4	3	2	1	0
i16							
R/W-0h							

Table 8-50. _MVECTBGRND Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	i16	R/W	0h	MPC Start Address: These bits specify the start address for the background task . The value in this register is forced into the MPC register, when the background task starts. Reset type: SYSRSn

8.8.4.15 MIFR Register (Offset = 20h) [Reset = 0h]

MIFR is shown in Figure 8-24 and described in Table 8-51.

Return to the Summary Table.

Each bit in the interrupt flag register corresponds to a CLA task. The corresponding bit is automatically set when the task request is received from the peripheral interrupt. The bit can also be set by the main CPU writing to the MIFRC register or using the IACK instruction to start the task. To use the IACK instruction to begin a task first enable this feature in the MCTL register. If the bit is already set when a new peripheral interrupt is received, then the corresponding overflow bit will be set in the MIOVF register.
The corresponding MIFR bit is automatically cleared when the task begins execution. This will occur if the interrupt is enabled in the MIER register and no other higher priority task is pending. The bits can also be cleared manually by writing to the MICLR register. Writes to the MIFR register are ignored.

Figure 8-24. MIFR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 8-51. MIFR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	INT8	R	0h	These bits, when set to "1", indicate a valid peripheral interrupt has been latched by the CLA. Writes to this register are ignored. The IFR flag bit is automatically cleared if the respective interrupt is enabled in the MIER register and the respective task starts running. If a new peripheral interrupt attempts to set the bit to "1" while on the same cycle the task tries to clear it, then the peripheral interrupt will have priority. The IFR flag bits can also be set and cleared by the MIFRC and MICLR registers. If the MIFRC register is trying to set the respective bit while a new task tries to clear it, then the MIFRC event has priority. If the MICLR register is trying to clear the respective bit and a peripheral interrupt occurs on the same cycle, then the peripheral interrupt has priority. The respective overflow flag in the MIOVF register will not be set under this condition. Reset type: SYSRSn 0h (R/W) = TASK_FLAG_DISABLE Task 8 interrupt is currently not flagged (default) 1h (R/W) = TASK_FLAG_ENABLE Task 8 interrupt has been received and is pending execution

Table 8-51. MIFR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	INT7	R	0h	These bits, when set to "1", indicate a valid peripheral interrupt has been latched by the CLA. Writes to this register are ignored. The IFR flag bit is automatically cleared if the respective interrupt is enabled in the MIER register and the respective task starts running. If a new peripheral interrupt attempts to set the bit to "1" while on the same cycle the task tries to clear it, then the peripheral interrupt will have priority. The IFR flag bits can also be set and cleared by the MIFRC and MICLR registers. If the MIFRC register is trying to set the respective bit while a new task tries to clear it, then the MIFRC event has priority. If the MICLR register is trying to clear the respective bit and a peripheral interrupt occurs on the same cycle, then the peripheral interrupt has priority. The respective overflow flag in the MIOVF register will not be set under this condition. Reset type: SYSRSn 0h (R/W) = TASK_FLAG_DISABLE Task 7 interrupt is currently not flagged (default) 1h (R/W) = TASK_FLAG_ENABLE Task 7 interrupt has been received and is pending execution
5	INT6	R	0h	These bits, when set to "1", indicate a valid peripheral interrupt has been latched by the CLA. Writes to this register are ignored. The IFR flag bit is automatically cleared if the respective interrupt is enabled in the MIER register and the respective task starts running. If a new peripheral interrupt attempts to set the bit to "1" while on the same cycle the task tries to clear it, then the peripheral interrupt will have priority. The IFR flag bits can also be set and cleared by the MIFRC and MICLR registers. If the MIFRC register is trying to set the respective bit while a new task tries to clear it, then the MIFRC event has priority. If the MICLR register is trying to clear the respective bit and a peripheral interrupt occurs on the same cycle, then the peripheral interrupt has priority. The respective overflow flag in the MIOVF register will not be set under this condition. Reset type: SYSRSn 0h (R/W) = TASK_FLAG_DISABLE Task 6 interrupt is currently not flagged (default) 1h (R/W) = TASK_FLAG_ENABLE Task 6 interrupt has been received and is pending execution
4	INT5	R	0h	These bits, when set to "1", indicate a valid peripheral interrupt has been latched by the CLA. Writes to this register are ignored. The IFR flag bit is automatically cleared if the respective interrupt is enabled in the MIER register and the respective task starts running. If a new peripheral interrupt attempts to set the bit to "1" while on the same cycle the task tries to clear it, then the peripheral interrupt will have priority. The IFR flag bits can also be set and cleared by the MIFRC and MICLR registers. If the MIFRC register is trying to set the respective bit while a new task tries to clear it, then the MIFRC event has priority. If the MICLR register is trying to clear the respective bit and a peripheral interrupt occurs on the same cycle, then the peripheral interrupt has priority. The respective overflow flag in the MIOVF register will not be set under this condition. Reset type: SYSRSn 0h (R/W) = TASK_FLAG_DISABLE Task 5 interrupt is currently not flagged (default) 1h (R/W) = TASK_FLAG_ENABLE Task 5 interrupt has been received and is pending execution

Table 8-51. MIFR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	INT4	R	0h	These bits, when set to "1", indicate a valid peripheral interrupt has been latched by the CLA. Writes to this register are ignored. The IFR flag bit is automatically cleared if the respective interrupt is enabled in the MIER register and the respective task starts running. If a new peripheral interrupt attempts to set the bit to "1" while on the same cycle the task tries to clear it, then the peripheral interrupt will have priority. The IFR flag bits can also be set and cleared by the MIFRC and MICLR registers. If the MIFRC register is trying to set the respective bit while a new task tries to clear it, then the MIFRC event has priority. If the MICLR register is trying to clear the respective bit and a peripheral interrupt occurs on the same cycle, then the peripheral interrupt has priority. The respective overflow flag in the MIOVF register will not be set under this condition. Reset type: SYSRSn 0h (R/W) = TASK_FLAG_DISABLE Task 4 interrupt is currently not flagged (default) 1h (R/W) = TASK_FLAG_ENABLE Task 4 interrupt has been received and is pending execution
2	INT3	R	0h	These bits, when set to "1", indicate a valid peripheral interrupt has been latched by the CLA. Writes to this register are ignored. The IFR flag bit is automatically cleared if the respective interrupt is enabled in the MIER register and the respective task starts running. If a new peripheral interrupt attempts to set the bit to "1" while on the same cycle the task tries to clear it, then the peripheral interrupt will have priority. The IFR flag bits can also be set and cleared by the MIFRC and MICLR registers. If the MIFRC register is trying to set the respective bit while a new task tries to clear it, then the MIFRC event has priority. If the MICLR register is trying to clear the respective bit and a peripheral interrupt occurs on the same cycle, then the peripheral interrupt has priority. The respective overflow flag in the MIOVF register will not be set under this condition. Reset type: SYSRSn 0h (R/W) = TASK_FLAG_DISABLE Task 3 interrupt is currently not flagged (default) 1h (R/W) = TASK_FLAG_ENABLE Task 3 interrupt has been received and is pending execution
1	INT2	R	0h	These bits, when set to "1", indicate a valid peripheral interrupt has been latched by the CLA. Writes to this register are ignored. The IFR flag bit is automatically cleared if the respective interrupt is enabled in the MIER register and the respective task starts running. If a new peripheral interrupt attempts to set the bit to "1" while on the same cycle the task tries to clear it, then the peripheral interrupt will have priority. The IFR flag bits can also be set and cleared by the MIFRC and MICLR registers. If the MIFRC register is trying to set the respective bit while a new task tries to clear it, then the MIFRC event has priority. If the MICLR register is trying to clear the respective bit and a peripheral interrupt occurs on the same cycle, then the peripheral interrupt has priority. The respective overflow flag in the MIOVF register will not be set under this condition. Reset type: SYSRSn 0h (R/W) = TASK_FLAG_DISABLE Task 2 interrupt is currently not flagged (default) 1h (R/W) = TASK_FLAG_ENABLE Task 2 interrupt has been received and is pending execution

Table 8-51. MIFR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INT1	R	0h	These bits, when set to "1", indicate a valid peripheral interrupt has been latched by the CLA. Writes to this register are ignored. The IFR flag bit is automatically cleared if the respective interrupt is enabled in the MIER register and the respective task starts running. If a new peripheral interrupt attempts to set the bit to "1" while on the same cycle the task tries to clear it, then the peripheral interrupt will have priority. The IFR flag bits can also be set and cleared by the MIFRC and MICLR registers. If the MIFRC register is trying to set the respective bit while a new task tries to clear it, then the MIFRC event has priority. If the MICLR register is trying to clear the respective bit and a peripheral interrupt occurs on the same cycle, then the peripheral interrupt has priority. The respective overflow flag in the MIOVF register will not be set under this condition. Reset type: SYSRSn 0h (R/W) = TASK_FLAG_DISABLE Task 1 interrupt is currently not flagged (default) 1h (R/W) = TASK_FLAG_ENABLE Task 1 interrupt has been received and is pending execution

8.8.4.16 MIOVF Register (Offset = 21h) [Reset = 0h]

MIOVF is shown in Figure 8-25 and described in Table 8-52.

Return to the Summary Table.

Each bit in the overflow flag register corresponds to a CLA task. The bit is set when an interrupt overflow event has occurred for the specific task. An overflow event occurs when the MIFR register bit is already set when a new interrupt is received from a peripheral source. The MIOVF bits are only affected by peripheral interrupt events. They do not respond to a task request by the main CPU IACK instruction or by directly setting MIFR bits. The overflow flag will remain latched and can only be cleared by writing to the overflow flag clear (MICLROVF) register. Writes to the MIOVF register are ignored.

Figure 8-25. MIOVF Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 8-52. MIOVF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	INT8	R	0h	These bits, when set to "1", indicate an interrupt overflow event occurred. Such an event occurs when the IFR bit is already set. An overflow event remains latched and respective bits can only be cleared by writing to the MICLROVF register. If the MIFR bit is being cleared by a new task on the same cycle as a new peripheral interrupt occurs, the overflow flag will not be affected and the respective MIFR bit will be set. If the MIOVF bit is being cleared by the MICLROVF register on the same cycle as the overflow bit is being set by hardware, then the hardware will have priority. Notes: [1] The MIOVF bits are only affected by peripheral interrupt events. Forcing an interrupt using the MIFRC or IACK operation will not set the overflow flag even if the MIFR bit is set. Reset type: SYSRSn 0h (R/W) = A task 8 interrupt overflow has not occurred (default) 1h (R/W) = A task 8 interrupt overflow has occurred
6	INT7	R	0h	These bits, when set to "1", indicate an interrupt overflow event occurred. Such an event occurs when the IFR bit is already set. An overflow event remains latched and respective bits can only be cleared by writing to the MICLROVF register. If the MIFR bit is being cleared by a new task on the same cycle as a new peripheral interrupt occurs, the overflow flag will not be affected and the respective MIFR bit will be set. If the MIOVF bit is being cleared by the MICLROVF register on the same cycle as the overflow bit is being set by hardware, then the hardware will have priority. Notes: [1] The MIOVF bits are only affected by peripheral interrupt events. Forcing an interrupt using the MIFRC or IACK operation will not set the overflow flag even if the MIFR bit is set. Reset type: SYSRSn 0h (R/W) = A task 7 interrupt overflow has not occurred (default) 1h (R/W) = A task 7 interrupt overflow has occurred

Table 8-52. MIOVF Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	INT6	R	0h	These bits, when set to "1", indicate an interrupt overflow event occurred. Such an event occurs when the IFR bit is already set. An overflow event remains latched and respective bits can only be cleared by writing to the MICLROVF register. If the MIFR bit is being cleared by a new task on the same cycle as a new peripheral interrupt occurs, the overflow flag will not be affected and the respective MIFR bit will be set. If the MIOVF bit is being cleared by the MICLROVF register on the same cycle as the overflow bit is being set by hardware, then the hardware will have priority. Notes: [1] The MIOVF bits are only affected by peripheral interrupt events. Forcing an interrupt using the MIFRC or IACK operation will not set the overflow flag even if the MIFR bit is set. Reset type: SYSRSn 0h (R/W) = A task 6 interrupt overflow has not occurred (default) 1h (R/W) = A task 6 interrupt overflow has occurred
4	INT5	R	0h	These bits, when set to "1", indicate an interrupt overflow event occurred. Such an event occurs when the IFR bit is already set. An overflow event remains latched and respective bits can only be cleared by writing to the MICLROVF register. If the MIFR bit is being cleared by a new task on the same cycle as a new peripheral interrupt occurs, the overflow flag will not be affected and the respective MIFR bit will be set. If the MIOVF bit is being cleared by the MICLROVF register on the same cycle as the overflow bit is being set by hardware, then the hardware will have priority. Notes: [1] The MIOVF bits are only affected by peripheral interrupt events. Forcing an interrupt using the MIFRC or IACK operation will not set the overflow flag even if the MIFR bit is set. Reset type: SYSRSn 0h (R/W) = A task 5 interrupt overflow has not occurred (default) 1h (R/W) = A task 5 interrupt overflow has occurred
3	INT4	R	0h	These bits, when set to "1", indicate an interrupt overflow event occurred. Such an event occurs when the IFR bit is already set. An overflow event remains latched and respective bits can only be cleared by writing to the MICLROVF register. If the MIFR bit is being cleared by a new task on the same cycle as a new peripheral interrupt occurs, the overflow flag will not be affected and the respective MIFR bit will be set. If the MIOVF bit is being cleared by the MICLROVF register on the same cycle as the overflow bit is being set by hardware, then the hardware will have priority. Notes: [1] The MIOVF bits are only affected by peripheral interrupt events. Forcing an interrupt using the MIFRC or IACK operation will not set the overflow flag even if the MIFR bit is set. Reset type: SYSRSn 0h (R/W) = A task 4 interrupt overflow has not occurred (default) 1h (R/W) = A task 4 interrupt overflow has occurred
2	INT3	R	0h	These bits, when set to "1", indicate an interrupt overflow event occurred. Such an event occurs when the IFR bit is already set. An overflow event remains latched and respective bits can only be cleared by writing to the MICLROVF register. If the MIFR bit is being cleared by a new task on the same cycle as a new peripheral interrupt occurs, the overflow flag will not be affected and the respective MIFR bit will be set. If the MIOVF bit is being cleared by the MICLROVF register on the same cycle as the overflow bit is being set by hardware, then the hardware will have priority. Notes: [1] The MIOVF bits are only affected by peripheral interrupt events. Forcing an interrupt using the MIFRC or IACK operation will not set the overflow flag even if the MIFR bit is set. Reset type: SYSRSn 0h (R/W) = A task 3 interrupt overflow has not occurred (default) 1h (R/W) = A task 3 interrupt overflow has occurred

Table 8-52. MIOVF Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	INT2	R	0h	These bits, when set to "1", indicate an interrupt overflow event occurred. Such an event occurs when the IFR bit is already set. An overflow event remains latched and respective bits can only be cleared by writing to the MICLROVF register. If the MIFR bit is being cleared by a new task on the same cycle as a new peripheral interrupt occurs, the overflow flag will not be affected and the respective MIFR bit will be set. If the MIOVF bit is being cleared by the MICLROVF register on the same cycle as the overflow bit is being set by hardware, then the hardware will have priority. Notes: [1] The MIOVF bits are only affected by peripheral interrupt events. Forcing an interrupt using the MIFRC or IACK operation will not set the overflow flag even if the MIFR bit is set. Reset type: SYSRSn 0h (R/W) = A task 2 interrupt overflow has not occurred (default) 1h (R/W) = A task 2 interrupt overflow has occurred
0	INT1	R	0h	These bits, when set to "1", indicate an interrupt overflow event occurred. Such an event occurs when the IFR bit is already set. An overflow event remains latched and respective bits can only be cleared by writing to the MICLROVF register. If the MIFR bit is being cleared by a new task on the same cycle as a new peripheral interrupt occurs, the overflow flag will not be affected and the respective MIFR bit will be set. If the MIOVF bit is being cleared by the MICLROVF register on the same cycle as the overflow bit is being set by hardware, then the hardware will have priority. Notes: [1] The MIOVF bits are only affected by peripheral interrupt events. Forcing an interrupt using the MIFRC or IACK operation will not set the overflow flag even if the MIFR bit is set. Reset type: SYSRSn 0h (R/W) = A task 1 interrupt overflow has not occurred (default) 1h (R/W) = A task 1 interrupt overflow has occurred

8.8.4.17 MIFRC Register (Offset = 22h) [Reset = 0h]

MIFRC is shown in Figure 8-26 and described in Table 8-53.

Return to the Summary Table.

The interrupt force register can be used by the main CPU to start tasks through software. Writing a 1 to a MIFRC bit will set the corresponding bit in the MIFR register. Writes of 0 are ignored and reads always return 0. The IACK #16bit operation can also be used to start tasks and has the same effect as the MIFRC register. To enable IACK to set MIFR bits you must first set the MCTL[IACKE] bit. Using IACK has the advantage of not having to first set the EALLOW bit. This allows the main CPU to efficiently trigger CLA tasks through software.

Figure 8-26. MIFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 8-53. MIFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	INT8	R-0/W1S	0h	Writing a "1" to any of the bits will set the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to force the task 8 interrupt
6	INT7	R-0/W1S	0h	Writing a "1" to any of the bits will set the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to force the task 7 interrupt
5	INT6	R-0/W1S	0h	Writing a "1" to any of the bits will set the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to force the task 6 interrupt
4	INT5	R-0/W1S	0h	Writing a "1" to any of the bits will set the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to force the task 5 interrupt

Table 8-53. MIFRC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	INT4	R-0/W1S	0h	Writing a "1" to any of the bits will set the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to force the task 4 interrupt
2	INT3	R-0/W1S	0h	Writing a "1" to any of the bits will set the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to force the task 3 interrupt
1	INT2	R-0/W1S	0h	Writing a "1" to any of the bits will set the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to force the task 2 interrupt
0	INT1	R-0/W1S	0h	Writing a "1" to any of the bits will set the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to force the task 1 interrupt

8.8.4.18 MICLR Register (Offset = 23h) [Reset = 0h]

MICLR is shown in Figure 8-27 and described in Table 8-54.

Return to the Summary Table.

Normally bits in the MIFR register are automatically cleared when a task begins. The interrupt flag clear register can be used to instead manually clear bits in the interrupt flag (MIFR) register. Writing a 1 to a MICLR bit will clear the corresponding bit in the MIFR register. Writes of 0 are ignored and reads always return 0.

Figure 8-27. MICLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 8-54. MICLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	INT8	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 8 interrupt flag
6	INT7	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 7 interrupt flag
5	INT6	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 6 interrupt flag
4	INT5	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 5 interrupt flag

Table 8-54. MICLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	INT4	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 4 interrupt flag
2	INT3	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 3 interrupt flag
1	INT2	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 2 interrupt flag
0	INT1	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 1 interrupt flag

8.8.4.19 MICLROVF Register (Offset = 24h) [Reset = 0h]

MICLROVF is shown in Figure 8-28 and described in Table 8-55.

Return to the Summary Table.

Overflow flag bits in the MIOVF register are latched until manually cleared using the MICLROVF register. Writing a 1 to a MICLROVF bit will clear the corresponding bit in the MIOVF register. Writes of 0 are ignored and reads always return 0.

Figure 8-28. MICLROVF Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 8-55. MICLROVF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	INT8	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIOVF bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIOVF register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 8 interrupt overflow flag
6	INT7	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIOVF bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIOVF register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 7 interrupt overflow flag
5	INT6	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIOVF bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIOVF register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 6 interrupt overflow flag
4	INT5	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIOVF bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIOVF register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 5 interrupt overflow flag
3	INT4	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIOVF bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIOVF register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 4 interrupt overflow flag

Table 8-55. MICLROVF Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	INT3	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIOVF bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIOVF register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 3 interrupt overflow flag
1	INT2	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIOVF bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIOVF register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 2 interrupt overflow flag
0	INT1	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIOVF bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIOVF register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 1 interrupt overflow flag

8.8.4.20 MIER Register (Offset = 25h) [Reset = 0h]

MIER is shown in Figure 8-29 and described in Table 8-56.

Return to the Summary Table.

Setting the bits in the interrupt enable register (MIER) allow an incoming interrupt or main CPU software to start the corresponding CLA task. Writing a 0 will block the task, but the interrupt request will still be latched in the flag register (MIFLG). Setting the MIER register bit to 0 while the corresponding task is executing will have no effect on the task. The task will continue to run until it hits the MSTOP instruction. When a soft reset is issued, the MIER bits are cleared. There should always be at least a 1 SYSCLKOUT delay between issuing the soft reset and reconfiguring the MIER bits.

Figure 8-29. MIER Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 8-56. MIER Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	INT8	R/W	0h	Setting any of the bits to "1" enables the corresponding interrupt from triggering a corresponding CLA task. Writing a "0" blocks the interrupt, but the interrupt can still be latched by the MIFR register. When an interrupt is enabled and the corresponding MIFR bit is set to "1", the CLA will start executing the corresponding task and automatically clear the corresponding MIFR bit. Interrupts are be serviced in normal priority order. Notes: [1] If a task is currently executing and the corresponding MIER bit is cleared to "0", it will have no effect on the task. The task will run until it hits the STOP instruction. Reset type: SYSRSn 0h (R/W) = TASK_INT_DISABLE Task 8 interrupt is disabled (default) 1h (R/W) = TASK_INT_ENABLE Task 8 interrupt is enabled
6	INT7	R/W	0h	Setting any of the bits to "1" enables the corresponding interrupt from triggering a corresponding CLA task. Writing a "0" blocks the interrupt, but the interrupt can still be latched by the MIFR register. When an interrupt is enabled and the corresponding MIFR bit is set to "1", the CLA will start executing the corresponding task and automatically clear the corresponding MIFR bit. Interrupts are be serviced in normal priority order. Notes: [1] If a task is currently executing and the corresponding MIER bit is cleared to "0", it will have no effect on the task. The task will run until it hits the STOP instruction. Reset type: SYSRSn 0h (R/W) = TASK_INT_DISABLE Task 7 interrupt is disabled (default) 1h (R/W) = TASK_INT_ENABLE Task 7 interrupt is enabled

Table 8-56. MIER Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	INT6	R/W	0h	Setting any of the bits to "1" enables the corresponding interrupt from triggering a corresponding CLA task. Writing a "0" blocks the interrupt, but the interrupt can still be latched by the MIFR register. When an interrupt is enabled and the corresponding MIFR bit is set to "1", the CLA will start executing the corresponding task and automatically clear the corresponding MIFR bit. Interrupts are be serviced in normal priority order. Notes: [1] If a task is currently executing and the corresponding MIER bit is cleared to "0", it will have no effect on the task. The task will run until it hits the STOP instruction. Reset type: SYSRSn 0h (R/W) = TASK_INT_DISABLE Task 6 interrupt is disabled (default) 1h (R/W) = TASK_INT_ENABLE Task 6 interrupt is enabled
4	INT5	R/W	0h	Setting any of the bits to "1" enables the corresponding interrupt from triggering a corresponding CLA task. Writing a "0" blocks the interrupt, but the interrupt can still be latched by the MIFR register. When an interrupt is enabled and the corresponding MIFR bit is set to "1", the CLA will start executing the corresponding task and automatically clear the corresponding MIFR bit. Interrupts are be serviced in normal priority order. Notes: [1] If a task is currently executing and the corresponding MIER bit is cleared to "0", it will have no effect on the task. The task will run until it hits the STOP instruction. Reset type: SYSRSn 0h (R/W) = TASK_INT_DISABLE Task 5 interrupt is disabled (default) 1h (R/W) = TASK_INT_ENABLE Task 5 interrupt is enabled
3	INT4	R/W	0h	Setting any of the bits to "1" enables the corresponding interrupt from triggering a corresponding CLA task. Writing a "0" blocks the interrupt, but the interrupt can still be latched by the MIFR register. When an interrupt is enabled and the corresponding MIFR bit is set to "1", the CLA will start executing the corresponding task and automatically clear the corresponding MIFR bit. Interrupts are be serviced in normal priority order. Notes: [1] If a task is currently executing and the corresponding MIER bit is cleared to "0", it will have no effect on the task. The task will run until it hits the STOP instruction. Reset type: SYSRSn 0h (R/W) = TASK_INT_DISABLE Task 4 interrupt is disabled (default) 1h (R/W) = TASK_INT_ENABLE Task 4 interrupt is enabled
2	INT3	R/W	0h	Setting any of the bits to "1" enables the corresponding interrupt from triggering a corresponding CLA task. Writing a "0" blocks the interrupt, but the interrupt can still be latched by the MIFR register. When an interrupt is enabled and the corresponding MIFR bit is set to "1", the CLA will start executing the corresponding task and automatically clear the corresponding MIFR bit. Interrupts are be serviced in normal priority order. Notes: [1] If a task is currently executing and the corresponding MIER bit is cleared to "0", it will have no effect on the task. The task will run until it hits the STOP instruction. Reset type: SYSRSn 0h (R/W) = TASK_INT_DISABLE Task 3 interrupt is disabled (default) 1h (R/W) = TASK_INT_ENABLE Task 3 interrupt is enabled

Table 8-56. MIER Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	INT2	R/W	0h	Setting any of the bits to "1" enables the corresponding interrupt from triggering a corresponding CLA task. Writing a "0" blocks the interrupt, but the interrupt can still be latched by the MIFR register. When an interrupt is enabled and the corresponding MIFR bit is set to "1", the CLA will start executing the corresponding task and automatically clear the corresponding MIFR bit. Interrupts are be serviced in normal priority order. Notes: [1] If a task is currently executing and the corresponding MIER bit is cleared to "0", it will have no effect on the task. The task will run until it hits the STOP instruction. Reset type: SYSRSn 0h (R/W) = TASK_INT_DISABLE Task 2 interrupt is disabled (default) 1h (R/W) = TASK_INT_ENABLE Task 2 interrupt is enabled
0	INT1	R/W	0h	Setting any of the bits to "1" enables the corresponding interrupt from triggering a corresponding CLA task. Writing a "0" blocks the interrupt, but the interrupt can still be latched by the MIFR register. When an interrupt is enabled and the corresponding MIFR bit is set to "1", the CLA will start executing the corresponding task and automatically clear the corresponding MIFR bit. Interrupts are be serviced in normal priority order. Notes: [1] If a task is currently executing and the corresponding MIER bit is cleared to "0", it will have no effect on the task. The task will run until it hits the STOP instruction. Reset type: SYSRSn 0h (R/W) = TASK_INT_DISABLE Task 1 interrupt is disabled (default) 1h (R/W) = TASK_INT_ENABLE Task 1 interrupt is enabled

8.8.4.21 MIRUN Register (Offset = 26h) [Reset = 0h]

MIRUN is shown in Figure 8-30 and described in Table 8-57.

Return to the Summary Table.

The interrupt run status register (MIRUN) indicates which task is currently executing. Only one MIRUN bit will ever be set to a 1 at any given time. The bit is automatically cleared when the task competes and the respective interrupt is fed to the peripheral interrupt expansion (PIE) block of the device. This lets the main CPU know when a task has completed. The main CPU can stop a currently running task by writing to the MCTL[SOFTRESET] bit. This will clear the MIRUN flag and stop the task. In this case no interrupt will be generated to the PIE.

Figure 8-30. MIRUN Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 8-57. MIRUN Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	INT8	R	0h	These bits indicate which task is currently active. Only one bit can be set to "1" at any one time. The bit is automatically cleared to "0" when the task completes and the respective CLAINTxn line is toggled to indicate task completion. The CLAINTxn interrupt line can be fed to the PIE of the CPU so the CPU knows when a task has completed. A currently running task can be stopped by a SOFTRESET. The RUN flag is cleared, the task is stopped, but no CLAINTxn interrupt is generated. Reset type: SYSRSn 0h (R/W) = Task 8 is not executing (default) 1h (R/W) = Task 8 is executing
6	INT7	R	0h	These bits indicate which task is currently active. Only one bit can be set to "1" at any one time. The bit is automatically cleared to "0" when the task completes and the respective CLAINTxn line is toggled to indicate task completion. The CLAINTxn interrupt line can be fed to the PIE of the CPU so the CPU knows when a task has completed. A currently running task can be stopped by a SOFTRESET. The RUN flag is cleared, the task is stopped, but no CLAINTxn interrupt is generated. Reset type: SYSRSn 0h (R/W) = Task 7 is not executing (default) 1h (R/W) = Task 7 is executing
5	INT6	R	0h	These bits indicate which task is currently active. Only one bit can be set to "1" at any one time. The bit is automatically cleared to "0" when the task completes and the respective CLAINTxn line is toggled to indicate task completion. The CLAINTxn interrupt line can be fed to the PIE of the CPU so the CPU knows when a task has completed. A currently running task can be stopped by a SOFTRESET. The RUN flag is cleared, the task is stopped, but no CLAINTxn interrupt is generated. Reset type: SYSRSn 0h (R/W) = Task 6 is not executing (default) 1h (R/W) = Task 6 is executing

Table 8-57. MIRUN Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	INT5	R	0h	These bits indicate which task is currently active. Only one bit can be set to "1" at any one time. The bit is automatically cleared to "0" when the task completes and the respective CLAINTxn line is toggled to indicate task completion. The CLAINTxn interrupt line can be fed to the PIE of the CPU so the CPU knows when a task has completed. A currently running task can be stopped by a SOFTRESET. The RUN flag is cleared, the task is stopped, but no CLAINTxn interrupt is generated. Reset type: SYSRSn 0h (R/W) = Task 5 is not executing (default) 1h (R/W) = Task 5 is executing
3	INT4	R	0h	These bits indicate which task is currently active. Only one bit can be set to "1" at any one time. The bit is automatically cleared to "0" when the task completes and the respective CLAINTxn line is toggled to indicate task completion. The CLAINTxn interrupt line can be fed to the PIE of the CPU so the CPU knows when a task has completed. A currently running task can be stopped by a SOFTRESET. The RUN flag is cleared, the task is stopped, but no CLAINTxn interrupt is generated. Reset type: SYSRSn 0h (R/W) = Task 4 is not executing (default) 1h (R/W) = Task 4 is executing
2	INT3	R	0h	These bits indicate which task is currently active. Only one bit can be set to "1" at any one time. The bit is automatically cleared to "0" when the task completes and the respective CLAINTxn line is toggled to indicate task completion. The CLAINTxn interrupt line can be fed to the PIE of the CPU so the CPU knows when a task has completed. A currently running task can be stopped by a SOFTRESET. The RUN flag is cleared, the task is stopped, but no CLAINTxn interrupt is generated. Reset type: SYSRSn 0h (R/W) = Task 3 is not executing (default) 1h (R/W) = Task 3 is executing
1	INT2	R	0h	These bits indicate which task is currently active. Only one bit can be set to "1" at any one time. The bit is automatically cleared to "0" when the task completes and the respective CLAINTxn line is toggled to indicate task completion. The CLAINTxn interrupt line can be fed to the PIE of the CPU so the CPU knows when a task has completed. A currently running task can be stopped by a SOFTRESET. The RUN flag is cleared, the task is stopped, but no CLAINTxn interrupt is generated. Reset type: SYSRSn 0h (R/W) = Task 2 is not executing (default) 1h (R/W) = Task 2 is executing
0	INT1	R	0h	These bits indicate which task is currently active. Only one bit can be set to "1" at any one time. The bit is automatically cleared to "0" when the task completes and the respective CLAINTxn line is toggled to indicate task completion. The CLAINTxn interrupt line can be fed to the PIE of the CPU so the CPU knows when a task has completed. A currently running task can be stopped by a SOFTRESET. The RUN flag is cleared, the task is stopped, but no CLAINTxn interrupt is generated. Reset type: SYSRSn 0h (R/W) = Task 1 is not executing (default) 1h (R/W) = Task 1 is executing

8.8.4.22 _MPC Register (Offset = 28h) [Reset = 0h]

_MPC is shown in Figure 8-31 and described in Table 8-58.

Return to the Summary Table.

CLA Program Counter

Figure 8-31. _MPC Register

15	14	13	12	11	10	9	8
_MPC							
R-0h							
7	6	5	4	3	2	1	0
_MPC							
R-0h							

Table 8-58. _MPC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	_MPC	R	0h	Program Counter: The PC value is initialized by the appropriate MVECTx register when an interrupt (task) is serviced. The MPC register address 16-bits and not 32-bits. Hence the address range of the CLA with a 16-bit MPC is 64Kx16 words or 32K CLA instructions. Notes: [1] To be consistent with C28 core implementation, the PC value points to the instruction in D2 stage of pipeline. [2] After a STOP operation, and with no other task pending, the PC will remain pointing to the STOP operation. Reset type: SYSRSn

8.8.4.23 _MAR0 Register (Offset = 2Ah) [Reset = 0h]

_MAR0 is shown in Figure 8-32 and described in Table 8-59.

Return to the Summary Table.

CLA Auxiliary Register 0

Figure 8-32. _MAR0 Register

15	14	13	12	11	10	9	8
_MAR0							
R-0h							
7	6	5	4	3	2	1	0
_MAR0							
R-0h							

Table 8-59. _MAR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	_MAR0	R	0h	CLA Auxillary Register 0 Reset type: SYSRSn

8.8.4.24 _MAR1 Register (Offset = 2Bh) [Reset = 0h]

_MAR1 is shown in Figure 8-33 and described in Table 8-60.

Return to the Summary Table.

CLA Auxiliary Register 1

Figure 8-33. _MAR1 Register

15	14	13	12	11	10	9	8
_MAR1							
R-0h							
7	6	5	4	3	2	1	0
_MAR1							
R-0h							

Table 8-60. _MAR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	_MAR1	R	0h	CLA Auxillary Register 1 Reset type: SYSRSn

8.8.4.25 _MSTF Register (Offset = 2Eh) [Reset = 0h]

_MSTF is shown in Figure 8-34 and described in Table 8-61.

Return to the Summary Table.

The CLA status register (MSTF) reflects the results of different operations. These are the basic rules for the flags:
- Zero and negative flags are cleared or set based on:
- floating-point moves to registers
- the result of compare, minimum, maximum, negative and absolute value operations
- the integer result of operations such as MMOV16, MAND32, MOR32, MXOR32, MCMP32, MASR32, MLSR32
- Overflow and underflow flags are set by floating-point math instructions such as multiply, add, subtract and 1/x. These flags may also be connected to the peripheral interrupt expansion (PIE) block on your device. This can be useful for debugging underflow and overflow conditions within an application.

Figure 8-34. _MSTF Register

31	30	29	28	27	26	25	24
RESERVED				_RPC			
R-0h				R-0h			
23	22	21	20	19	18	17	16
_RPC							
R-0h							
15	14	13	12	11	10	9	8
_RPC				MEALLOW	RESERVED	RNDF32	RESERVED
R-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	TF	RESERVED		ZF	NF	LUF	LVF
R-0h	R-0h	R-0h		R-0h	R-0h	R-0h	R-0h

Table 8-61. _MSTF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-12	_RPC	R	0h	Return program counter The _RPC is used to save and restore the MPC address by the MCCNDD and MRCNDD operations Reset type: SYSRSn
11	MEALLOW	R	0h	MEALLOW Status This bit enables and disables CLA write access to EALLOW protected registers This is independent of the state of the EALLOW bit in the main CPU status register This status bit can be saved and restored by the MMOV32 STF, mem32 instruction Reset type: SYSRSn 0h (R/W) = The CLA cannot write to EALLOW protected registers. This bit is cleared by the CLA instruction, MEDIS. 1h (R/W) = The CLA is allowed to write to EALLOW protected registers. This bit is set by the CLA instruction, MEALLOW.
10	RESERVED	R	0h	Reserved
9	RNDF32	R	0h	Round 32-bit Floating-Point Mode Use the MSETFLG and MMOV32 MSTF, mem32 instructions to change the rounding mode Reset type: SYSRSn 0h (R/W) = If this bit is zero, the MMPYF32, MADDF32 and MSUBF32 instructions will round to zero (truncate). 1h (R/W) = If this bit is one, the MMPYF32, MADDF32 and MSUBF32 instructions will round to the nearest even value.

Table 8-61. _MSTF Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8-7	RESERVED	R	0h	Reserved
6	TF	R	0h	Test Flag The MTESTTF instruction can modify this flag based on the condition tested The MSETFLG and MMOV32 MSTF, mem32 instructions can also be used to modify this flag Reset type: SYSRSn 0h (R/W) = The condition tested with the MTESTTF instruction is false. 1h (R/W) = The condition tested with the MTESTTF instruction is true.
5-4	RESERVED	R	0h	Reserved
3	ZF	R	0h	Zero Flag - Instructions that modify this flag based on the floating-point value stored in the destination register: MMOV32, MMOVD32, MABSF32, MNEGF32 - Instructions that modify this flag based on the floating-point result of the operation: MCMPF32, MMAXF32, and MMINF32 - Instructions that modify this flag based on the integer result of the operation: MMOV16, MAND32, MOR32, MXOR32, MCMP32, MASR32, MLSR32 and MLSL32 The MSETFLG and MMOV32 MSTF, mem32 instructions can also be used to modify this flag Reset type: SYSRSn 0h (R/W) = The value is not zero 1h (R/W) = The value is zero
2	NF	R	0h	Negative Flag - Instructions that modify this flag based on the floating-point value stored in the destination register: MMOV32, MMOVD32, MABSF32, MNEGF32 - Instructions that modify this flag based on the floating-point result of the operation: MCMPF32, MMAXF32, and MMINF32 - Instructions that modify this flag based on the integer result of the operation: MMOV16, MAND32, MOR32, MXOR32, MCMP32, MASR32, MLSR32 and MLSL32 The MSETFLG and MMOV32 MSTF, mem32 instructions can also be used to modify this flag Reset type: SYSRSn 0h (R/W) = The value is not negative 1h (R/W) = The value is negative
1	LUF	R	0h	Latched Underflow Flag The following instructions will set this flag to 1 if an underflow occurs: MMPYF32, MADDF32, MSUBF32, MMACF32, MEINVF32, MEISQRTF32 The MSETFLG and MMOV32 MSTF, mem32 instructions can also be used to modify this flag Reset type: SYSRSn 0h (R/W) = An underflow condition has not been latched 1h (R/W) = An underflow condition has been latched

Table 8-61. _MSTF Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	LVF	R	0h	Latched Overflow Flag The following instructions will set this flag to 1 if an overflow occurs: MMPYF32, MADDF32, MSUBF32, MMACF32, MEINVF32, MEISQRTF32 The MSETFLG and MMOV32 MSTF, mem32 instructions can also be used to modify this flag Reset type: SYSRSn 0h (R/W) = An overflow condition has not been latched 1h (R/W) = An overflow condition has been latched

8.8.4.26 _MR0 Register (Offset = 30h) [Reset = 0h]

_MR0 is shown in Figure 8-35 and described in Table 8-62.

Return to the Summary Table.

CLA Floating-Point Result Register 0

Figure 8-35. _MR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
i32																															
R-0h																															

Table 8-62. _MR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	i32	R	0h	CLA Result Register 0 Reset type: SYSRSn

8.8.4.27 _MR1 Register (Offset = 34h) [Reset = 0h]

_MR1 is shown in Figure 8-36 and described in Table 8-63.

Return to the Summary Table.

CLA Floating-Point Result Register 1

Figure 8-36. _MR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
i32																															
R-0h																															

Table 8-63. _MR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	i32	R	0h	CLA Result Register 1 Reset type: SYSRSn

8.8.4.28 _MR2 Register (Offset = 38h) [Reset = 0h]

_MR2 is shown in Figure 8-37 and described in Table 8-64.

Return to the Summary Table.

CLA Floating-Point Result Register 2

Figure 8-37. _MR2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
i32																															
R-0h																															

Table 8-64. _MR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	i32	R	0h	CLA Result Register 2 Reset type: SYSRSn

8.8.4.29 _MR3 Register (Offset = 3Ch) [Reset = 0h]

_MR3 is shown in Figure 8-38 and described in Table 8-65.

Return to the Summary Table.

CLA Floating-Point Result Register 3

Figure 8-38. _MR3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
i32																															
R-0h																															

Table 8-65. _MR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	i32	R	0h	CLA Result Register 3 Reset type: SYSRSn

8.8.4.30 _MPSACTL Register (Offset = 42h) [Reset = 0h]

_MPSACTL is shown in Figure 8-39 and described in Table 8-66.

Return to the Summary Table.

PSA Control Register

Figure 8-39. _MPSACTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
MPSA2CFG		MPSA2CLEAR	MPSA1CLEAR	MDWDBCYC	MDWDBSTART	MPABCYC	MPABSTART
R/W-0h		R-0/W1S-0h	R-0/W1S-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 8-66. _MPSACTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-6	MPSA2CFG	R/W	0h	CLA PSA2 Polynomial Configuration Bits: These bits configure the type of polynomial used for PSA2. The polynomials chosen are commonly used in the industry: Mode Polynomial Type 0,0 PSA 0,1 CRC32 1,0 CRC16 1,1 CRC16-CCITT Note: [1] Polynomial configuration should be performed when PSA2 is stopped. Reset type: SYSRSn
5	MPSA2CLEAR	R-0/W1S	0h	CLA PSA2 Clear Bit: Writing of "1" will clear contents of PSA2 register. Writes of "0" are ignored. Always reads back a "0" Note: Clearing operation should be performed when PSA2 is stopped. Reset type: SYSRSn
4	MPSA1CLEAR	R-0/W1S	0h	CLA PSA1 Clear Bit: Writing of "1" will clear contents of PSA1 register. Writes of "0" are ignored. Always reads back a "0" Note: Clearing operation should be performed when PSA1 is stopped. Reset type: SYSRSn
3	MDWDBCYC	R/W	0h	CLA Data Write Data Bus PSA2 Cycle or Event Based Bit: 0 PSA2 calculated on every cycle 1 PSA2 calculated on every bus event Reset type: SYSRSn
2	MDWDBSTART	R/W	0h	CLA Data Write Data Bus PSA2 Start/Stop Bit: 0 PSA2 stopped 1 PSA2 start Reset type: SYSRSn
1	MPABCYC	R/W	0h	CLA Program Address Bus PSA1 Cycle/Event Based Bit: 0 PSA1 calculated on every cycle 1 PSA1 calculated on every bus event Reset type: SYSRSn
0	MPABSTART	R/W	0h	CLA Program Address Bus PSA1 Start/Stop Bit: 0 PSA1 stopped 1 PSA1 start Reset type: SYSRSn

8.8.4.31 _MPSA1 Register (Offset = 44h) [Reset = 0h]

_MPSA1 is shown in Figure 8-40 and described in Table 8-67.

Return to the Summary Table.

PSA1 Register

Figure 8-40. _MPSA1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
i32																															
R/W-0h																															

Table 8-67. _MPSA1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	i32	R/W	0h	PSA1 Value: Reading this register gives the current PSA1 value. The value can be read at any time. Writes to this register are allowed to initialize the PSA1 to a known value. Writes to this register should only be made when PSA1 is stopped. Register value is cleared to zero by reset or by writing to the MPSA1CLEAR bit in the MPSACTL register. Reset type: SYSRSn

8.8.4.32 _MPSA2 Register (Offset = 46h) [Reset = 0h]

_MPSA2 is shown in Figure 8-41 and described in Table 8-68.

Return to the Summary Table.

PSA2 Register

Figure 8-41. _MPSA2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
i32																															
R/W-0h																															

Table 8-68. _MPSA2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	i32	R/W	0h	PSA2 Value: Reading this register gives the current PSA2 value. The value can be read at any time. Writes to this register are allowed to initialize the PSA2 to a known value. Writes to this register should only be made when PSA2 is stopped. Register value is cleared to zero by reset or by writing to the MPSA2CLEAR bit in the MPSACTL register. Reset type: SYSRSn

8.8.5 CLA Registers to Driverlib Functions

Table 8-69. CLA Registers to Driverlib Functions

File	Driverlib Function
MVECT1	
cla.h	CLA_mapTaskVector
MVECT2	
-	See MVECT1
MVECT3	
-	See MVECT1
MVECT4	
-	See MVECT1
MVECT5	
-	See MVECT1
MVECT6	
-	See MVECT1
MVECT7	
-	See MVECT1
MVECT8	
-	See MVECT1
MCTL	
cla.h	CLA_performHardReset
cla.h	CLA_performSoftReset
cla.h	CLA_enableIACK
cla.h	CLA_disableIACK
cla.h	CLA_enableBackgroundTask
cla.h	CLA_disableBackgroundTask
cla.h	CLA_startBackgroundTask
cla.h	CLA_enableHardwareTrigger

Table 8-69. CLA Registers to Driverlib Functions (continued)

File	Driverlib Function
cla.h	CLA_disableHardwareTrigger
MVECTBGRNDACTIVE	
cla.h	CLA_getBackgroundActiveVector
SOFTINTEN	
cla.h	CLA_enableSoftwareInterrupt
cla.h	CLA_disableSoftwareInterrupt
MSTSBGRND	
cla.h	CLA_getBackgroundTaskStatus
MCTLBGRND	
cla.h	CLA_enableBackgroundTask
cla.h	CLA_disableBackgroundTask
cla.h	CLA_startBackgroundTask
cla.h	CLA_enableHardwareTrigger
cla.h	CLA_disableHardwareTrigger
MVECTBGRND	
cla.h	CLA_getBackgroundActiveVector
cla.h	CLA_mapBackgroundTaskVector
MIFR	
cla.h	CLA_getPendingTaskFlag
cla.h	CLA_getAllPendingTaskFlags
cla.h	CLA_forceTasks
MIOVF	
cla.h	CLA_getTaskOverflowFlag
cla.h	CLA_getAllTaskOverflowFlags
MIFRC	
cla.h	CLA_forceTasks
MICLR	
cla.h	CLA_clearTaskFlags
MICLROVF	
-	
MIER	
cla.h	CLA_enableTasks
cla.h	CLA_disableTasks
MIRUN	
cla.h	CLA_getTaskRunStatus
cla.h	CLA_getAllTaskRunStatus
MPC	
-	
MAR0	
-	
MAR1	
-	
MSTF	
-	
MR0	

Table 8-69. CLA Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
MR1	
-	
MR2	
-	
MR3	
-	
MPSACTL	
-	
MPSA1	
-	
MPSA2	
-	
MVECTBGRNDACTIVE	
cla.h	CLA_getBackgroundActiveVector
MPSACTL	
-	
MPSA1	
-	
MPSA2	
-	
SOFTINTEN	
cla.h	CLA_enableSoftwareInterrupt
cla.h	CLA_disableSoftwareInterrupt
SOFTINTFRC	
cla.h	CLA_forceSoftwareInterrupt

Chapter 9

Configurable Logic Block (CLB)

This chapter describes the features and operation of the configurable logic block (CLB) that is a collection of configurable blocks that can be inter-connected using software to implement custom digital logic functions.

9.1 Introduction

The configurable logic block (CLB) is a collection of configurable blocks that can be inter-connected using software to implement custom digital logic functions. The CLB is able to enhance existing peripherals through a set of crossbar interconnections, which provide a high level of connectivity to existing control peripherals such as enhanced pulse width modulators (ePWM), enhanced capture modules (eCAP), and enhanced quadrature encoder pulse modules (eQEP). The crossbars also allow the CLB to be connected to external GPIO pins. In this way, the CLB can be configured to interact with device peripherals to perform small logical functions such as simple PWM generators, or to implement custom serial data exchange protocols.

The CLB peripheral is configured through the CLB tool. For more information on the CLB tool, available examples, application reports, and user's guide, refer to the following location in your C2000WARE package (C2000Ware_2_00_00_03 and higher): C2000WARE_INSTALL_LOCATION\utilities\clb_tool\clb_syscfg\doc

9.1.1 CLB Related Collateral

Foundational Materials

- C2000 Academy - CLB
- C2000™ Configurable Logic Block (CLB) Series (Video)
- Customizing on-chip peripherals defies conventional logic
- *Enable Differentiation and win with CLB in various applications Application Report*
- Enable Differentiation with Configurable Logic in Various Automotive Applications (Video)

Getting Started Materials

- *C2000™ Position Manager PTO API Reference Guide Application Report*
- *CLB Tool User Guide*
  - Basic examples are 7 - 15 (start with these). More involved examples are 1-6.
- *Designing With The C2000 Configurable Logic Block Application Report*
- How do I add SYSCONFIG support (Pinmux and Peripheral Initialization) to an existing driverlib project?
- *How to Migrate Custom Logic From an FPGA/CPLD to C2000 Microcontrollers Application Report*
  - Chapters 1-3 are very useful for getting started and learning the CLB. Later chapters are very useful Expert materials for migrating from FPGA/CPLD to C2000's CLB.

Expert Materials

- *Achieve Delayed Protection for Three-Level Inverter With CLB Application Report*
- *Diagnosing Delta-Sigma Modulator Bitstream Using C2000™ Configurable Logic Block Application Report*
- *How to Implement Custom Serial Interfaces Using Configurable Logic Block (CLB) Application Report*
- *Tamagawa T-Format Absolute-Encoder Master Interface Reference Design for C2000™ MCUs*

9.2 Description

The CLB subsystem contains a number of identical tiles. There are four such tiles in the CLB subsystem; other devices can contain more or fewer tiles. Tiles are numbered 1 to N, where N is the total tile count on the device. Each tile contains combinational and sequential logic blocks, as well as other dedicated hardware to be described later in this document. Figure 9-1 shows the structure of the CLB subsystem in the device.

The tile contains the core logic, providing the logic reconfiguration capability. Each CLB tile is associated with a separate CPU interface, which contains the registers needed to control and configure the logic in the tile. The CPU interface also contains data transfer buffers that can be used as part of the configurable logic to exchange data with the rest of the device. Figure 9-2 shows the connections between the tile, the CPU interface, and the device.

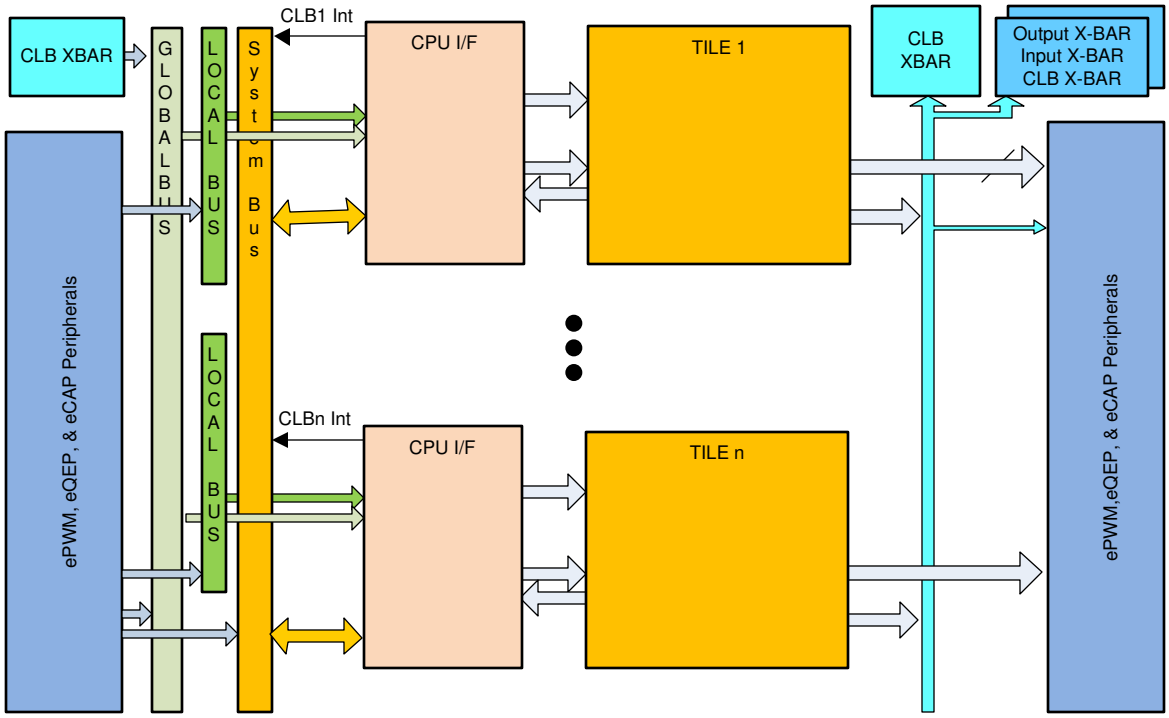


Figure 9-1. Block Diagram of the CLB Subsystem in the Device

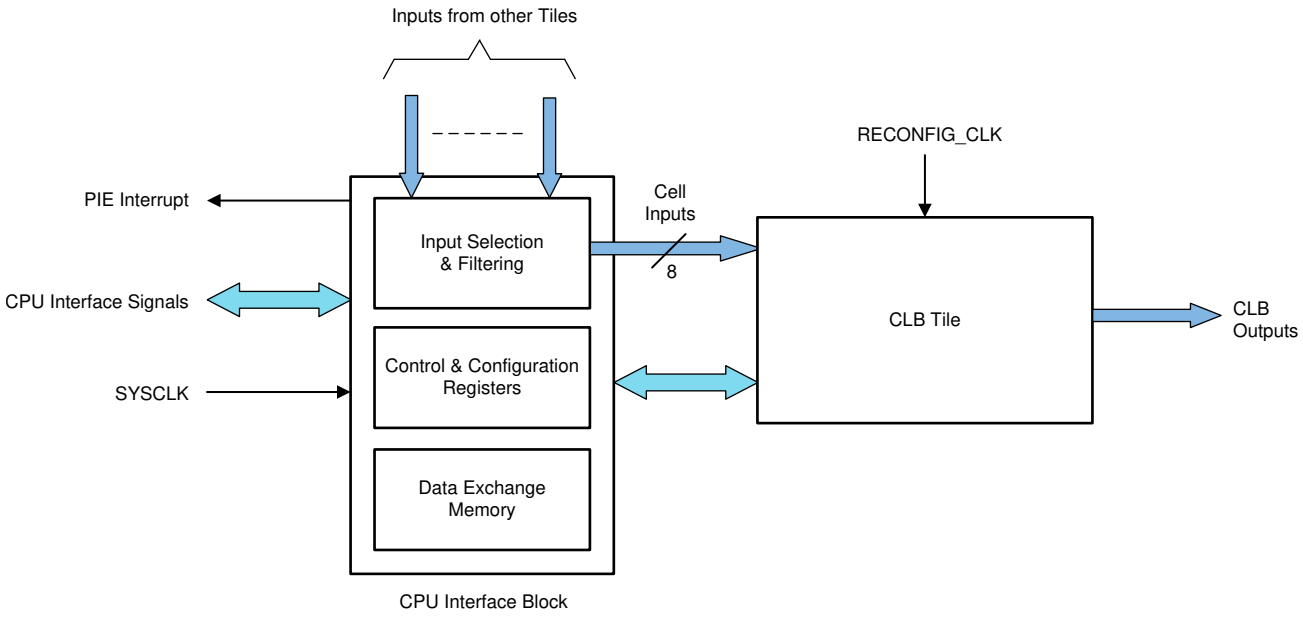


Figure 9-2. Block Diagram of a CLB Tile and CPU Interface

9.2.1 CLB Clock

In this device, the CLB clock is called CLBx clock that can be enabled or disabled by SYSCTL_PERIPH_CLK_CLBx through the SysCtl_enablePeripheral function. The maximum frequency is 150 MHz and the clock can be enabled and configured by modifying the CLBx clock.

Note

When clock frequencies are above 100 MHz (ex: 150 MHz), PIPELINE mode must be enabled.

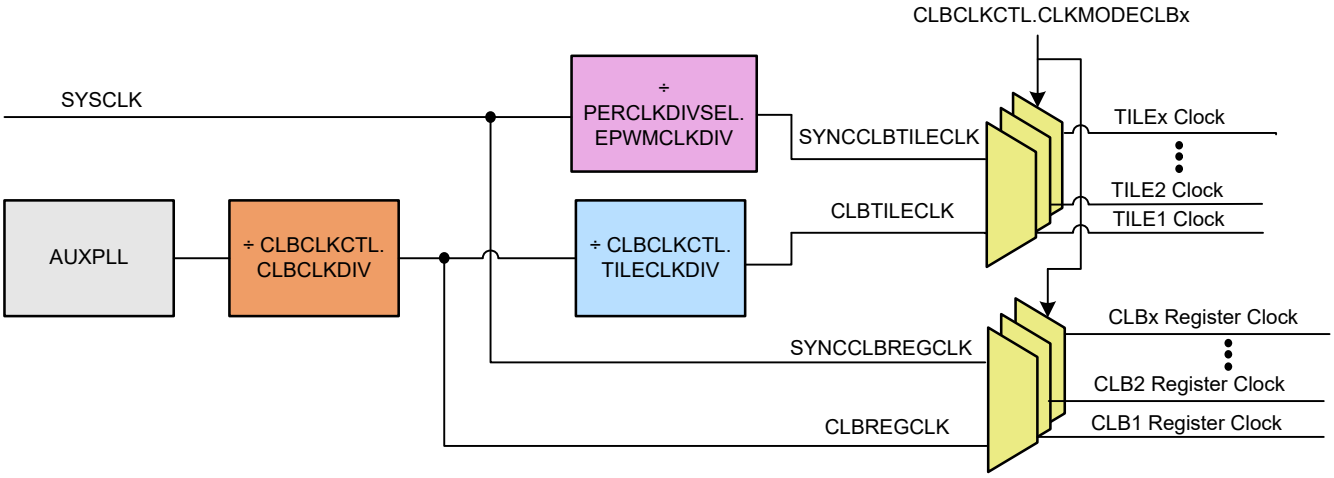


Figure 9-3. CLB Clocking

The CLB TILE clock and CLB register clock can be in ASYNC/SYNC mode with the SYSCLK. An example CLB clock configuration is shown in Table 9-1. Check the device data sheet for details on clocking specifications.

Table 9-1. Example CLB Clocking Configuration

Clock	SYNC Mode (CLKMODECLBx = 0)	ASYNC Mode (CLKMODECLBx = 1)	
		TILECLKDIV = 1	TILECLKDIV = 0
CLB Register Clock	SYSCLK	SYSCLK	SYSCLK
CLB TILE Clock	SYSCLK	SYSCLK / 2	SYSCLK

Starting with CLB Type 2, a clock prescalar module is available. The prescalar module can generate a prescaled version of the CLB clock signal that can be used as an input to the CLB TILEs.

Note

The prescaler logic does not change the actual clocking speed of the CLB. The prescalar generates a strobe (that can toggle at the defined prescaled rate) that is made available as another input signal to the CLB logic and the strobe is only used when required.

The prescalar module is shown in Figure 9-4.

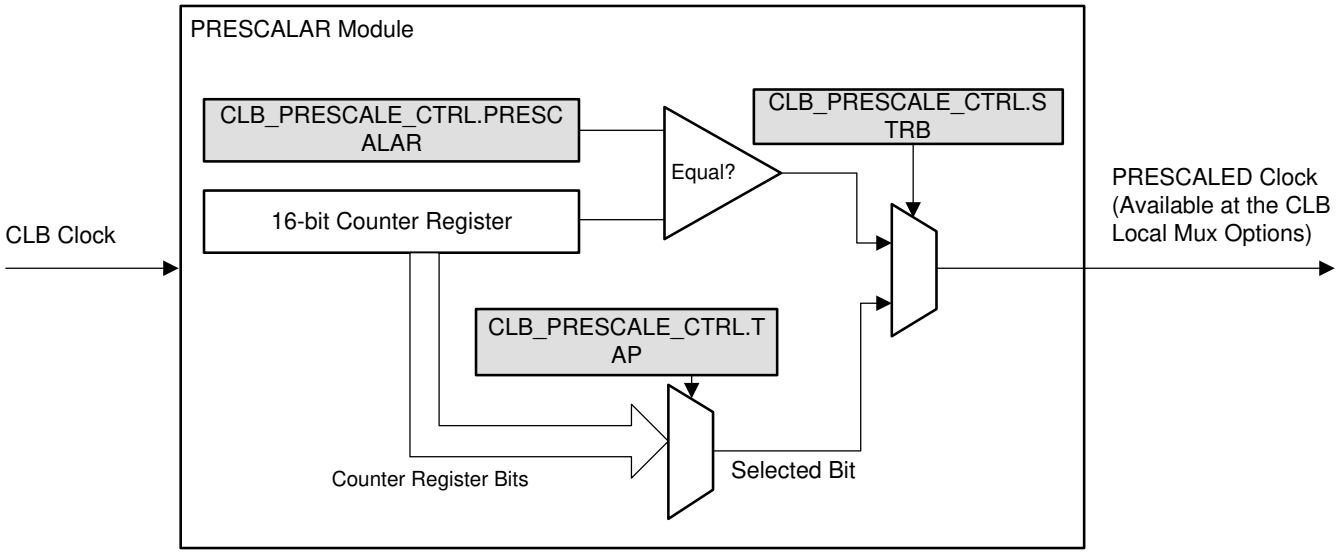


Figure 9-4. CLB Clock Prescalar

9.3 CLB Input/Output Connection

9.3.1 Overview

There are four instances of the CLB module in the device. Each CLB instance has a common set of input signals referred to as global input signals. Additionally, each CLB instance has a specific set of input signals that are unique to each instance, and are referred to as local input signals. Each of the eight inputs of a CLB can be chosen from any of the global input signals or the local input signals.

Note

Signals routed into the CLB using the XBAR must be synchronized within the CLB itself.

9.3.2 CLB Input Selection

Each CLB module has eight inputs that are applied to the reconfigurable logic cell. Each of these inputs can be selectively driven by a predefined set of signals. A two-level mux structure allows each input of each CLB instance to select a signal.

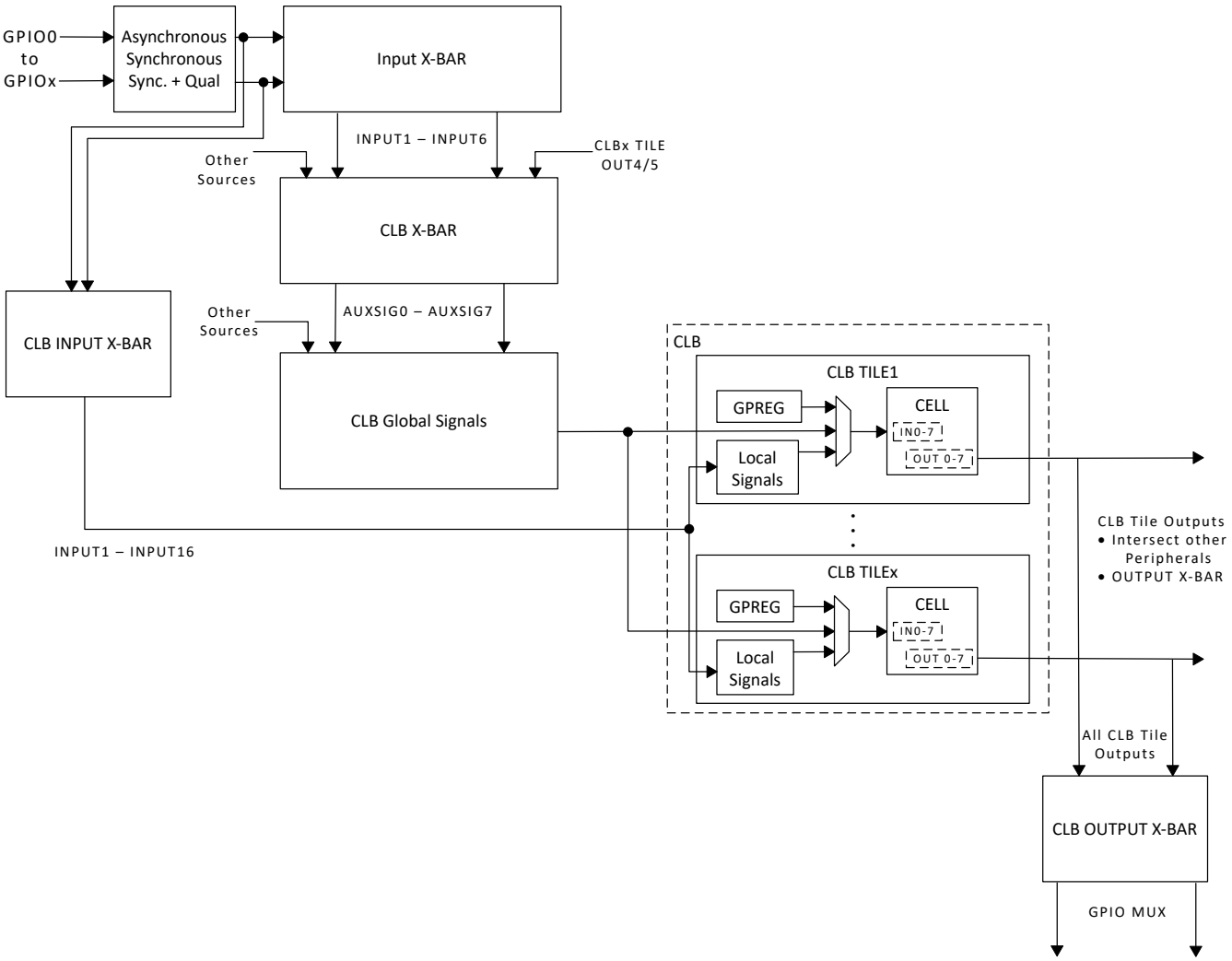


Figure 9-5. GPIO to CLB Tile Connections

A set of signals is common to all the CLB instances. These are referred to as global inputs in Figure 9-6. A separate set of signals is unique to each instance of the CLB. These are referred to as local inputs in Figure 9-6.

Registers CLB_LCL_MUX_SEL_1 and CLB_LCL_MUX_SEL_2 control the local mux selection for each of the eight inputs. The mux control registers CLB_GLBL_MUX_SEL_1 and CLB_GLBL_MUX_SEL_2 control the global mux selection for each of the eight inputs.

The local mux select value of 0 causes the selected global mux input signal to be connected to the corresponding CLB Input. For example, setting CLB_LCL_MUX_SEL_IN_0 = 0 and CLB_GLBL_MUX_SEL_IN_0 = 8 causes the global mux input number 8 to be connected to CLB Input 0. The input filter feature can be used to enable edge detection on the CLB inputs. The input filter feature can also synchronize the input with the CLB clock.

The global mux settings are shown in Table 9-2 and Table 9-3. The local input mux settings are shown in Table 9-4 and Table 9-5.

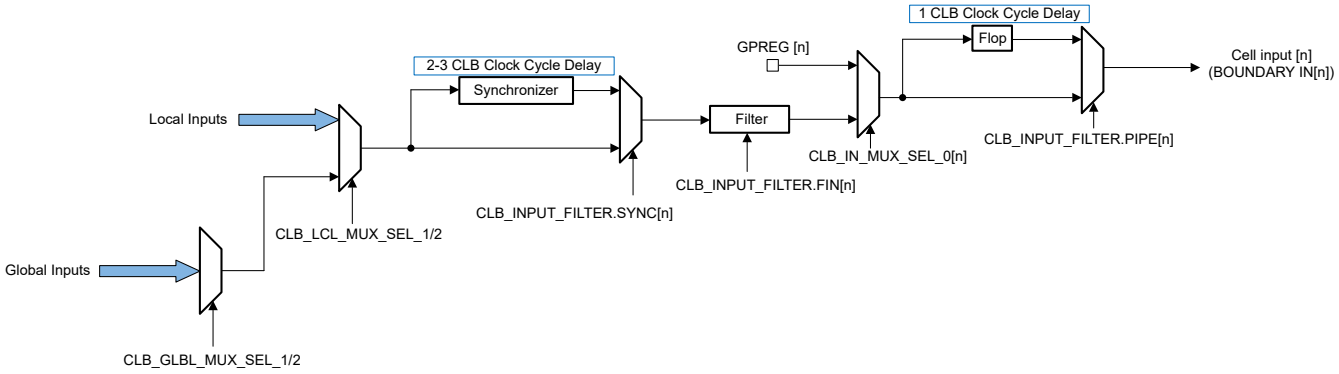


Figure 9-6. CLB Input Mux and Filter

Figure 9-7 shows an example of how to use synchronization for an asynchronous signal, in this case the ePWM signal. Figure 9-8 shows an instance of using input pipelining for a synchronous signal, which here is the ePWM TBCLK signal. Note that these two input configurations are not used simultaneously, and each have their own cycle delay that add to the input path.

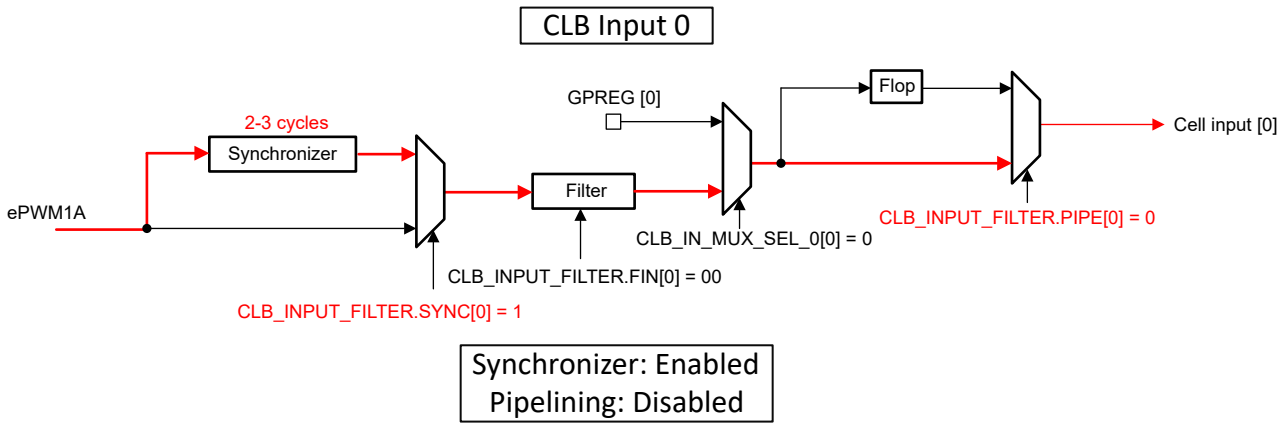


Figure 9-7. CLB Input Synchronization Example

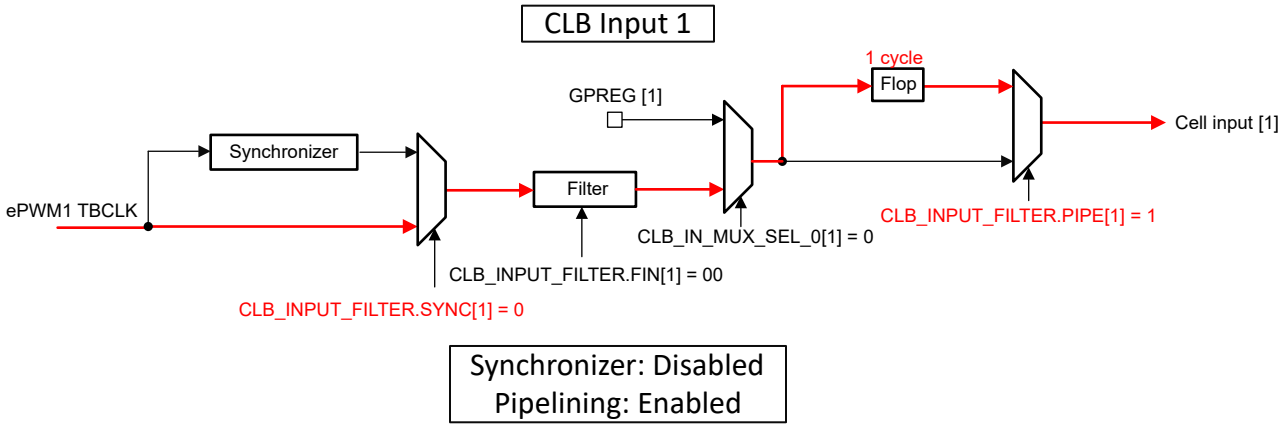


Figure 9-8. CLB Input Pipelining Example

Note

If a signal in the following table indicates that synchronization is required, then the CLB input synchronizer must be enabled using the appropriate SYNC bit in the CLB_INPUT_FILTER register. This synchronization adds a 2-3 CLB clock cycle delay to the input. This delay is either 2 or 3 cycles and is not predictable. There is a potential for a metastability hazard, if the indicated signals are not first synchronized before going into the CLB tile. This metastability can cause errors dependent on voltage, temperature, and wafer fab process. Note that this requirement is in addition to and separate from GPIO input synchronization.

If a signal in the following table indicates that synchronization is not required, as the signal is already synchronous, then pipelining is required and must be enabled using the PIPE bit in the CLB_INPUT_FILTER register. This pipelining adds a 1 CLB clock cycle delay to the input. This is not to be mistaken with the PIPELINE_EN bit in the CLB_LOAD_EN register, which controls pipelining of the CLB operations in the HLC and counter blocks. This PIPELINE_EN bit is also used when the device is run above 100 MHz. Having synchronization and pipelining both enabled or both disabled is not recommended. Enabling both synchronization and pipelining introduces a delay of more than 2-3 CLB clock cycles on the signal path. Disabling both allows the completely asynchronous signal to be routed as an input.

Table 9-2. Global Signals and Mux Selection

Select Value	CLB1 Input	CLB2 Input	CLB3 Input	CLB4 Input	Synchronization Requirement
0	EPWM1A	EPWM1A	EPWM1A	EPWM1A	Enable
1	EPWM1A_OE	EPWM1A_OE	EPWM1A_OE	EPWM1A_OE	Enable
2	EPWM1B	EPWM1B	EPWM1B	EPWM1B	Enable
3	EPWM1B_OE	EPWM1B_OE	EPWM1B_OE	EPWM1B_OE	Enable
4	EPWM1_CTR_ZERO	EPWM1_CTR_ZERO	EPWM1_CTR_ZERO	EPWM1_CTR_ZERO	Disable
5	EPWM1_CTR_PRD	EPWM1_CTR_PRD	EPWM1_CTR_PRD	EPWM1_CTR_PRD	Disable
6	EPWM1_CTR_DIR	EPWM1_CTR_DIR	EPWM1_CTR_DIR	EPWM1_CTR_DIR	Disable
7	EPWM1_TBCLK	EPWM1_TBCLK	EPWM1_TBCLK	EPWM1_TBCLK	Disable
8	EPWM1_CTR_CMPA	EPWM1_CTR_CMPA	EPWM1_CTR_CMPA	EPWM1_CTR_CMPA	Disable
9	EPWM1_CTR_CMPB	EPWM1_CTR_CMPB	EPWM1_CTR_CMPB	EPWM1_CTR_CMPB	Disable
10	EPWM1_CTR_CMPC	EPWM1_CTR_CMPC	EPWM1_CTR_CMPC	EPWM1_CTR_CMPC	Disable

Table 9-2. Global Signals and Mux Selection (continued)

Select Value	CLB1 Input	CLB2 Input	CLB3 Input	CLB4 Input	Synchronization Requirement
11	EPWM1_CTR_CMPD	EPWM1_CTR_CMPD	EPWM1_CTR_CMPD	EPWM1_CTR_CMPD	Disable
12	EPWM1A_AQ	EPWM1A_AQ	EPWM1A_AQ	EPWM1A_AQ	Disable
13	EPWM1B_AQ	EPWM1B_AQ	EPWM1B_AQ	EPWM1B_AQ	Disable
14	EPWM1A_DB	EPWM1A_DB	EPWM1A_DB	EPWM1A_DB	Disable
15	EPWM1B_DB	EPWM1B_DB	EPWM1B_DB	EPWM1B_DB	Disable
16	EPWM2A	EPWM2A	EPWM2A	EPWM2A	Enable
17	EPWM2A_OE	EPWM2A_OE	EPWM2A_OE	EPWM2A_OE	Enable
18	EPWM2B	EPWM2B	EPWM2B	EPWM2B	Enable
19	EPWM2B_OE	EPWM2B_OE	EPWM2B_OE	EPWM2B_OE	Enable
20	EPWM2_CTR_ZERO	EPWM2_CTR_ZERO	EPWM2_CTR_ZERO	EPWM2_CTR_ZERO	Disable
21	EPWM2_CTR_PRD	EPWM2_CTR_PRD	EPWM2_CTR_PRD	EPWM2_CTR_PRD	Disable
22	EPWM2_CTR_DIR	EPWM2_CTR_DIR	EPWM2_CTR_DIR	EPWM2_CTR_DIR	Disable
23	EPWM2_TBCLK	EPWM2_TBCLK	EPWM2_TBCLK	EPWM2_TBCLK	Disable
24	EPWM2_CTR_CMPA	EPWM2_CTR_CMPA	EPWM2_CTR_CMPA	EPWM2_CTR_CMPA	Disable
25	EPWM2_CTR_CMPB	EPWM2_CTR_CMPB	EPWM2_CTR_CMPB	EPWM2_CTR_CMPB	Disable
26	EPWM2_CTR_CMPC	EPWM2_CTR_CMPC	EPWM2_CTR_CMPC	EPWM2_CTR_CMPC	Disable
27	EPWM2_CTR_CMPD	EPWM2_CTR_CMPD	EPWM2_CTR_CMPD	EPWM2_CTR_CMPD	Disable
28	EPWM2A_AQ	EPWM2A_AQ	EPWM2A_AQ	EPWM2A_AQ	Disable
29	EPWM2B_AQ	EPWM2B_AQ	EPWM2B_AQ	EPWM2B_AQ	Disable
30	EPWM2A_DB	EPWM2A_DB	EPWM2A_DB	EPWM2A_DB	Disable
31	EPWM2B_DB	EPWM2B_DB	EPWM2B_DB	EPWM2B_DB	Disable
32	EPWM3A	EPWM3A	EPWM3A	EPWM3A	Enable
33	EPWM3A_OE	EPWM3A_OE	EPWM3A_OE	EPWM3A_OE	Enable
34	EPWM3B	EPWM3B	EPWM3B	EPWM3B	Enable
35	EPWM3B_OE	EPWM3B_OE	EPWM3B_OE	EPWM3B_OE	Enable
36	EPWM3_CTR_ZERO	EPWM3_CTR_ZERO	EPWM3_CTR_ZERO	EPWM3_CTR_ZERO	Disable
37	EPWM3_CTR_PRD	EPWM3_CTR_PRD	EPWM3_CTR_PRD	EPWM3_CTR_PRD	Disable
38	EPWM3_CTR_DIR	EPWM3_CTR_DIR	EPWM3_CTR_DIR	EPWM3_CTR_DIR	Disable
39	EPWM3_TBCLK	EPWM3_TBCLK	EPWM3_TBCLK	EPWM3_TBCLK	Disable
40	EPWM3_CTR_CMPA	EPWM3_CTR_CMPA	EPWM3_CTR_CMPA	EPWM3_CTR_CMPA	Disable
41	EPWM3_CTR_CMPB	EPWM3_CTR_CMPB	EPWM3_CTR_CMPB	EPWM3_CTR_CMPB	Disable
42	EPWM3_CTR_CMPC	EPWM3_CTR_CMPC	EPWM3_CTR_CMPC	EPWM3_CTR_CMPC	Disable
43	EPWM3_CTR_CMPD	EPWM3_CTR_CMPD	EPWM3_CTR_CMPD	EPWM3_CTR_CMPD	Disable
44	EPWM3A_AQ	EPWM3A_AQ	EPWM3A_AQ	EPWM3A_AQ	Disable
45	EPWM3B_AQ	EPWM3B_AQ	EPWM3B_AQ	EPWM3B_AQ	Disable
46	EPWM3A_DB	EPWM3A_DB	EPWM3A_DB	EPWM3A_DB	Disable
47	EPWM3B_DB	EPWM3B_DB	EPWM3B_DB	EPWM3B_DB	Disable
48	EPWM4A	EPWM4A	EPWM4A	EPWM4A	Enable

Table 9-2. Global Signals and Mux Selection (continued)

Select Value	CLB1 Input	CLB2 Input	CLB3 Input	CLB4 Input	Synchronization Requirement
49	EPWM4A_OE	EPWM4A_OE	EPWM4A_OE	EPWM4A_OE	Enable
50	EPWM4B	EPWM4B	EPWM4B	EPWM4B	Enable
51	EPWM4B_OE	EPWM4B_OE	EPWM4B_OE	EPWM4B_OE	Enable
52	EPWM4_CTR_ZERO	EPWM4_CTR_ZERO	EPWM4_CTR_ZERO	EPWM4_CTR_ZERO	Disable
53	EPWM4_CTR_PRD	EPWM4_CTR_PRD	EPWM4_CTR_PRD	EPWM4_CTR_PRD	Disable
54	EPWM4_CTR_DIR	EPWM4_CTR_DIR	EPWM4_CTR_DIR	EPWM4_CTR_DIR	Disable
55	EPWM4_TBCLK	EPWM4_TBCLK	EPWM4_TBCLK	EPWM4_TBCLK	Disable
56	EPWM4_CTR_CMPA	EPWM4_CTR_CMPA	EPWM4_CTR_CMPA	EPWM4_CTR_CMPA	Disable
57	EPWM4_CTR_CMPB	EPWM4_CTR_CMPB	EPWM4_CTR_CMPB	EPWM4_CTR_CMPB	Disable
58	EPWM4_CTR_CMPC	EPWM4_CTR_CMPC	EPWM4_CTR_CMPC	EPWM4_CTR_CMPC	Disable
59	EPWM4_CTR_CMPD	EPWM4_CTR_CMPD	EPWM4_CTR_CMPD	EPWM4_CTR_CMPD	Disable
60	EPWM4A_AQ	EPWM4A_AQ	EPWM4A_AQ	EPWM4A_AQ	Disable
61	EPWM4B_AQ	EPWM4B_AQ	EPWM4B_AQ	EPWM4B_AQ	Disable
62	EPWM4A_DB	EPWM4A_DB	EPWM4A_DB	EPWM4A_DB	Disable
63	EPWM4B_DB	EPWM4B_DB	EPWM4B_DB	EPWM4B_DB	Disable
64	AUXSIG0	AUXSIG0	AUXSIG0	AUXSIG0	Enable
65	AUXSIG1	AUXSIG1	AUXSIG1	AUXSIG1	Enable
66	AUXSIG2	AUXSIG2	AUXSIG2	AUXSIG2	Enable
67	AUXSIG3	AUXSIG3	AUXSIG3	AUXSIG3	Enable
68	AUXSIG4	AUXSIG4	AUXSIG4	AUXSIG4	Enable
69	AUXSIG5	AUXSIG5	AUXSIG5	AUXSIG5	Enable
70	AUXSIG6	AUXSIG6	AUXSIG6	AUXSIG6	Enable
71	AUXSIG7	AUXSIG7	AUXSIG7	AUXSIG7	Enable
72	CLB1_OUT16	CLB1_OUT16	CLB1_OUT16	CLB1_OUT16	Disable
73	CLB1_OUT17	CLB1_OUT17	CLB1_OUT17	CLB1_OUT17	Disable
74	CLB1_OUT18	CLB1_OUT18	CLB1_OUT18	CLB1_OUT18	Disable
75	CLB1_OUT19	CLB1_OUT19	CLB1_OUT19	CLB1_OUT19	Disable
76	CLB1_OUT20	CLB1_OUT20	CLB1_OUT20	CLB1_OUT20	Disable
77	CLB1_OUT21	CLB1_OUT21	CLB1_OUT21	CLB1_OUT21	Disable
78	CLB1_OUT22	CLB1_OUT22	CLB1_OUT22	CLB1_OUT22	Disable
79	CLB1_OUT23	CLB1_OUT23	CLB1_OUT23	CLB1_OUT23	Disable
80	CLB2_OUT16	CLB2_OUT16	CLB2_OUT16	CLB2_OUT16	Disable
81	CLB2_OUT17	CLB2_OUT17	CLB2_OUT17	CLB2_OUT17	Disable
82	CLB2_OUT18	CLB2_OUT18	CLB2_OUT18	CLB2_OUT18	Disable
83	CLB2_OUT19	CLB2_OUT19	CLB2_OUT19	CLB2_OUT19	Disable
84	CLB2_OUT20	CLB2_OUT20	CLB2_OUT20	CLB2_OUT20	Disable
85	CLB2_OUT21	CLB2_OUT21	CLB2_OUT21	CLB2_OUT21	Disable
86	CLB2_OUT22	CLB2_OUT22	CLB2_OUT22	CLB2_OUT22	Disable

Table 9-2. Global Signals and Mux Selection (continued)

Select Value	CLB1 Input	CLB2 Input	CLB3 Input	CLB4 Input	Synchronization Requirement
87	CLB2_OUT23	CLB2_OUT23	CLB2_OUT23	CLB2_OUT23	Disable
88	CLB3_OUT16	CLB3_OUT16	CLB3_OUT16	CLB3_OUT16	Disable
89	CLB3_OUT17	CLB3_OUT17	CLB3_OUT17	CLB3_OUT17	Disable
90	CLB3_OUT18	CLB3_OUT18	CLB3_OUT18	CLB3_OUT18	Disable
91	CLB3_OUT19	CLB3_OUT19	CLB3_OUT19	CLB3_OUT19	Disable
92	CLB3_OUT20	CLB3_OUT20	CLB3_OUT20	CLB3_OUT20	Disable
93	CLB3_OUT21	CLB3_OUT21	CLB3_OUT21	CLB3_OUT21	Disable
94	CLB3_OUT22	CLB3_OUT22	CLB3_OUT22	CLB3_OUT22	Disable
95	CLB3_OUT23	CLB3_OUT23	CLB3_OUT23	CLB3_OUT23	Disable
96	CLB4_OUT16	CLB4_OUT16	CLB4_OUT16	CLB4_OUT16	Disable
97	CLB4_OUT17	CLB4_OUT17	CLB4_OUT17	CLB4_OUT17	Disable
98	CLB4_OUT18	CLB4_OUT18	CLB4_OUT18	CLB4_OUT18	Disable
99	CLB4_OUT19	CLB4_OUT19	CLB4_OUT19	CLB4_OUT19	Disable
100	CLB4_OUT20	CLB4_OUT20	CLB4_OUT20	CLB4_OUT20	Disable
101	CLB4_OUT21	CLB4_OUT21	CLB4_OUT21	CLB4_OUT21	Disable
102	CLB4_OUT22	CLB4_OUT22	CLB4_OUT22	CLB4_OUT22	Disable
103	CLB4_OUT23	CLB4_OUT23	CLB4_OUT23	CLB4_OUT23	Disable
104	ERAD_EVT0	ERAD_EVT0	ERAD_EVT0	ERAD_EVT0	Disable
105	ERAD_EVT1	ERAD_EVT1	ERAD_EVT1	ERAD_EVT1	Disable
106	ERAD_EVT2	ERAD_EVT2	ERAD_EVT2	ERAD_EVT2	Disable
107	ERAD_EVT3	ERAD_EVT3	ERAD_EVT3	ERAD_EVT3	Disable
108	ERAD_EVT4	ERAD_EVT4	ERAD_EVT4	ERAD_EVT4	Disable
109	ERAD_EVT5	ERAD_EVT5	ERAD_EVT5	ERAD_EVT5	Disable
110	ERAD_EVT6	ERAD_EVT6	ERAD_EVT6	ERAD_EVT6	Disable
111	ERAD_EVT7	ERAD_EVT7	ERAD_EVT7	ERAD_EVT7	Disable
112	FSIRXA_DATA_PKT_RCVD	FSIRXA_DATA_PKT_RCVD	FSIRXA_DATA_PKT_RCVD	FSIRXA_DATA_PKT_RCVD	Disable
113	FSIRXA_ERROR_PKT_RCVD	FSIRXA_ERROR_PKT_RCVD	FSIRXA_ERROR_PKT_RCVD	FSIRXA_ERROR_PKT_RCVD	Disable
114	FSIRXA_PING_PKT_RCVD	FSIRXA_PING_PKT_RCVD	FSIRXA_PING_PKT_RCVD	FSIRXA_PING_PKT_RCVD	Disable
115	FSIRXA_FRAME_DONE	FSIRXA_FRAME_DONE	FSIRXA_FRAME_DONE	FSIRXA_FRAME_DONE	Disable
116	FSIRXA_PKT_TAG0	FSIRXA_PKT_TAG0	FSIRXA_PKT_TAG0	FSIRXA_PKT_TAG0	Disable
117	FSIRXA_PKT_TAG1	FSIRXA_PKT_TAG1	FSIRXA_PKT_TAG1	FSIRXA_PKT_TAG1	Disable
118	FSIRXA_PKT_TAG2	FSIRXA_PKT_TAG2	FSIRXA_PKT_TAG2	FSIRXA_PKT_TAG2	Disable
119	FSIRXA_PKT_TAG3	FSIRXA_PKT_TAG3	FSIRXA_PKT_TAG3	FSIRXA_PKT_TAG3	Disable
120	SPIA_CLK_OUT	SPIA_CLK_OUT	SPIA_CLK_OUT	SPIA_CLK_OUT	Enable
121	SPIA_SOMI_IN	SPIA_SOMI_IN	SPIA_SOMI_IN	SPIA_SOMI_IN	Enable

Table 9-2. Global Signals and Mux Selection (continued)

Select Value	CLB1 Input	CLB2 Input	CLB3 Input	CLB4 Input	Synchronization Requirement
122	SPIA_STE_OUT	SPIA_STE_OUT	SPIA_STE_OUT	SPIA_STE_OUT	Enable
123	SPIB_CLK_OUT	SPIB_CLK_OUT	SPIB_CLK_OUT	SPIB_CLK_OUT	Enable
124	SPIB_SOMI_IN	SPIB_SOMI_IN	SPIB_SOMI_IN	SPIB_SOMI_IN	Enable
125	SPIB_STE_OUT	SPIB_STE_OUT	SPIB_STE_OUT	SPIB_STE_OUT	Enable
126	CPU2_HALT	CPU2_HALT	CPU2_HALT	CPU2_HALT	Disable
127	Reserved	Reserved	Reserved	Reserved	Reserved

Table 9-3. Global Signals and Mux Selection

Select Value	CLB5 Input	CLB6 Input	CLB7 Input	CLB8 Input	Synchronization Requirement
0	EPWM5A	EPWM5A	EPWM5A	EPWM5A	Enable
1	EPWM5A_OE	EPWM5A_OE	EPWM5A_OE	EPWM5A_OE	Enable
2	EPWM5B	EPWM5B	EPWM5B	EPWM5B	Enable
3	EPWM5B_OE	EPWM5B_OE	EPWM5B_OE	EPWM5B_OE	Enable
4	EPWM5_CTR_ZERO	EPWM5_CTR_ZERO	EPWM5_CTR_ZERO	EPWM5_CTR_ZERO	Disable
5	EPWM5_CTR_PRD	EPWM5_CTR_PRD	EPWM5_CTR_PRD	EPWM5_CTR_PRD	Disable
6	EPWM5_CTR_DIR	EPWM5_CTR_DIR	EPWM5_CTR_DIR	EPWM5_CTR_DIR	Disable
7	EPWM5_TBCLK	EPWM5_TBCLK	EPWM5_TBCLK	EPWM5_TBCLK	Disable
8	EPWM5_CTR_CMPA	EPWM5_CTR_CMPA	EPWM5_CTR_CMPA	EPWM5_CTR_CMPA	Disable
9	EPWM5_CTR_CMPB	EPWM5_CTR_CMPB	EPWM5_CTR_CMPB	EPWM5_CTR_CMPB	Disable
10	EPWM5_CTR_CMPC	EPWM5_CTR_CMPC	EPWM5_CTR_CMPC	EPWM5_CTR_CMPC	Disable
11	EPWM5_CTR_CMPD	EPWM5_CTR_CMPD	EPWM5_CTR_CMPD	EPWM5_CTR_CMPD	Disable
12	EPWM5A_AQ	EPWM5A_AQ	EPWM5A_AQ	EPWM5A_AQ	Disable
13	EPWM5B_AQ	EPWM5B_AQ	EPWM5B_AQ	EPWM5B_AQ	Disable
14	EPWM5A_DB	EPWM5A_DB	EPWM5A_DB	EPWM5A_DB	Disable
15	EPWM5B_DB	EPWM5B_DB	EPWM5B_DB	EPWM5B_DB	Disable
16	EPWM6A	EPWM6A	EPWM6A	EPWM6A	Enable
17	EPWM6A_OE	EPWM6A_OE	EPWM6A_OE	EPWM6A_OE	Enable
18	EPWM6B	EPWM6B	EPWM6B	EPWM6B	Enable
19	EPWM6B_OE	EPWM6B_OE	EPWM6B_OE	EPWM6B_OE	Enable
20	EPWM6_CTR_ZERO	EPWM6_CTR_ZERO	EPWM6_CTR_ZERO	EPWM6_CTR_ZERO	Disable
21	EPWM6_CTR_PRD	EPWM6_CTR_PRD	EPWM6_CTR_PRD	EPWM6_CTR_PRD	Disable
22	EPWM6_CTR_DIR	EPWM6_CTR_DIR	EPWM6_CTR_DIR	EPWM6_CTR_DIR	Disable
23	EPWM6_TBCLK	EPWM6_TBCLK	EPWM6_TBCLK	EPWM6_TBCLK	Disable
24	EPWM6_CTR_CMPA	EPWM6_CTR_CMPA	EPWM6_CTR_CMPA	EPWM6_CTR_CMPA	Disable
25	EPWM6_CTR_CMPB	EPWM6_CTR_CMPB	EPWM6_CTR_CMPB	EPWM6_CTR_CMPB	Disable
26	EPWM6_CTR_CMPC	EPWM6_CTR_CMPC	EPWM6_CTR_CMPC	EPWM6_CTR_CMPC	Disable
27	EPWM6_CTR_CMPD	EPWM6_CTR_CMPD	EPWM6_CTR_CMPD	EPWM6_CTR_CMPD	Disable

Table 9-3. Global Signals and Mux Selection (continued)

Select Value	CLB5 Input	CLB6 Input	CLB7 Input	CLB8 Input	Synchronization Requirement
28	EPWM6A_AQ	EPWM6A_AQ	EPWM6A_AQ	EPWM6A_AQ	Disable
29	EPWM6B_AQ	EPWM6B_AQ	EPWM6B_AQ	EPWM6B_AQ	Disable
30	EPWM6A_DB	EPWM6A_DB	EPWM6A_DB	EPWM6A_DB	Disable
31	EPWM6B_DB	EPWM6B_DB	EPWM6B_DB	EPWM6B_DB	Disable
32	EPWM7A	EPWM7A	EPWM7A	EPWM7A	Enable
33	EPWM7A_OE	EPWM7A_OE	EPWM7A_OE	EPWM7A_OE	Enable
34	EPWM7B	EPWM7B	EPWM7B	EPWM7B	Enable
35	EPWM7B_OE	EPWM7B_OE	EPWM7B_OE	EPWM7B_OE	Enable
36	EPWM7_CTR_ZERO	EPWM7_CTR_ZERO	EPWM7_CTR_ZERO	EPWM7_CTR_ZERO	Disable
37	EPWM7_CTR_PRD	EPWM7_CTR_PRD	EPWM7_CTR_PRD	EPWM7_CTR_PRD	Disable
38	EPWM7_CTR_DIR	EPWM7_CTR_DIR	EPWM7_CTR_DIR	EPWM7_CTR_DIR	Disable
39	EPWM7_TBCLK	EPWM7_TBCLK	EPWM7_TBCLK	EPWM7_TBCLK	Disable
40	EPWM7_CTR_CMPA	EPWM7_CTR_CMPA	EPWM7_CTR_CMPA	EPWM7_CTR_CMPA	Disable
41	EPWM7_CTR_CMPB	EPWM7_CTR_CMPB	EPWM7_CTR_CMPB	EPWM7_CTR_CMPB	Disable
42	EPWM7_CTR_CMPC	EPWM7_CTR_CMPC	EPWM7_CTR_CMPC	EPWM7_CTR_CMPC	Disable
43	EPWM7_CTR_CMPD	EPWM7_CTR_CMPD	EPWM7_CTR_CMPD	EPWM7_CTR_CMPD	Disable
44	EPWM7A_AQ	EPWM7A_AQ	EPWM7A_AQ	EPWM7A_AQ	Disable
45	EPWM7B_AQ	EPWM7B_AQ	EPWM7B_AQ	EPWM7B_AQ	Disable
46	EPWM7A_DB	EPWM7A_DB	EPWM7A_DB	EPWM7A_DB	Disable
47	EPWM7B_DB	EPWM7B_DB	EPWM7B_DB	EPWM7B_DB	Disable
48	EPWM8A	EPWM8A	EPWM8A	EPWM8A	Enable
49	EPWM8A_OE	EPWM8A_OE	EPWM8A_OE	EPWM8A_OE	Enable
50	EPWM8B	EPWM8B	EPWM8B	EPWM8B	Enable
51	EPWM8B_OE	EPWM8B_OE	EPWM8B_OE	EPWM8B_OE	Enable
52	EPWM8_CTR_ZERO	EPWM8_CTR_ZERO	EPWM8_CTR_ZERO	EPWM8_CTR_ZERO	Disable
53	EPWM8_CTR_PRD	EPWM8_CTR_PRD	EPWM8_CTR_PRD	EPWM8_CTR_PRD	Disable
54	EPWM8_CTR_DIR	EPWM8_CTR_DIR	EPWM8_CTR_DIR	EPWM8_CTR_DIR	Disable
55	EPWM8_TBCLK	EPWM8_TBCLK	EPWM8_TBCLK	EPWM8_TBCLK	Disable
56	EPWM8_CTR_CMPA	EPWM8_CTR_CMPA	EPWM8_CTR_CMPA	EPWM8_CTR_CMPA	Disable
57	EPWM8_CTR_CMPB	EPWM8_CTR_CMPB	EPWM8_CTR_CMPB	EPWM8_CTR_CMPB	Disable
58	EPWM8_CTR_CMPC	EPWM8_CTR_CMPC	EPWM8_CTR_CMPC	EPWM8_CTR_CMPC	Disable
59	EPWM8_CTR_CMPD	EPWM8_CTR_CMPD	EPWM8_CTR_CMPD	EPWM8_CTR_CMPD	Disable
60	EPWM8A_AQ	EPWM8A_AQ	EPWM8A_AQ	EPWM8A_AQ	Disable
61	EPWM8B_AQ	EPWM8B_AQ	EPWM8B_AQ	EPWM8B_AQ	Disable
62	EPWM8A_DB	EPWM8A_DB	EPWM8A_DB	EPWM8A_DB	Disable
63	EPWM8B_DB	EPWM8B_DB	EPWM8B_DB	EPWM8B_DB	Disable
64	AUXSIG0	AUXSIG0	AUXSIG0	AUXSIG0	Enable
65	AUXSIG1	AUXSIG1	AUXSIG1	AUXSIG1	Enable

Table 9-3. Global Signals and Mux Selection (continued)

Select Value	CLB5 Input	CLB6 Input	CLB7 Input	CLB8 Input	Synchronization Requirement
66	AUXSIG2	AUXSIG2	AUXSIG2	AUXSIG2	Enable
67	AUXSIG3	AUXSIG3	AUXSIG3	AUXSIG3	Enable
68	AUXSIG4	AUXSIG4	AUXSIG4	AUXSIG4	Enable
69	AUXSIG5	AUXSIG5	AUXSIG5	AUXSIG5	Enable
70	AUXSIG6	AUXSIG6	AUXSIG6	AUXSIG6	Enable
71	AUXSIG7	AUXSIG7	AUXSIG7	AUXSIG7	Enable
72	CLB5_OUT16	CLB5_OUT16	CLB5_OUT16	CLB5_OUT16	Disable
73	CLB5_OUT17	CLB5_OUT17	CLB5_OUT17	CLB5_OUT17	Disable
74	CLB5_OUT18	CLB5_OUT18	CLB5_OUT18	CLB5_OUT18	Disable
75	CLB5_OUT19	CLB5_OUT19	CLB5_OUT19	CLB5_OUT19	Disable
76	CLB5_OUT20	CLB5_OUT20	CLB5_OUT20	CLB5_OUT20	Disable
77	CLB5_OUT21	CLB5_OUT21	CLB5_OUT21	CLB5_OUT21	Disable
78	CLB5_OUT22	CLB5_OUT22	CLB5_OUT22	CLB5_OUT22	Disable
79	CLB5_OUT23	CLB5_OUT23	CLB5_OUT23	CLB5_OUT23	Disable
80	CLB6_OUT16	CLB6_OUT16	CLB6_OUT16	CLB6_OUT16	Disable
81	CLB6_OUT17	CLB6_OUT17	CLB6_OUT17	CLB6_OUT17	Disable
82	CLB6_OUT18	CLB6_OUT18	CLB6_OUT18	CLB6_OUT18	Disable
83	CLB6_OUT19	CLB6_OUT19	CLB6_OUT19	CLB6_OUT19	Disable
84	CLB6_OUT20	CLB6_OUT20	CLB6_OUT20	CLB6_OUT20	Disable
85	CLB6_OUT21	CLB6_OUT21	CLB6_OUT21	CLB6_OUT21	Disable
86	CLB6_OUT22	CLB6_OUT22	CLB6_OUT22	CLB6_OUT22	Disable
87	CLB6_OUT23	CLB6_OUT23	CLB6_OUT23	CLB6_OUT23	Disable
88	CLB3_OUT16	CLB3_OUT16	CLB3_OUT16	CLB3_OUT16	Disable
89	CLB3_OUT17	CLB3_OUT17	CLB3_OUT17	CLB3_OUT17	Disable
90	CLB3_OUT18	CLB3_OUT18	CLB3_OUT18	CLB3_OUT18	Disable
91	CLB3_OUT19	CLB3_OUT19	CLB3_OUT19	CLB3_OUT19	Disable
92	CLB3_OUT20	CLB3_OUT20	CLB3_OUT20	CLB3_OUT20	Disable
93	CLB3_OUT21	CLB3_OUT21	CLB3_OUT21	CLB3_OUT21	Disable
94	CLB3_OUT22	CLB3_OUT22	CLB3_OUT22	CLB3_OUT22	Disable
95	CLB3_OUT23	CLB3_OUT23	CLB3_OUT23	CLB3_OUT23	Disable
96	CLB4_OUT16	CLB4_OUT16	CLB4_OUT16	CLB4_OUT16	Disable
97	CLB4_OUT17	CLB4_OUT17	CLB4_OUT17	CLB4_OUT17	Disable
98	CLB4_OUT18	CLB4_OUT18	CLB4_OUT18	CLB4_OUT18	Disable
99	CLB4_OUT19	CLB4_OUT19	CLB4_OUT19	CLB4_OUT19	Disable
100	CLB4_OUT20	CLB4_OUT20	CLB4_OUT20	CLB4_OUT20	Disable
101	CLB4_OUT21	CLB4_OUT21	CLB4_OUT21	CLB4_OUT21	Disable
102	CLB4_OUT22	CLB4_OUT22	CLB4_OUT22	CLB4_OUT22	Disable
103	CLB4_OUT23	CLB4_OUT23	CLB4_OUT23	CLB4_OUT23	Disable

Table 9-3. Global Signals and Mux Selection (continued)

Select Value	CLB5 Input	CLB6 Input	CLB7 Input	CLB8 Input	Synchronization Requirement
104	ERAD_EVT0	ERAD_EVT0	ERAD_EVT0	ERAD_EVT0	Disable
105	ERAD_EVT1	ERAD_EVT1	ERAD_EVT1	ERAD_EVT1	Disable
106	ERAD_EVT2	ERAD_EVT2	ERAD_EVT2	ERAD_EVT2	Disable
107	ERAD_EVT3	ERAD_EVT3	ERAD_EVT3	ERAD_EVT3	Disable
108	ERAD_EVT4	ERAD_EVT4	ERAD_EVT4	ERAD_EVT4	Disable
109	ERAD_EVT5	ERAD_EVT5	ERAD_EVT5	ERAD_EVT5	Disable
110	ERAD_EVT6	ERAD_EVT6	ERAD_EVT6	ERAD_EVT6	Disable
111	ERAD_EVT7	ERAD_EVT7	ERAD_EVT7	ERAD_EVT7	Disable
112	FSIRXA_PING_TAG_MATCH	FSIRXA_PING_TAG_MATCH	FSIRXA_PING_TAG_MATCH	FSIRXA_PING_TAG_MATCH	Disable
113	FSIRXA_DATA_TAG_MATCH	FSIRXA_DATA_TAG_MATCH	FSIRXA_DATA_TAG_MATCH	FSIRXA_DATA_TAG_MATCH	Disable
114	FSIRXA_ERROR_TAG_MATCH	FSIRXA_ERROR_TAG_MATCH	FSIRXA_ERROR_TAG_MATCH	FSIRXA_ERROR_TAG_MATCH	Disable
115	FSIRXB_PING_TAG_MATCH	FSIRXB_PING_TAG_MATCH	FSIRXB_PING_TAG_MATCH	FSIRXB_PING_TAG_MATCH	Disable
116	FSIRXB_DATA_TAG_MATCH	FSIRXB_DATA_TAG_MATCH	FSIRXB_DATA_TAG_MATCH	FSIRXB_DATA_TAG_MATCH	Disable
117	FSIRXB_ERROR_TAG_MATCH	FSIRXB_ERROR_TAG_MATCH	FSIRXB_ERROR_TAG_MATCH	FSIRXB_ERROR_TAG_MATCH	Disable
118	ECAT_SOF	ECAT_SOF	ECAT_SOF	ECAT_SOF	Disable
119	ECAT_EOF	ECAT_EOF	ECAT_EOF	ECAT_EOF	Disable
120	SPIC_CLK_OUT	SPIC_CLK_OUT	SPIC_CLK_OUT	SPIC_CLK_OUT	Enable
121	SPIC_SOMI_IN	SPIC_SOMI_IN	SPIC_SOMI_IN	SPIC_SOMI_IN	Enable
122	SPIC_STE_OUT	SPIC_STE_OUT	SPIC_STE_OUT	SPIC_STE_OUT	Enable
123	SPID_CLK_OUT	SPID_CLK_OUT	SPID_CLK_OUT	SPID_CLK_OUT	Enable
124	SPID_SOMI_IN	SPID_SOMI_IN	SPID_SOMI_IN	SPID_SOMI_IN	Enable
125	SPID_STE_OUT	SPID_STE_OUT	SPID_STE_OUT	SPID_STE_OUT	Enable
126	ECAT_SYNC0	ECAT_SYNC0	ECAT_SYNC0	ECAT_SYNC0	Disable
127	ECAT_SYNC1	ECAT_SYNC1	ECAT_SYNC1	ECAT_SYNC1	Disable

Note

EPWMxA_OE and EPWMxB_OE refer to trip outputs from the respective EPWM module.

EPWMxA_AQ and EPWMxB_AQ refer to the output of the AQ submodule in the respective EPWM module.

EPWMxA_DB and EPWMBx_DB refer to the output of the DB submodule in the respective EPWM module.

ECAT_SYNC0 and ECAT_SYNC1 must not be gated off, if EtherCAT is allocated to CM.

Note

If a signal in the following table indicates that synchronization is required, then the CLB input synchronizer must be enabled using the appropriate SYNC bit in the CLB_INPUT_FILTER register. This synchronization adds a 2-3 CLB clock cycle delay to the input. This delay is either 2 or 3 cycles and is not predictable. There is a potential for a metastability hazard, if the indicated signals are not first synchronized before going into the CLB tile. This metastability can cause errors dependent on voltage, temperature, and wafer fab process. Note that this requirement is in addition to and separate from GPIO input synchronization.

If a signal in the following table indicates that synchronization is not required, as the signal is already synchronous, then pipelining is required and must be enabled using the PIPE bit in the CLB_INPUT_FILTER register. This pipelining adds a 1 CLB clock cycle delay to the input. This is not to be mistaken with the PIPELINE_EN bit in the CLB_LOAD_EN register, which controls pipelining of the CLB operations in the HLC and counter blocks. This PIPELINE_EN bit is also used when the device is run above 100 MHz. Having synchronization and pipelining both enabled or both disabled is not recommended. Enabling both synchronization and pipelining introduces a delay of more than 2-3 CLB clock cycles on the signal path. Disabling both allows the completely asynchronous signal to be routed as an input.

Table 9-4. Local Signals and Mux Selection

Select Value	CLB1 Input	CLB2 Input	CLB3 Input	CLB4 Input	Synchronization Requirement
0	CLB1_GLB_MUX_OUT	CLB2_GLB_MUX_OUT	CLB3_GLB_MUX_OUT	CLB4_GLB_MUX_OUT	Enable
1	EPWM1_DCAEVT1	EPWM2_DCAEVT1	EPWM3_DCAEVT1	EPWM4_DCAEVT1	Enable
2	EPWM1_DCAEVT2	EPWM2_DCAEVT2	EPWM3_DCAEVT2	EPWM4_DCAEVT2	Enable
3	EPWM1_DCBEVT1	EPWM2_DCBEVT1	EPWM3_DCBEVT1	EPWM4_DCBEVT1	Enable
4	EPWM1_DCBEVT2	EPWM2_DCBEVT2	EPWM3_DCBEVT2	EPWM4_DCBEVT2	Enable
5	EPWM1_DCAH	EPWM2_DCAH	EPWM3_DCAH	EPWM4_DCAH	Enable
6	EPWM1_DCAL	EPWM2_DCAL	EPWM3_DCAL	EPWM4_DCAL	Enable
7	EPWM1_DCBH	EPWM2_DCBH	EPWM3_DCBH	EPWM4_DCBH	Enable
8	EPWM1_DCBL	EPWM2_DCBL	EPWM3_DCBL	EPWM4_DCBL	Enable
9	EPWM1_OST	EPWM2_OST	EPWM3_OST	EPWM4_OST	Enable
10	EPWM1_CBC	EPWM2_CBC	EPWM3_CBC	EPWM4_CBC	Enable
11	ECAP1IN0	ECAP2IN0	ECAP6IN0	ECAP7IN0	Enable
12	ECAP1_OUT	ECAP2_OUT	ECAP6_OUT	ECAP7_OUT	Disable
13	ECAP1_OUT_EN	ECAP2_OUT_EN	ECAP6_OUT_EN	ECAP7_OUT_EN	Disable
14	ECAP1_CEVT1	ECAP2_CEVT1	ECAP6_CEVT1	ECAP7_CEVT1	Disable
15	ECAP1_CEVT2	ECAP2_CEVT2	ECAP6_CEVT2	ECAP7_CEVT2	Disable
16	ECAP1_CEVT3	ECAP2_CEVT3	ECAP6_CEVT3	ECAP7_CEVT3	Disable
17	ECAP1_CEVT4	ECAP2_CEVT4	ECAP6_CEVT4	ECAP7_CEVT4	Disable
18	EQEP1A	EQEP2A	EQEP3A	Reserved	Enable
19	EQEP1B	EQEP2B	EQEP3B	Reserved	Enable
20	EQEP1I	EQEP2I	EQEP3I	Reserved	Enable
21	EQEP1S	EQEP2S	EQEP3S	Reserved	Enable
22	CPU1_TBCLKSYNC	CPU1_TBCLKSYNC	CPU1_TBCLKSYNC	CPU1_TBCLKSYNC	Enable

Table 9-4. Local Signals and Mux Selection (continued)

Select Value	CLB1 Input	CLB2 Input	CLB3 Input	CLB4 Input	Synchronization Requirement
23	CPU2_TBCLKSYNC	CPU2_TBCLKSYNC	CPU2_TBCLKSYNC	CPU2_TBCLKSYNC	Enable
24	CPU1_HALT	CPU1_HALT	CPU1_HALT	CPU1_HALT	Enable
25	SPIA_SIMO_OUT	SPIB_SIMO_OUT	SPIC_SIMO_OUT	SPID_SIMO_OUT	Enable
26	SPIA_CLK_IN	SPIB_CLK_IN	SPIC_CLK_IN	SPID_CLK_IN	Enable
27	SPIA_SIMO_IN	SPIB_SIMO_IN	SPIC_SIMO_IN	SPID_SIMO_IN	Enable
28	SPIA_STE_IN	SPIB_STE_IN	SPIC_STE_IN	SPID_STE_IN	Enable
29	SCIA_TX	SCIB_TX	SCIA_TX	SCIB_TX	Enable
30	SPIA_SOMI_OUT	SPIB_SOMI_OUT	SPIC_SOMI_OUT	SPID_SOMI_OUT	Enable
31	CLB1_PSCLK	CLB2_PSCLK	CLB3_PSCLK	CLB4_PSCLK	Enable
32	EPWM9A	EPWM9A	EPWM9A	EPWM9A	Enable
33	EPWM9A_OE	EPWM9A_OE	EPWM9A_OE	EPWM9A_OE	Enable
34	EPWM9B	EPWM9B	EPWM9B	EPWM9B	Enable
35	EPWM9B_OE	EPWM9B_OE	EPWM9B_OE	EPWM9B_OE	Enable
36	EPWM10A	EPWM10A	EPWM10A	EPWM10A	Enable
37	EPWM10A_OE	EPWM10A_OE	EPWM10A_OE	EPWM10A_OE	Enable
38	EPWM10B	EPWM10B	EPWM10B	EPWM10B	Enable
39	EPWM10B_OE	EPWM10B_OE	EPWM10B_OE	EPWM10B_OE	Enable
40	EPWM11A	EPWM11A	EPWM11A	EPWM11A	Enable
41	EPWM11A_OE	EPWM11A_OE	EPWM11A_OE	EPWM11A_OE	Enable
42	EPWM11B	EPWM11B	EPWM11B	EPWM11B	Enable
43	EPWM11B_OE	EPWM11B_OE	EPWM11B_OE	EPWM11B_OE	Enable
44	EPWM12A	EPWM12A	EPWM12A	EPWM12A	Enable
45	EPWM12A_OE	EPWM12A_OE	EPWM12A_OE	EPWM12A_OE	Enable
46	EPWM12B	EPWM12B	EPWM12B	EPWM12B	Enable
47	EPWM12B_OE	EPWM12B_OE	EPWM12B_OE	EPWM12B_OE	Enable
48	CLBINPUTXBAR1	CLBINPUTXBAR1	CLBINPUTXBAR1	CLBINPUTXBAR1	Enable
49	CLBINPUTXBAR2	CLBINPUTXBAR2	CLBINPUTXBAR2	CLBINPUTXBAR2	Enable
50	CLBINPUTXBAR3	CLBINPUTXBAR3	CLBINPUTXBAR3	CLBINPUTXBAR3	Enable
51	CLBINPUTXBAR4	CLBINPUTXBAR4	CLBINPUTXBAR4	CLBINPUTXBAR4	Enable
52	CLBINPUTXBAR5	CLBINPUTXBAR5	CLBINPUTXBAR5	CLBINPUTXBAR5	Enable
53	CLBINPUTXBAR6	CLBINPUTXBAR6	CLBINPUTXBAR6	CLBINPUTXBAR6	Enable
54	CLBINPUTXBAR7	CLBINPUTXBAR7	CLBINPUTXBAR7	CLBINPUTXBAR7	Enable
55	CLBINPUTXBAR8	CLBINPUTXBAR8	CLBINPUTXBAR8	CLBINPUTXBAR8	Enable
56	CLBINPUTXBAR9	CLBINPUTXBAR9	CLBINPUTXBAR9	CLBINPUTXBAR9	Enable
57	CLBINPUTXBAR10	CLBINPUTXBAR10	CLBINPUTXBAR10	CLBINPUTXBAR10	Enable
58	CLBINPUTXBAR11	CLBINPUTXBAR11	CLBINPUTXBAR11	CLBINPUTXBAR11	Enable
59	CLBINPUTXBAR12	CLBINPUTXBAR12	CLBINPUTXBAR12	CLBINPUTXBAR12	Enable
60	CLBINPUTXBAR13	CLBINPUTXBAR13	CLBINPUTXBAR13	CLBINPUTXBAR13	Enable

Table 9-4. Local Signals and Mux Selection (continued)

Select Value	CLB1 Input	CLB2 Input	CLB3 Input	CLB4 Input	Synchronization Requirement
61	CLBINPUTXBAR14	CLBINPUTXBAR14	CLBINPUTXBAR14	CLBINPUTXBAR14	Enable
62	CLBINPUTXBAR15	CLBINPUTXBAR15	CLBINPUTXBAR15	CLBINPUTXBAR15	Enable
63	CLBINPUTXBAR16	CLBINPUTXBAR16	CLBINPUTXBAR16	CLBINPUTXBAR16	Enable

Table 9-5. Local Signals and Mux Selection

Select Value	CLB5 Input	CLB6 Input	CLB7 Input	CLB8 Input	Synchronization Requirement
0	CLB5_GLB_MUX_OUT	CLB6_GLB_MUX_OUT	CLB7_GLB_MUX_OUT	CLB8_GLB_MUX_OUT	Enable
1	EPWM5_DCAEVT1	EPWM6_DCAEVT1	EPWM7_DCAEVT1	EPWM8_DCAEVT1	Enable
2	EPWM5_DCAEVT2	EPWM6_DCAEVT2	EPWM7_DCAEVT2	EPWM8_DCAEVT2	Enable
3	EPWM5_DCBEVT1	EPWM6_DCBEVT1	EPWM7_DCBEVT1	EPWM8_DCBEVT1	Enable
4	EPWM5_DCBEVT2	EPWM6_DCBEVT2	EPWM7_DCBEVT2	EPWM8_DCBEVT2	Enable
5	EPWM5_DCAH	EPWM6_DCAH	EPWM7_DCAH	EPWM8_DCAH	Enable
6	EPWM5_DCAL	EPWM6_DCAL	EPWM7_DCAL	EPWM8_DCAL	Enable
7	EPWM5_DCBH	EPWM6_DCBH	EPWM7_DCBH	EPWM8_DCBH	Enable
8	EPWM5_DCBL	EPWM6_DCBL	EPWM7_DCBL	EPWM8_DCBL	Enable
9	EPWM5_OST	EPWM6_OST	EPWM7_OST	EPWM8_OST	Enable
10	EPWM5_CBC	EPWM6_CBC	EPWM7_CBC	EPWM8_CBC	Enable
11	ECAP3IN0	ECAP4IN0	ECAP6IN0	ECAP7IN0	Enable
12	ECAP3_OUT	ECAP4_OUT	ECAP5_OUT	ECAP6_OUT	Disable
13	ECAP3_OUT_EN	ECAP4_OUT_EN	ECAP5_OUT_EN	ECAP6_OUT_EN	Disable
14	ECAP3_CEVT1	ECAP4_CEVT1	ECAP5_CEVT1	ECAP6_CEVT1	Disable
15	ECAP3_CEVT2	ECAP4_CEVT2	ECAP5_CEVT2	ECAP6_CEVT2	Disable
16	ECAP3_CEVT3	ECAP4_CEVT3	ECAP5_CEVT3	ECAP6_CEVT3	Disable
17	ECAP3_CEVT4	ECAP4_CEVT4	ECAP5_CEVT4	ECAP6_CEVT4	Disable
18	FSIRXC_DATA_PKT_RCVD	FSIRXD_DATA_PKT_RCVD	FSIRXE_DATA_PKT_RCVD	FSIRXF_DATA_PKT_RCVD	Enable
19	FSIRXC_ERROR_PKT_RCVD	FSIRXD_ERROR_PKT_RCVD	FSIRXE_ERROR_PKT_RCVD	FSIRXF_ERROR_PKT_RCVD	Enable
20	FSIRXC_PING_PKT_RCVD	FSIRXD_PING_PKT_RCVD	FSIRXE_PING_PKT_RCVD	FSIRXF_PING_PKT_RCVD	Enable
21	CPU2_HALT	CPU2_HALT	CPU2_HALT	CPU2_HALT	Enable
22	CPU1_TBCLKSYNC	CPU1_TBCLKSYNC	CPU1_TBCLKSYNC	CPU1_TBCLKSYNC	Enable
23	CPU2_TBCLKSYNC	CPU2_TBCLKSYNC	CPU2_TBCLKSYNC	CPU2_TBCLKSYNC	Enable
24	CPU1_HALT	CPU1_HALT	CPU1_HALT	CPU1_HALT	Enable
25	SPIC_SIMO_OUT	SPID_SIMO_OUT	SPIC_SIMO_OUT	SPID_SIMO_OUT	Enable
26	SPIC_CLK_IN	SPID_CLK_IN	SPIC_CLK_IN	SPID_CLK_IN	Enable
27	SPIC_SIMO_IN	SPID_SIMO_IN	SPIC_SIMO_IN	SPID_SIMO_IN	Enable

Table 9-5. Local Signals and Mux Selection (continued)

Select Value	CLB5 Input	CLB6 Input	CLB7 Input	CLB8 Input	Synchronization Requirement
28	SPIC_STE_IN	SPID_STE_IN	SPIC_STE_IN	SPID_STE_IN	Enable
29	SCIC_TX	SCID_TX	SCIC_TX	SCID_TX	Enable
30	SPIC_SOMI_OUT	SPID_SOMI_OUT	SPIC_SOMI_OUT	SPID_SOMI_OUT	Enable
31	CLB5_PSCLK	CLB6_PSCLK	CLB7_PSCLK	CLB8_PSCLK	Enable
32	EPWM13A	EPWM13A	EPWM13A	EPWM13A	Enable
33	EPWM13A_OE	EPWM13A_OE	EPWM13A_OE	EPWM13A_OE	Enable
34	EPWM13B	EPWM13B	EPWM13B	EPWM13B	Enable
35	EPWM13B_OE	EPWM13B_OE	EPWM13B_OE	EPWM13B_OE	Enable
36	EPWM14A	EPWM14A	EPWM14A	EPWM14A	Enable
37	EPWM14A_OE	EPWM14A_OE	EPWM14A_OE	EPWM14A_OE	Enable
38	EPWM14B	EPWM14B	EPWM14B	EPWM14B	Enable
39	EPWM14B_OE	EPWM14B_OE	EPWM14B_OE	EPWM14B_OE	Enable
40	EPWM15A	EPWM15A	EPWM15A	EPWM15A	Enable
41	EPWM15A_OE	EPWM15A_OE	EPWM15A_OE	EPWM15A_OE	Enable
42	EPWM15B	EPWM15B	EPWM15B	EPWM15B	Enable
43	EPWM15B_OE	EPWM15B_OE	EPWM15B_OE	EPWM15B_OE	Enable
44	EPWM16A	EPWM16A	EPWM16A	EPWM16A	Enable
45	EPWM16A_OE	EPWM16A_OE	EPWM16A_OE	EPWM16A_OE	Enable
46	EPWM16B	EPWM16B	EPWM16B	EPWM16B	Enable
47	EPWM16B_OE	EPWM16B_OE	EPWM16B_OE	EPWM16B_OE	Enable
48	CLBINPUTXBAR1	CLBINPUTXBAR1	CLBINPUTXBAR1	CLBINPUTXBAR1	Enable
49	CLBINPUTXBAR2	CLBINPUTXBAR2	CLBINPUTXBAR2	CLBINPUTXBAR2	Enable
50	CLBINPUTXBAR3	CLBINPUTXBAR3	CLBINPUTXBAR3	CLBINPUTXBAR3	Enable
51	CLBINPUTXBAR4	CLBINPUTXBAR4	CLBINPUTXBAR4	CLBINPUTXBAR4	Enable
52	CLBINPUTXBAR5	CLBINPUTXBAR5	CLBINPUTXBAR5	CLBINPUTXBAR5	Enable
53	CLBINPUTXBAR6	CLBINPUTXBAR6	CLBINPUTXBAR6	CLBINPUTXBAR6	Enable
54	CLBINPUTXBAR7	CLBINPUTXBAR7	CLBINPUTXBAR7	CLBINPUTXBAR7	Enable
55	CLBINPUTXBAR8	CLBINPUTXBAR8	CLBINPUTXBAR8	CLBINPUTXBAR8	Enable
56	CLBINPUTXBAR9	CLBINPUTXBAR9	CLBINPUTXBAR9	CLBINPUTXBAR9	Enable
57	CLBINPUTXBAR10	CLBINPUTXBAR10	CLBINPUTXBAR10	CLBINPUTXBAR10	Enable
58	CLBINPUTXBAR11	CLBINPUTXBAR11	CLBINPUTXBAR11	CLBINPUTXBAR11	Enable
59	CLBINPUTXBAR12	CLBINPUTXBAR12	CLBINPUTXBAR12	CLBINPUTXBAR12	Enable
60	CLBINPUTXBAR13	CLBINPUTXBAR13	CLBINPUTXBAR13	CLBINPUTXBAR13	Enable
61	CLBINPUTXBAR14	CLBINPUTXBAR14	CLBINPUTXBAR14	CLBINPUTXBAR14	Enable
62	CLBINPUTXBAR15	CLBINPUTXBAR15	CLBINPUTXBAR15	CLBINPUTXBAR15	Enable
63	CLBINPUTXBAR16	CLBINPUTXBAR16	CLBINPUTXBAR16	CLBINPUTXBAR16	Enable

The GPREG is accessible by the CPU and the bits of this register can be used as BOUNDARY INPUTs for the CLB Tiles. For example, CLB1s GPREG[0] can be used as BOUNDARY IN0 (Cell Input 0) for the corresponding CLB Tile.

To connect multiple tiles to each other, you can use the CLBx OUT4/5 and connect to CLBy BOUNDARY INz through the CLB X-BAR and the Global Signals Mux.

Another option is to connect the CLBx OUT0-7 to a GPIO and then use the INPUT X-BAR to bring the signal back in to the device and connect to the CLBy BOUNDARY INz through the CLB X-BAR and the Global Signals Mux.

To use GPIOs as inputs to the CLB, you must utilize the Input X-BAR and the CLB X-BAR. Figure 9-5 shows how GPIOs can be used as inputs to the CLB tiles.

9.3.3 CLB Output Selection

The eight outputs of the CLB are replicated to create 32 output signals. Each of these 32 outputs has a separate enable bit defined in the CLB output enable register, CLB_OUT_EN. The CLB outputs go to the ePWM, eCAP, eQEP and the crossbar module in the device. This allows the user to enhance the functionality of these modules with the CLB. Figure 9-9 shows the CLB outputs.

The user has the capability to disable updated to the CLB_OUT_EN register by blocking access to the register through setting the CLB_MISC_ACCESS_CTRL.BLKEN bit. The eight outputs are replicated to generate a total of 32 outputs (shown in Figure 9-9). Some of these new outputs can be used for TILE to TILE connection through the CLB Global Mux inputs.

Note

The output from OUTLUT0 is connected to OUT0, OUT8, OUT16, and OUT24. While the signal is the same, each OUTy has access to a different peripheral as shown in Section 9.3.4. Likewise, OUTLUT1 is connected to OUT1, OUT9, OUT17, and OUT25, and are the same signal.

CLBx_OUT12 through CLBx_OUT15 are unregistered and asynchronous to the CLB clock.

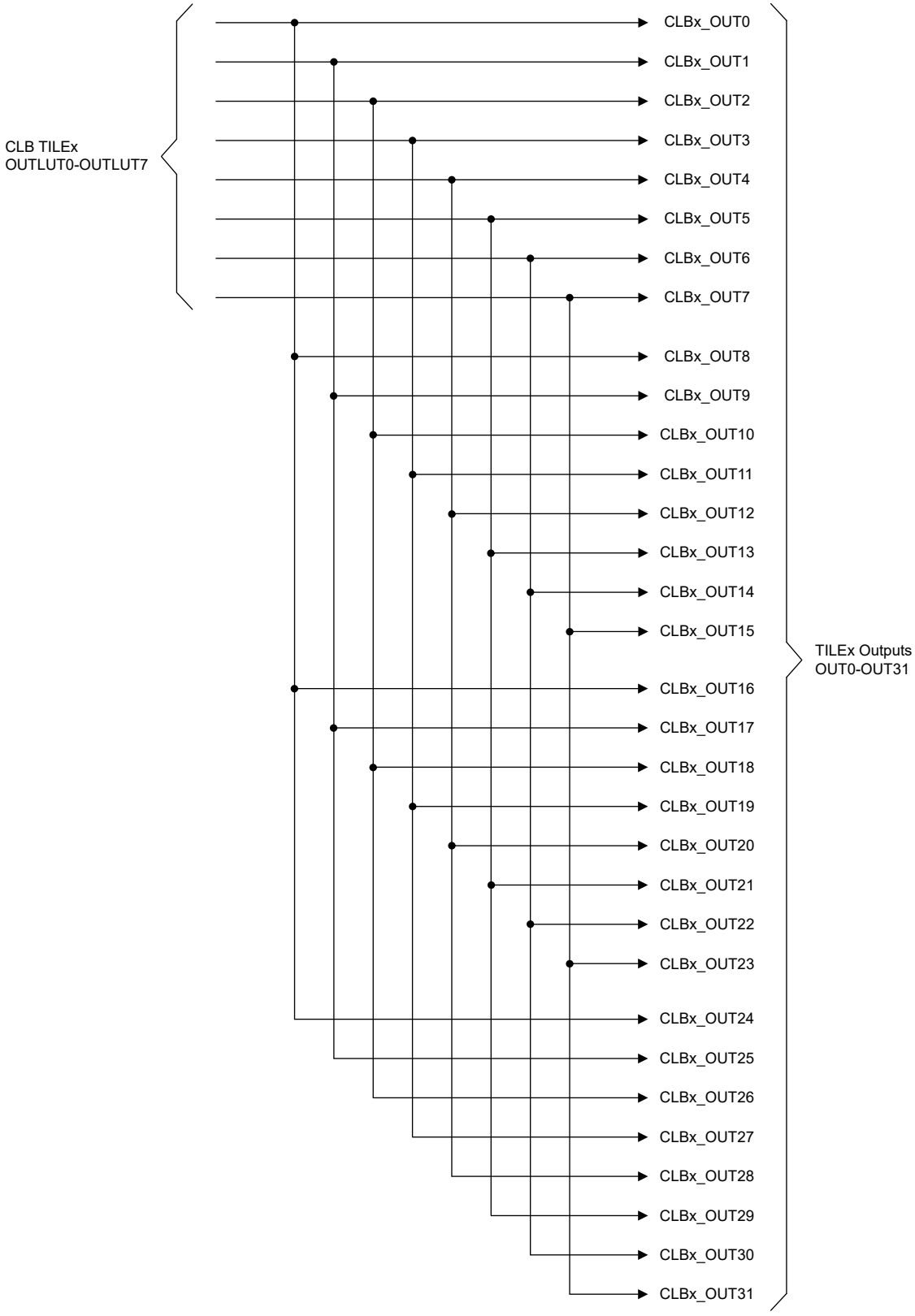


Figure 9-9. CLB Outputs

9.3.4 CLB Output Signal Multiplexer

Each CLB output signal passes through an external multiplexer that intersects a specific peripheral signal, see Figure 9-10. The output of the multiplexer is connected to the destination of the original peripheral signal and the default multiplexer setting is that the peripheral signal is passed through. The multiplexer is controlled by bit[n] in the CLB output enable register CLB_OUT_EN.

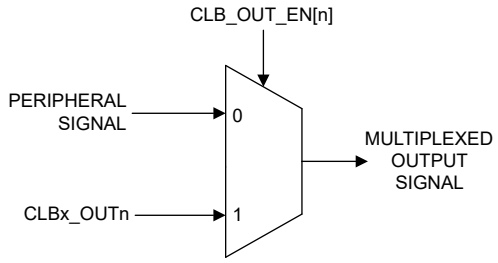


Figure 9-10. CLB Output Signal Multiplexer

For example, if the CLB1 OUT0 must override the EPWM1A signal, the OUPTUT ENABLE bit for OUT0 must be set to 1.

Table 9-6 and Table 9-7 shows the allocation of peripheral signals and the CLB outputs.

Table 9-6. CLB Output Signal Multiplexer Table

CLB Output	OUTLUT	CLB1 Destination	CLB2 Destination	CLB3 Destination	CLB4 Destination
0	OUTLUT0	EPWM1A	EPWM2A	EPWM3A	EPWM4A
1	OUTLUT1	EPWM1A_OE	EPWM2A_OE	EPWM3A_OE	EPWM4A_OE
2	OUTLUT2	EPWM1B	EPWM2B	EPWM3B	EPWM4B
3	OUTLUT3	EPWM1B_OE	EPWM2B_OE	EPWM3B_OE	EPWM4B_OE
4	OUTLUT4	EPWM1A_AQ	EPWM2A_AQ	EPWM3A_AQ	EPWM4A_AQ
5	OUTLUT5	EPWM1B_AQ	EPWM2B_AQ	EPWM3B_AQ	EPWM4B_AQ
6	OUTLUT6	EPWM1A_DB	EPWM2A_DB	EPWM3A_DB	EPWM4A_DB
7	OUTLUT7	EPWM1B_DB	EPWM2B_DB	EPWM3B_DB	EPWM4B_DB
8	OUTLUT0	EQEP1_QCLK	EQEP2_QCLK	Reserved	Reserved
9	OUTLUT1	EQEP1_QDIR	EQEP2_QDIR	Reserved	Reserved
10	OUTLUT2	Reserved	Reserved	EQEP3_QDIR	EQEP4_QDIR
11	OUTLUT3	Reserved	Reserved	EQEP3_QCLK	EQEP4_QCLK
12	OUTLUT4	XBARs	XBARs	XBARs	XBARs
13	OUTLUT5	XBARs	XBARs	XBARs	XBARs
14	OUTLUT6	ECAP Mux	ECAP Mux	ECAP Mux	ECAP Mux
15	OUTLUT7	ECAP Mux	ECAP Mux	ECAP Mux	ECAP Mux
16	OUTLUT0	Global Mux	Global Mux	Global Mux	Global Mux
17	OUTLUT1	Global Mux	Global Mux	Global Mux	Global Mux
18	OUTLUT2	Global Mux	Global Mux	Global Mux	Global Mux
19	OUTLUT3	Global Mux	Global Mux	Global Mux	Global Mux
20	OUTLUT4	Global Mux	Global Mux	Global Mux	Global Mux
21	OUTLUT5	Global Mux, SPIA_STE_OUT	Global Mux, SPIB_STE_OUT	Global Mux, SPIC_STE_OUT	Global Mux, SPID_STE_OUT

Table 9-6. CLB Output Signal Multiplexer Table (continued)

CLB Output	OUTLUT	CLB1 Destination	CLB2 Destination	CLB3 Destination	CLB4 Destination
22	OUTLUT6	Global Mux, SPIA_SIMO_OUT	Global Mux, SPIB_SIMO_OUT	Global Mux, SPIC_SIMO_OUT	Global Mux, SPID_SIMO_OUT
23	OUTLUT7	Global Mux, SPIA_SOMI_OUT	Global Mux, SPIB_SOMI_OUT	Global Mux, SPIC_SOMI_OUT	Global Mux, SPID_SOMI_OUT
24	OUTLUT0	SPIA_CLK_IN	SPIB_CLK_IN	SPIC_CLK_IN	SPID_CLK_IN
25	OUTLUT1	SPIA_SIMO_IN	SPIB_SIMO_IN	SPIC_SIMO_IN	SPID_SIMO_IN
26	OUTLUT2	SPIA_STE_IN	SPIB_STE_IN	SPIC_STE_IN	SPID_STE_IN
27	OUTLUT3	SCIA_RX	SCIB_RX	SCIC_RX	SCID_RX
28	OUTLUT4	ECAP1_OUT_EN	ECAP2_OUT_EN	ECAP3_OUT_EN	ECAP4_OUT_EN
29	OUTLUT5	ECAP1_OUT	ECAP2_OUT	ECAP3_OUT	ECAP4_OUT
30	OUTLUT6	FSITX Triggers	FSITX Triggers	FSITX Triggers	FSITX Triggers
31	OUTLUT7	FSITX Triggers	FSITX Triggers	FSITX Triggers	FSITX Triggers

Table 9-7. CLB Output Signal Multiplexer Table

CLB Output	OUTLUT	CLB5 Destination	CLB6 Destination	CLB7 Destination	CLB8 Destination
0	OUTLUT0	EPWM5A	EPWM6A	EPWM7A	EPWM8A
1	OUTLUT1	EPWM5A_OE	EPWM6A_OE	EPWM7A_OE	EPWM8A_OE
2	OUTLUT2	EPWM5B	EPWM6B	EPWM7B	EPWM8B
3	OUTLUT3	EPWM5B_OE	EPWM6B_OE	EPWM7B_OE	EPWM8B_OE
4	OUTLUT4	EPWM5A_AQ	EPWM6A_AQ	EPWM7A_AQ	EPWM8A_AQ
5	OUTLUT5	EPWM5B_AQ	EPWM6B_AQ	EPWM7B_AQ	EPWM8B_AQ
6	OUTLUT6	EPWM5A_DB	EPWM6A_DB	EPWM7A_DB	EPWM8A_DB
7	OUTLUT7	EPWM5B_DB	EPWM6B_DB	EPWM7B_DB	EPWM8B_DB
8	OUTLUT0	Reserved	Reserved	Reserved	Reserved
9	OUTLUT1	Reserved	Reserved	Reserved	Reserved
10	OUTLUT2	Reserved	Reserved	Reserved	Reserved
11	OUTLUT3	Reserved	Reserved	Reserved	Reserved
12	OUTLUT4	XBARs	XBARs	XBARs	XBARs
13	OUTLUT5	XBARs	XBARs	XBARs	XBARs
14	OUTLUT6	ECAP Mux	ECAP Mux	ECAP Mux	ECAP Mux
15	OUTLUT7	ECAP Mux	ECAP Mux	ECAP Mux	ECAP Mux
16	OUTLUT0	Global Mux	Global Mux	Global Mux	Global Mux
17	OUTLUT1	Global Mux	Global Mux	Global Mux	Global Mux
18	OUTLUT2	Global Mux	Global Mux	Global Mux	Global Mux
19	OUTLUT3	Global Mux	Global Mux	Global Mux	Global Mux
20	OUTLUT4	Global Mux	Global Mux	Global Mux	Global Mux
21	OUTLUT5	Global Mux	Global Mux	Global Mux	Global Mux
22	OUTLUT6	Global Mux	Global Mux	Global Mux	Global Mux
23	OUTLUT7	Global Mux	Global Mux	Global Mux	Global Mux

Table 9-7. CLB Output Signal Multiplexer Table (continued)

CLB Output	OUTLUT	CLB5 Destination	CLB6 Destination	CLB7 Destination	CLB8 Destination
24	OUTLUT0	Reserved	Reserved	Reserved	Reserved
25	OUTLUT1	FSITXA_SEL_TDM_PATH	FSITXB_SEL_TDM_PATH	Reserved	Reserved
26	OUTLUT2	Reserved	Reserved	Reserved	Reserved
27	OUTLUT3	Reserved	Reserved	Reserved	Reserved
28	OUTLUT4	ECAP5_OUT_EN	ECAP6_OUT_EN	ECAP7_OUT_EN	Reserved
29	OUTLUT5	ECAP5_OUT	ECAP6_OUT	ECAP7_OUT	Reserved
30	OUTLUT6	FSITX Triggers	FSITX Triggers	Reserved	Reserved
31	OUTLUT7	FSITX Triggers	FSITX Triggers	Reserved	Reserved

9.4 CLB Tile

The purpose of the CLB tile is to provide the logic reconfiguration capability of the CLB. The CLB tile contains the following submodules:

- **Counter:** The counter submodule can be configured as an adder, a counter, or a shifter. When functioning as an adder, the counter submodule can either add or subtract. When functioning as a counter, the counter submodule can count up or count down. When functioning as a shifter, the counter submodule can shift left or shift right. The counter event inputs, as well as the reset input, can be freely connected to any of the other submodules in the same tile. Starting with CLB Type 2, the counter module can also operate as a serializer or linear feedback shift register. There are three counters in each tile.
- **LUT4:** The LUT4 submodule has a 4-input look-up table functionality and is capable of realizing any combinatorial Boolean equation of up to four inputs. There are three LUT4 submodules in each CLB tile.
- **FSM:** The Finite State Machine (FSM) submodule can be configured either as a single four-state finite state machine, or as two independent two-state finite state machines. The FSM accepts two external inputs, and generates two state outputs and one combination output. When not used as a state machine, the FSM submodule can accept two external inputs and function as a 4-input LUT. There are three FSM submodules in each CLB tile.
- **Output LUT:** The output LUT is a 3-input lookup table submodule capable of realizing any combinatorial Boolean equation of up to three inputs. There are eight such blocks in a CLB tile, each associated with one of the tile outputs.
- **Asynchronous Output Conditioning Block:** The primary purpose of the Asynchronous Output Conditioning (AOC) block is to provide asynchronous conditioning capabilities on the TILE outputs or directly on the inputs of the TILE.
- **High Level Controller:** The High Level Controller (HLC) submodule is an event-driven block that can handle up to four concurrent events. The event can be an activity on any of the other block outputs. A predefined set of operations is executed when each event occurs. The HLC also provides a data exchange and interrupt mechanism to the CPU subsystem. There are four working registers (R0, R1, R2, and R3) that can be used for basic operations, and to modify or set up values for the three counter blocks. Unlike the other submodules, there is only one HLC in each CLB tile.
- **Static Switch Block:** The static switch block provides dynamic connectivity between all of the blocks listed above. Submodules can be connected by the user, with the only restriction that the submodules must not form a combinational loop within the tile.

A CLB tile consists of three sets each of the counter block, FSM, and LUT4, one high-level controller, and eight output LUT blocks. The submodule numbering is shown in Figure 9-11.

The functionality of the LUT submodules is configured using a register field containing the binary pattern of the output of the desired look-up table. For example, a 4-input LUT has 16 possible input permutations, each of which corresponds to a desired binary 0 or 1 at the output. The register field can, therefore, be 16-bits in length, with each bit representing the desired result of a binary pattern. Input pattern sequences start at 0000 and continue sequentially to 1111. A similar method is used to encode the 16-bit state equations in the FSM submodule.

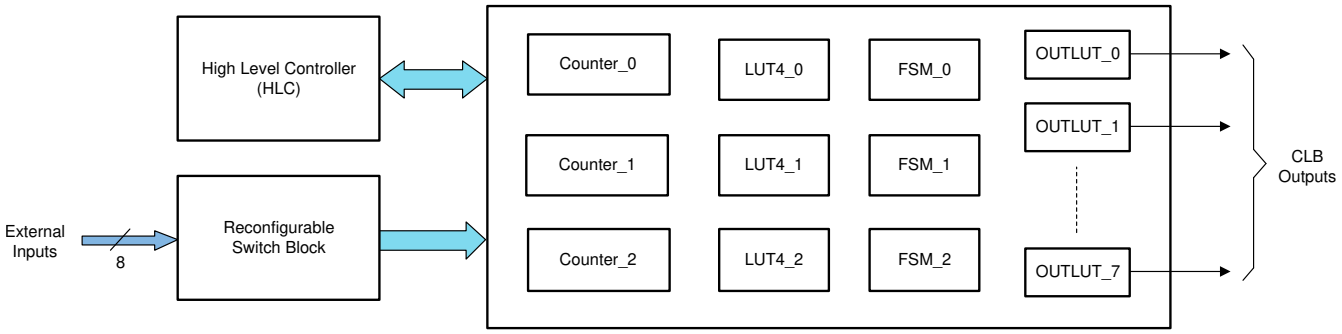


Figure 9-11. CLB Tile Submodules

9.4.1 Static Switch Block

The Static Switch Block provides the configurable connectivity between the submodules in the CLB tile. The outputs of all the submodules and the eight external inputs are connected to a common internal bus inside the tile. Every input port has a 32-to-1 multiplexer and an associated 5-bit selection value that allows the user to select one of the inputs on the bus. The only restrictions are certain signals (described below) that are tied off in the design to prevent creation of accidental combinatorial loops.

Table 9-8. Output Table

Bit Position	Signal Connection	Comment
0	Always 0	This select value is used to tie an input to 0.
1	COUNTER_0 MATCH2	
2	COUNTER_0 ZERO	
3	COUNTER_0 MATCH1	
4	FSM_0 STATE_BIT_0	
5	FSM_0 STATE_BIT_1	
6	FSM_0 LUT output	
7	LUT4_0 output	
8	Always 1	This select value is used to tie an input to 1.
9	COUNTER_1 MATCH2	
10	COUNTER_1 ZERO	
11	COUNTER_1 MATCH1	
12	FSM_1 STATE_BIT_0	
13	FSM_1 STATE_BIT_1	
14	FSM_1 LUT output	
15	LUT4_1 output	
16	Always ‘0’	
17	COUNTER_2 MATCH2	
18	COUNTER_2 ZERO	
19	COUNTER_2 MATCH1	
20	FSM_2 STATE_BIT_0	
21	FSM_2 STATE_BIT_1	
22	FSM_2 LUT output	
23	LUT4_2 output	
24	External Input 0	
25	External Input 1	
26	External Input 2	
27	External Input 3	
28	External Input 4	
29	External Input 5	
30	External Input 6	
31	External Input 7	

Table 9-9. Input Table

Module Name	Port Name	Description
Counter Block	RESET	Acts as an active high reset when used as a counter
	MODE_0	Acts as an enable when used as a counter. The counter counts only when this input is 1.
	MODE_1	Acts as a direction control when used as a counter. If this input is 1, then the counter counts up; else, the counter counts down.
LUT	IN0	Input 0 of the 4-input LUT.
	IN1	Input 1 of the 4-input LUT.
	IN2	Input 2 of the 4-input LUT.
	IN3	Input 3 of the 4-input LUT.
FSM	EXT_IN0	Input 0 of the FSM block.
	EXT_IN1	Input 1 of the FSM block.
	EXTRA_EXT_IN0	Extra external input 0 of the FSM block. This input matters only if configured in the LUT mode.
	EXTRA_EXT_IN1	Extra external input 1 of the FSM block. This input matters only if configured in the LUT mode.

The static switch block allows the user to define the input connection of any submodule to come from any of the outputs in Table 9-8. It is therefore easy to create a combinatorial loop. To prevent this, certain paths are broken in the input path of each submodule. These port positions are tied to 0, as shown in Table 9-10.

Table 9-10. Ports Tied Off to Prevent Combinatorial Loops

Module Name	Ports of Input MUX Tied Off to 0 to Prevent Combinational Loops
LUT_0	LUT_0 , LUT_1, and LUT_2 output, FSM_0, FSM_1, and FSM_2 output.
FSM_0	LUT_1 and LUT_2 output, FSM_0, FSM_1, and FSM_2 output.
LUT_1	LUT_1 and LUT_2 output, FSM_1 and FSM_2 output.
FSM_1	LUT_2 output, FSM_1 and FSM_2 output.
LUT_2	LUT_2 output, FSM_2 output.
FSM_2	FSM_2 output.

9.4.2 Counter Block

9.4.2.1 Counter Description

The counter block is a complex functional submodule that can be configured either as a counter, an adder, or a shifter. Apart from the normal operational control, this block has a dedicated EVENT input, which can trigger an addition, subtraction or shift operation, or load data into the counter register. The inputs to the counter submodule are shown in Figure 9-12.

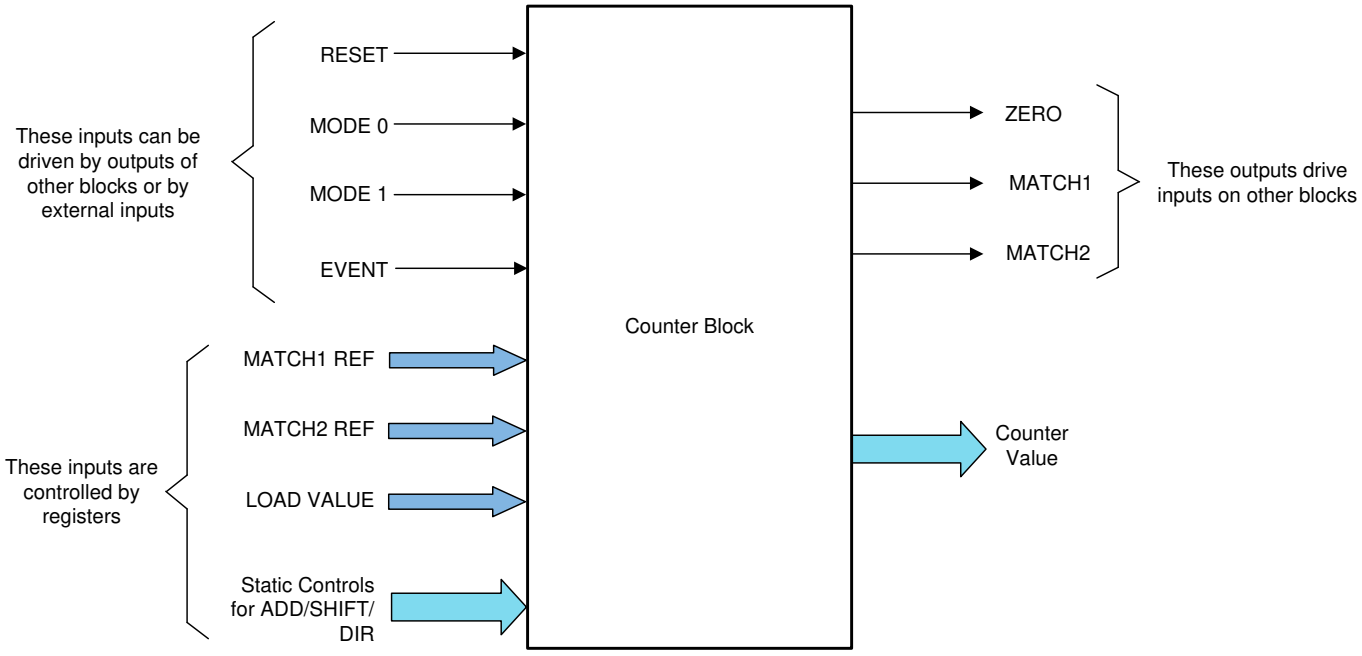


Figure 9-12. Counter Block

9.4.2.2 Counter Operation

At the heart of the counter block is a 32-bit count register. This register can either be loaded statically before counting commences, or dynamically at run time. The operation of the counter submodule is determined by the inputs described below. Note that each of the inputs can be connected to the outputs of any of the other blocks in the CLB tile. These connections are made by configuring the static switch block.

The counter inputs are:

- **RESET:** This is the highest priority input and takes precedence over all other inputs. The input is level sensitive and as long as the input remains high, the counter resets to 0 on the next clock cycle.
- **MODE_0:** This input is an enable for the counter. The counter begins counting (up or down depending on the MODE_1 setting) only when this input is high. If this input is low, then no counting takes place.
- **MODE_1:** This input is the direction control for the counter. If this input is high, then the counter increments on every clock cycle where MODE_0 is high. If this input is low, then the counter decrements on every clock cycle where MODE_0 is high. The counter wraps around to 0x00000000 after 0xFFFFFFFF when counting up. The counter wraps around to 0xFFFFFFFF after 0x00000000 when counting down. The only exception to this is when an EVENT occurs at exactly the same time, causing a different value to be loaded into the counter.

- **EVENT:** This input is defined for the purpose of triggering actions in the counter based on certain events. The event itself can be any of the outputs of the other blocks or an external input to the tile. The counter's static control inputs define the behavior of the counter on an active event. An active event is defined as a rising edge on the EVENT input. The counter can be configured to perform one of the following actions:
  - Load a predefined 32-bit value from the LOAD VALUE register into the count register
  - Shift the contents of the counter register left or right by a predefined amount between 0 and 31
  - Add or subtract a predefined 32-bit value. Addition and subtraction are treated as 32-bit unsigned operations and there is no saturation.

  Note that the effect of a rising edge on the EVENT input only lasts for one cycle. On the next cycle, the counter operation continues based on the MODE_0, MODE_1, and RESET inputs.

MATCH1 REF and MATCH2 REF are 32-bit reference values that are used to generate the MATCH1 and MATCH2 outputs. The MATCH1 output becomes active high whenever the counter register value matches the 32-bit MATCH1 REF value. MATCH2 behaves in a similar manner in relation to the MATCH2 REF register. The reference values for MATCH1 and MATCH2 can either be setup once before the start of operation, or can be modified dynamically. The High Level Controller can load desired values into the MATCH1 REF and MATCH2 REF registers.

Note that the counter load and match registers are not memory-mapped. For more information, see Section 9.5.2.

The three logic outputs of the counter block are:

- **ZERO:** This output goes high whenever the counter register is 0.
- **MATCH1:** This output goes high whenever the counter register is equal to the MATCH1 REF input register.
- **MATCH2:** This output goes high whenever the counter register is equal to the MATCH2 REF input register.

The operation of the counter block is controlled by the CFG_MISC_CTRL register. The following three bits of this register are relevant for each counter. The "x" below refers to the counter instance; 0, 1, or 2. For more information, see the CLB_MISC_CONTROL register description located in Section 9.10.

- COUNT_EVENT_CTRL_x: This bit defines whether the counter performs an addition or a shift on an event. A value of 0 means that on an event, the counter loads the static value; 1 means an add/shift operation is performed. This bit must be 0 for indirect loads and HLC loads of the counter to take effect.
- COUNT_ADD_SHIFT_x. 1 means add, 0 means shift.
- COUNT_DIR_x. 1 means left shift or add. 0 means right shift or subtract.

Table 9-11 shows the logical operation of the counter block in terms of the inputs and control register bits. Count up and down modes are the normal operation with EVENT = 0. The operations on the CNTVAL register are:

Load: CNTVAL = EVENT_LOAD_VAL

Shift right: CNTVAL = CNTVAL >> EVENT_LOAD_VAL

Shift left: CNTVAL = CNTVAL << EVENT_LOAD_VAL

Subtract: CNTVAL = CNTVAL - EVENT_LOAD_VAL

Add: CNTVAL = CNTVAL + EVENT_LOAD_VAL

Table 9-11. Counter Block Operating Modes

EVENT	MODE_0	MODE_1	COUNT_EVENT_CTRL_x	COUNT_ADD_SHIFT_x	COUNT_DIR_x	Action on CNTVAL
0	0	0	X	X	X	None
0	0	1	X	X	X	None
0	1	0	X	X	X	Count down
0	1	1	X	X	X	Count up
1	X	X	0	X	X	Load
1	X	X	1	0	0	Shift right
1	X	X	1	0	1	Shift left
1	X	X	1	1	0	Subtract
1	X	X	1	1	1	Add

9.4.2.3 Serializer Mode

Starting with CLB Type 2, the Counter module can operate as a serializer. In this mode of operation, this module acts as a shift register (also referred to as a serializer). In serializer mode of operation, the EVENT input is used to shift one bit of data into the serializer. Either of MATCH1 and MATCH2 can be configured to send out the shift register data.Using the MATCH1/2_TAP_SEL bit of CLB_COUNT_MATCH_TAP_SEL, any bit position of the counter can be brought out on the MATCH1/MATCH2 outputs. The shifting and direction of the counter in this mode is controlled by MODE_0 (enable) and MODE_1 (direction).

To enable the Serializer mode, CLB_MISC_CONTROL.COUNT_SERIALIZER_0 (for Counter 0) must be set.

9.4.2.4 Linear Feedback Shift Register (LFSR) Mode

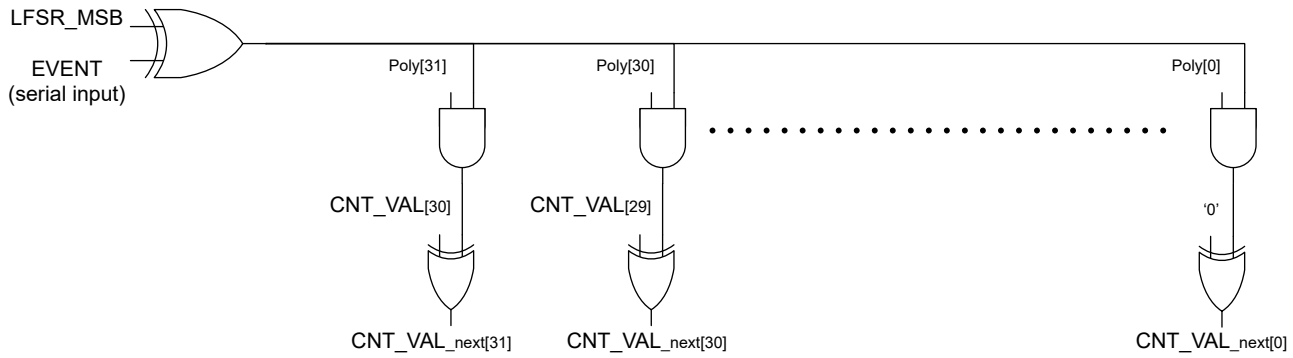
Starting with CLB Type 2, the Counter module operates as a linear feedback shift register. By configuring the characteristics of the LFSR, the counter module is used to compute the CRC on a serial bit stream. The polynomial for LFSR is in the MATCH2 reference register. The feedback bit position is in the MATCH1 reference register.

To enable the LFSR mode, CLB_MISC_CONTROL.COUNT_SERIALIZER_0 (for Counter 0) must be set along with COUNT0_LFSR_EN.

There are two types of LSFR that can be selected by changing the MODE1 value (0 or 1), as shown in Figure 9-13.

Structure for LFSR Type 1 (MODE_1 = 0)

CNT_VAL is the 32-bit counter's active register
CNT_VAL_next is the 32-bit value to be written into CNT_VAL on the next active cycle (clock edge when MODE_0 == 1)
LFSR_MSB = CNT_VAL[MATCH1_REF[4:0]]
Poly[31:0] is MATCH2_REF which acts as the CRC polynomial



Structure for LFSR Type 2 (MODE_1 = 1)

CNT_VAL is the 32-bit counter register
CNT_VAL_next is the 32-bit value to be written into CNT_VAL on the next active cycle (clock edge when MODE_0 == 1)
LFSR_MSB = CNT_VAL[MATCH1_REF[4:0]]
Poly[31:0] is MATCH2_REF which acts as the CRC polynomial

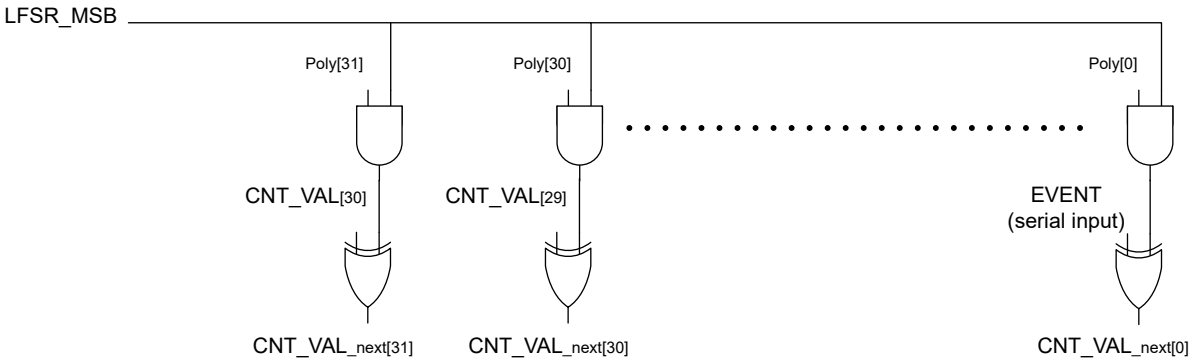


Figure 9-13. LFSR Modes

9.4.3 FSM Block

The Finite State Machine (FSM) block provides the ability to build programmable finite state machines with up to four states. The FSM block has two register bits and two external inputs, and can be programmed either as two 2-state machines or as a single 4-state machine. For additional flexibility, there are two auxiliary inputs (EXTRA_EXT_IN0 and EXTRA_EXT_IN1) that can be used to create larger combinational functions by giving up a state functionality. The structure of the FSM is shown in Figure 9-14.

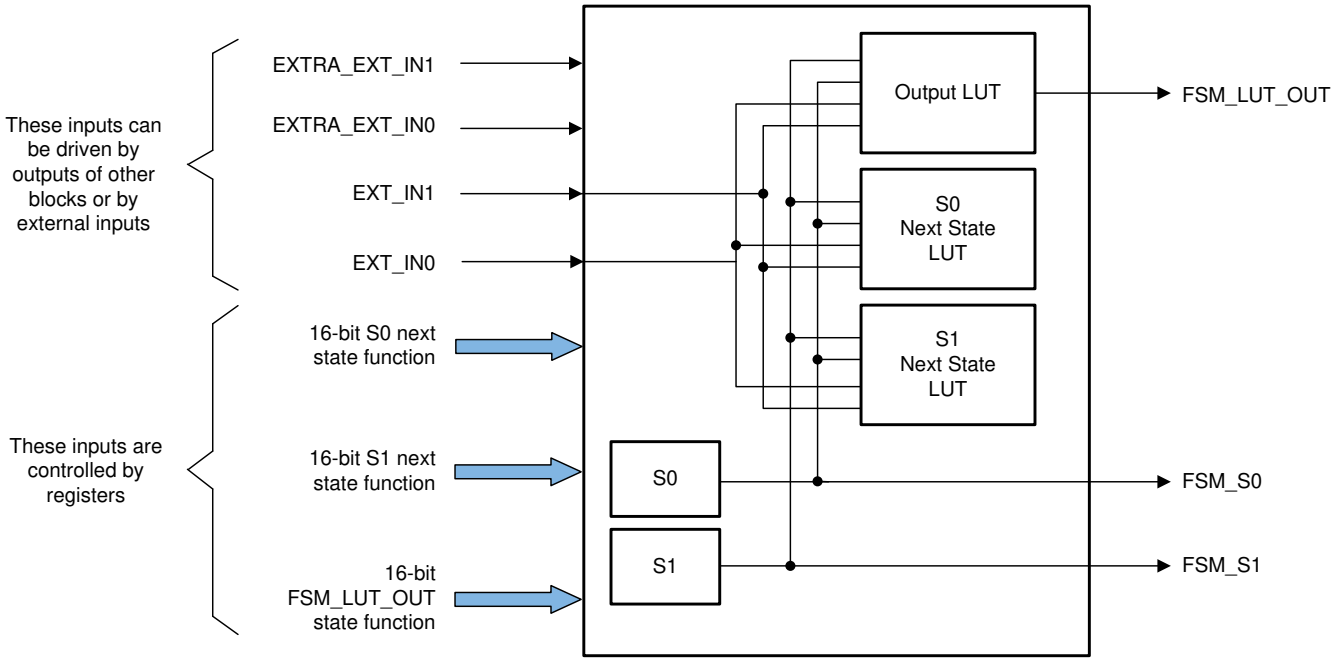


Figure 9-14. FSM Block

The signals and functionality of the FSM block are:

- **EXT_IN0 and EXT_IN1:** These are the two external inputs that can be used to control the output FSM_LUT_OUT or either of the states S0 and S1.
- **S0** and **S1** are two state bits that have independent state control equations.
- **16-bit S0 equation** defines a function (EXT_IN1, EXT_IN0, S1, S0). The four bits in the order defined are used as an index into the 16-bit register to decide the next state of S0.
- **16-bit S1 equation** defines a function (EXT_IN1, EXT_IN0, S1, S0). The four bits in the order defined are used as an index into the 16-bit register to decide the next state of S1.
- **16-bit output equation** defines a function (EXT_IN1, EXT_IN0, S1, S0). The four bits in the order defined are used as an index into the 16-bit register to decide the output value of FSM_LUT_OUT. An additional level of configurability is provided such that FSM_LUT_OUT can use extra inputs in case the states S0 and S1 are unused.

  One extra bit is used to select EXTRA_EXT_IN0 instead of S0. One extra bit is used to select EXTRA_EXT_IN1 instead of S1. Using these, one can effectively build 3-input or a 4-input LUT for the FSM_LUT_OUT by giving up one or two state bits, respectively.

The CFG_MISC_CTRL register controls the operation of the FSM block. Two bits in this register are used for each FSM Block to determine whether the FSM output LUT function uses the state variable S0/S1, or the corresponding extra external input signal FSM_EXTRA_EXT_INx. A 0 means use the state bit, and a 1 means use the FSM_EXTRA_EXT_IN0 / FSM_EXTRA_EXT_IN1 signal.

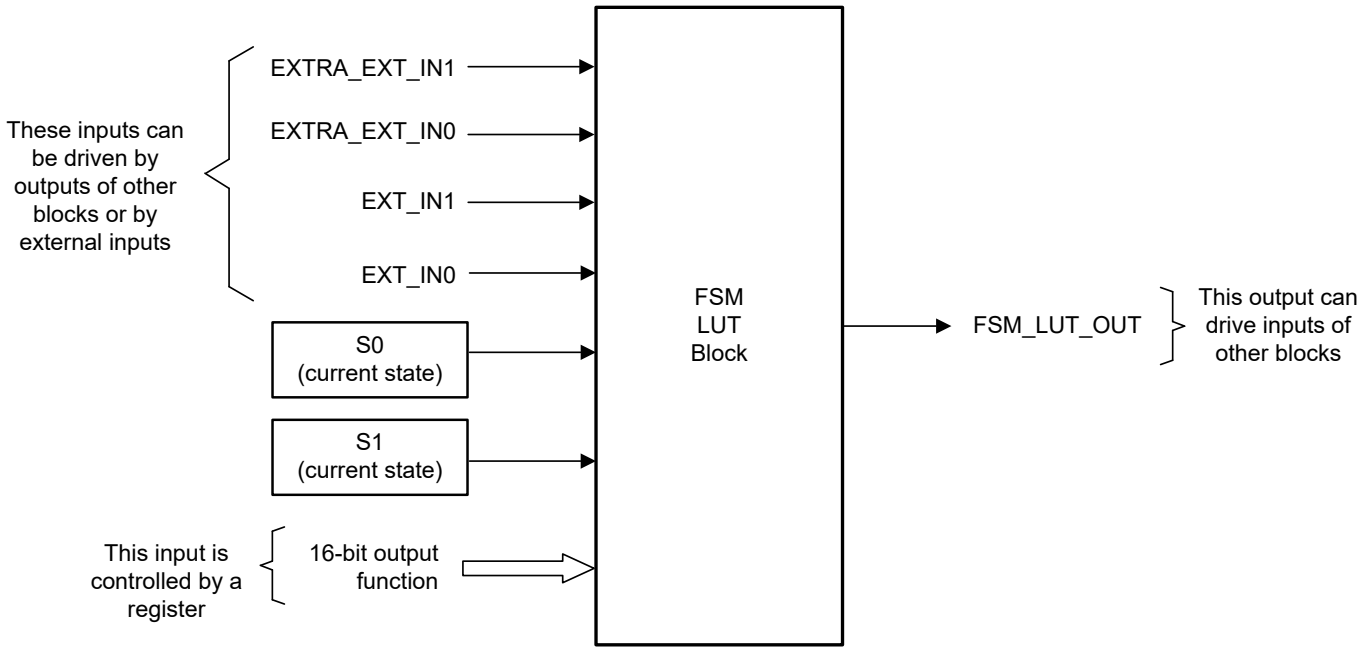


Figure 9-15. FSM LUT Block

9.4.4 LUT4 Block

This is a simple four input Look-Up table (LUT) block with inputs IN0, IN1, IN2, and IN3 (see Figure 9-16). Any combinatorial Boolean equation using the four inputs can be realized by programming the 16-bit control register associated with each LUT4 block. For more information, see the LUT4 register descriptions located in Section 9.10.

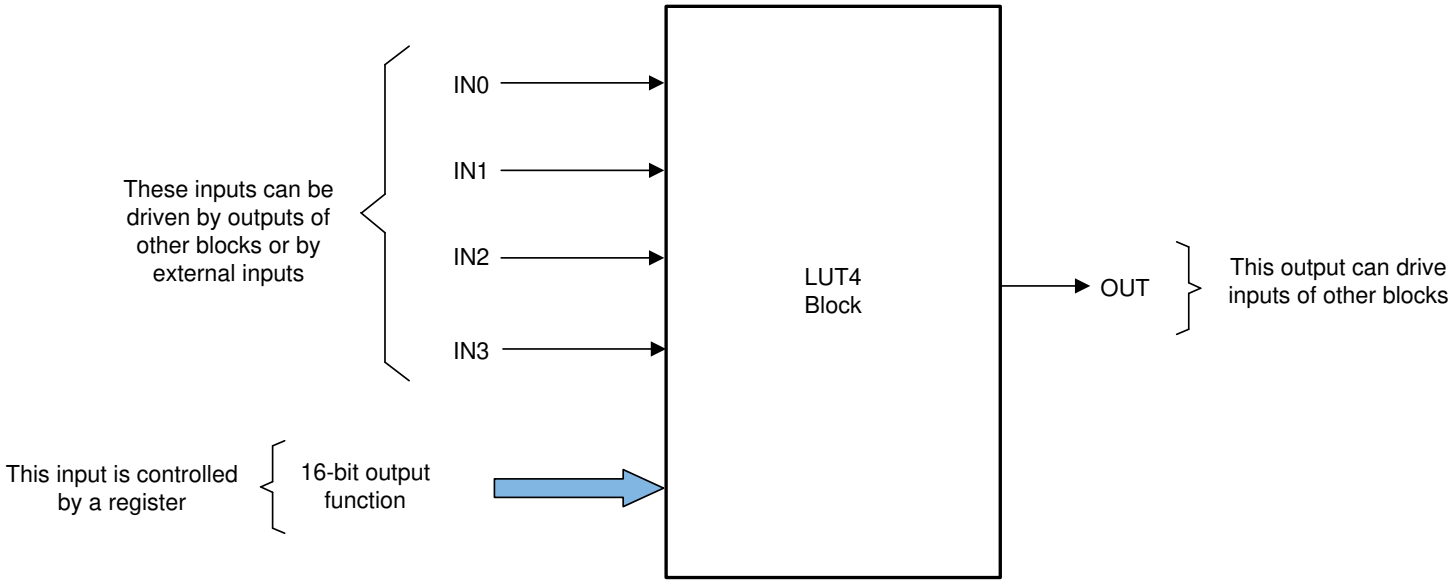


Figure 9-16. LUT4 Block

9.4.5 Output LUT Block

The output LUT block (Figure 9-17) is very similar in functionality to the LUT4 block, except that the output LUT block has three inputs. Unlike the other sub blocks, the outputs of these blocks are meant to go out of the tile and hence cannot be used by any other block within the tile. Any combinatorial function of the three inputs can be realized by the output LUT block. For more information, see the output LUT register descriptions located in Section 9.10.

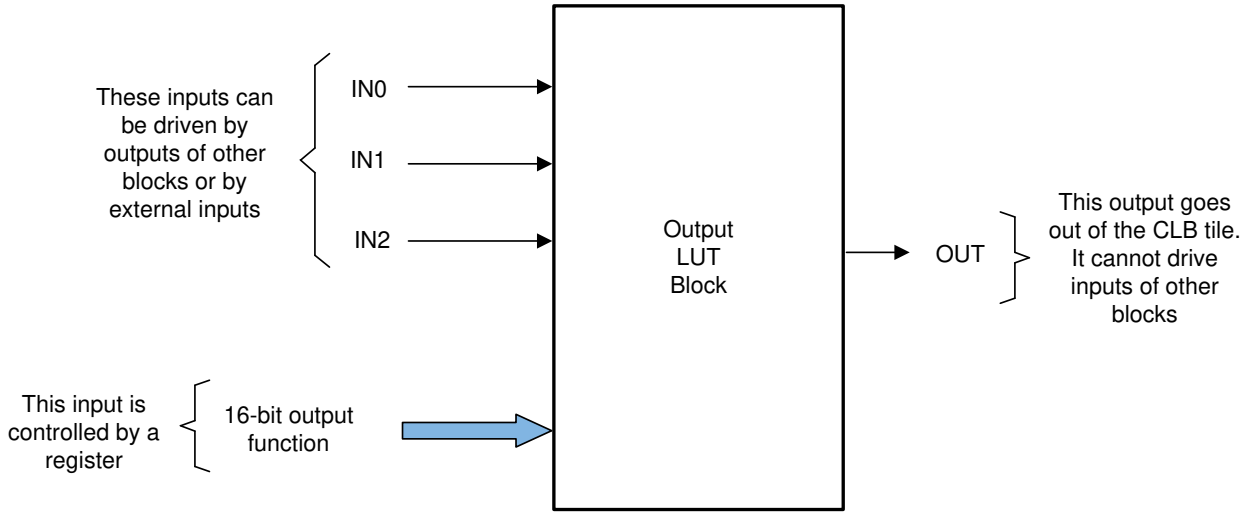


Figure 9-17. Output LUT Block

9.4.6 Asynchronous Output Conditioning (AOC) Block

The logic in the AOC block is organized into three stages with the inputs passing through different types of logic modification at each stage before proceeding to the next level. This is shown in Figure 9-18.

There are 8 inputs to this block. Each of these 8 inputs can pick the corresponding BOUNDARY input to the CLB or the CLB TILE output (for example, the INPUT 0 of the AOC block can choose between CLB BOUNDARY INPUT0 and CLB TILE OUTPUT 0). If the CLB TILE OUTPUT 0 is selected, the CLB TILE OUTPUT 0 is always registered before being sent to the subsequent asynchronous signal conditioning stages. In each of the three stages, there is always an option to do nothing and just send the signal as is to the next stage (bypass).

Stage 1: The input signal can be inverted before sending the signal to the next stage.

Stage 2: The signal coming from Stage 1 can be GATED with a gating control signal. The gating control signal can either be from a software register or can be any of the CLB TILE outputs. The GATING function can be a logical AND, OR, or XOR.

Stage 3: The input signal can be used to either set or clear the output on the rising edge of the signal. This is a purely asynchronous set/clear that occurs without needing any clocks. The release control signal, when high, restores the output to the default state (HIGH if asynchronous clear is selected and LOW if asynchronous set is selected). The release control signal can be either from a software register or can be any of the CLB TILE outputs. Optionally, instead of any of the asynchronous set and clear operations, the input signal can just be delayed by a clock cycle.

The interaction of the CLB TILE and the AOC block is shown in Figure 9-19.

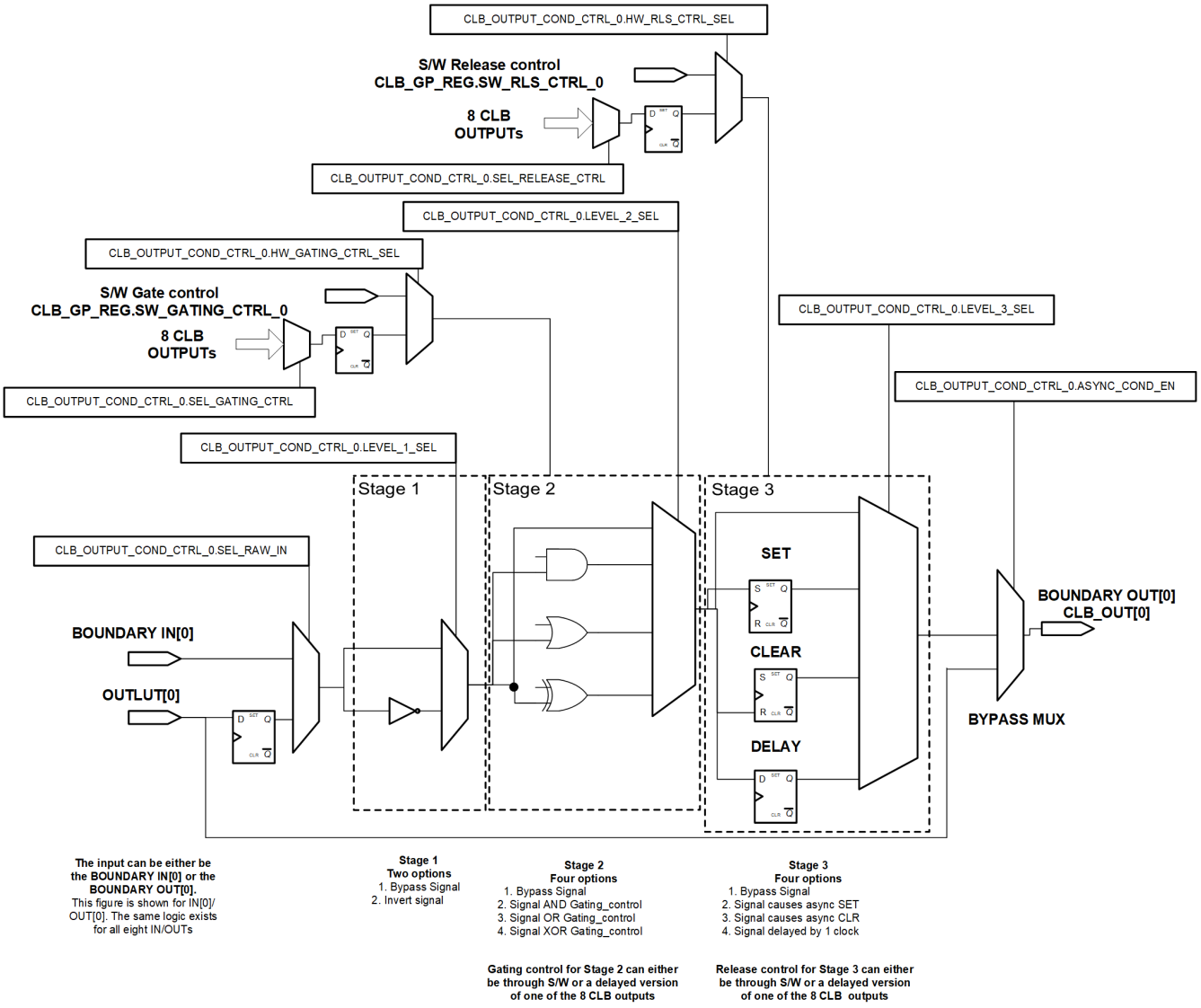


Figure 9-18. AOC Block

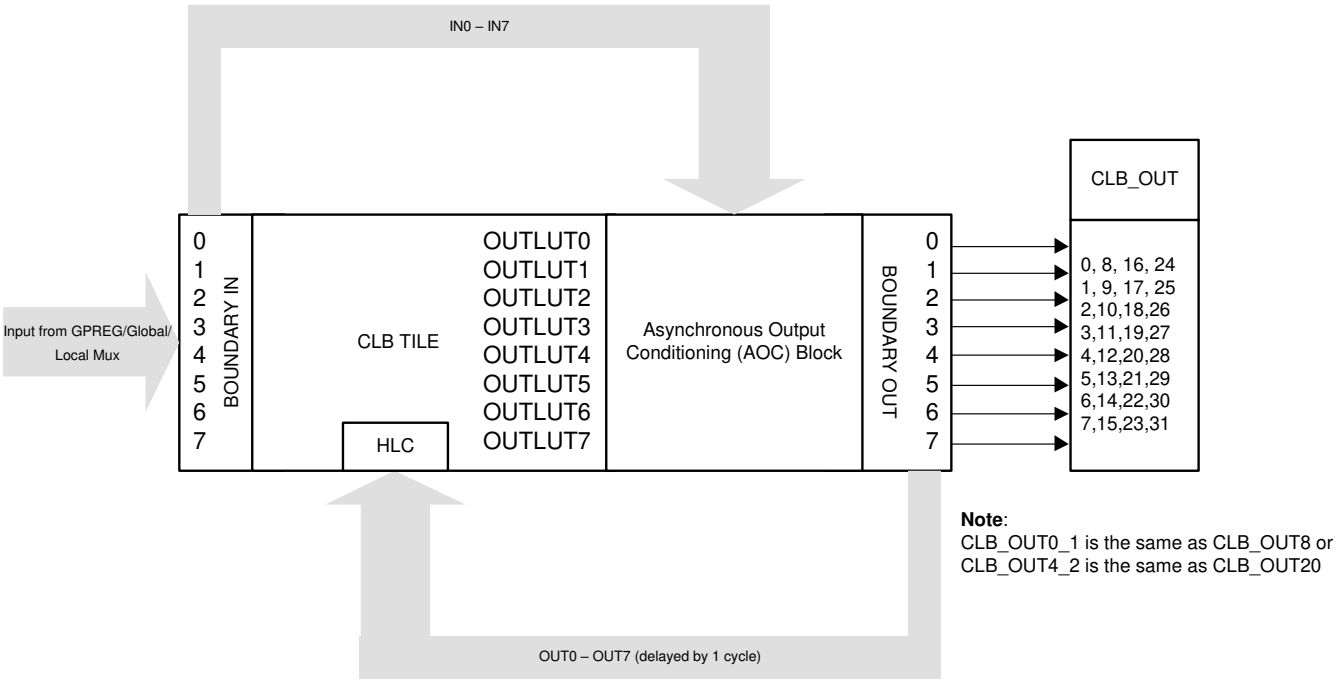


Figure 9-19. AOC Block and The CLB TILE

Note

Only CLB_OUT12 to CLB_OUT15 can be used as ASYNC outputs. This is the same as CLB_OUT4_1 to CLB_OUT7_1. GPIO Output XBAR can be used to route the OUT4_1 and OUT5_1 ASYNC outputs to GPIOs.

9.4.7 High Level Controller (HLC)

The High Level Controller (HLC) is significantly more complex than the other blocks in the CLB tile. The HLC performs two main functions:

- Provides a means of communication and data exchange with the rest of the device. This is done through two methods: a global access path to four general purpose HLC registers (R0 through R3) and PUSH and PULL FIFOs between the CPU and the HLC. The general-purpose HLC registers are designed to only be written to during device configuration time. To avoid unexpected behavior, the HLC registers must not be written to during run-time operation. The PUSH and PULL FIFOs are the primary avenues of data exchange during run-time, refer to Section 9.4.7.4 for more information.
- Provides a programmable, event-based action system, which performs computation, manipulation of logic functionality, and data movement. In other words, events can be configured to trigger a predefined set of actions in the CLB tile, or to initiate data exchange with the rest of the device.

The architecture of the HLC is shown in Figure 9-20. The HLC is an event-based system capable of handling up to four simultaneous events that can be selected from any outputs of the other blocks within the tile or from an external input.

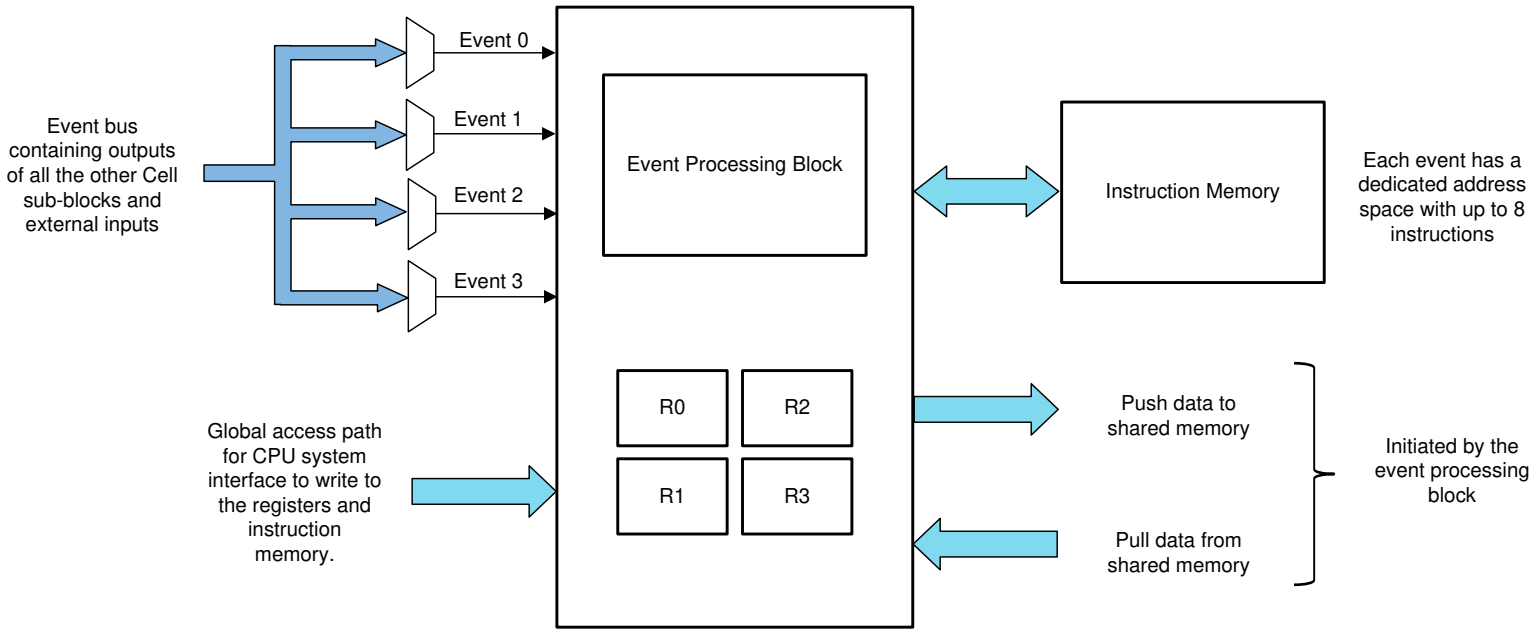


Figure 9-20. High Level Controller Block

9.4.7.1 High Level Controller Events

Each of the four HLC events has a dedicated address from which instructions are executed. Events are selected from the set of signals listed in Table 9-12. The lowest numbered event (Event 0) has the highest priority, and the highest numbered event (Event 3) has the lowest priority.

Table 9-12. HLC Event List

Index	HLC Event Mux
0	Always 0
1	COUNTER_0 MATCH2
2	COUNTER_0 ZERO
3	COUNTER_0 MATCH1
4	FSM_0 STATE_BIT_0
5	FSM_0 STATE_BIT_1
6	FSM_0 LUT output
7	LUT4_0 output
8	Always 1
9	COUNTER_1 MATCH2
10	COUNTER_1 ZERO
11	COUNTER_1 MATCH1
12	FSM_1 STATE_BIT_0
13	FSM_1 STATE_BIT_1
14	FSM_1 LUT output
15	LUT4_1 output
16	Always ‘0’
17	COUNTER_2 MATCH2
18	COUNTER_2 ZERO
19	COUNTER_2 MATCH1
20	FSM_2 STATE_BIT_0
21	FSM_2 STATE_BIT_1
22	FSM_2 LUT output
23	LUT4_2 output
24	External Input 0
25	External Input 1
26	External Input 2
27	External Input 3
28	External Input 4
29	External Input 5
30	External Input 6
31	External Input 7

Additional HLC inputs are available starting with Type 2 CLB and later. These inputs can be selected by choosing the alternate MUX options for the HLC module (HLC_ALT_MUX_SEL_n = 1) shown in Table 9-13.

Table 9-13. HLC ALT Event List

Index	HLC Event Mux
0	CLB_OUT_0
1	CLB_OUT_1
2	CLB_OUT_2
3	CLB_OUT_3
4	CLB_OUT_4
5	CLB_OUT_5
6	CLB_OUT_6
7	CLB_OUT_7
8	CLB_OUT_0.INVERTED
9	CLB_OUT_1.INVERTED
10	CLB_OUT_2.INVERTED
11	CLB_OUT_3.INVERTED
12	CLB_OUT_4.INVERTED
13	CLB_OUT_5.INVERTED
14	CLB_OUT_6.INVERTED
15	CLB_OUT_7.INVERTED
16	CLB_ASYNC_OUT_0
17	CLB_ASYNC_OUT_1
18	CLB_ASYNC_OUT_2
19	CLB_ASYNC_OUT_3
20	CLB_ASYNC_OUT_4
21	CLB_ASYNC_OUT_5
22	CLB_ASYNC_OUT_6
23	CLB_ASYNC_OUT_7
24	CLB_ASYNC_OUT_0.INVERTED
25	CLB_ASYNC_OUT_1.INVERTED
26	CLB_ASYNC_OUT_2.INVERTED
27	CLB_ASYNC_OUT_3.INVERTED
28	CLB_ASYNC_OUT_4.INVERTED
29	CLB_ASYNC_OUT_5.INVERTED
30	CLB_ASYNC_OUT_6.INVERTED
31	CLB_ASYNC_OUT_7.INVERTED

9.4.7.2 High Level Controller Instructions

The instruction memory supports up to eight instructions per event. Each instruction sequence gets triggered on the rising edge of the corresponding event. Starting with CLB Type 2, the option to trigger the execution of instructions using both falling edge and rising edge is available.

The HLC memory supports up to eight instructions per event, starting at the beginning of the fixed address range shown in Table 9-14. An instruction sequence is triggered on the rising edge of the corresponding event. If two or more events occur simultaneously, the associated instruction sequences each are executed sequentially in priority order.

Table 9-14. HLC Instruction Address Ranges

Address	Instructions for
00000 to 00111	Event 0
01000 to 01111	Event 1
10000 to 10111	Event 2
11000 to 11111	Event 3

The HLC instruction format is shown in Table 9-15.

Table 9-15. HLC Instruction Format

Last Instruction Bit	5-Bit Opcode	3-Bit Source	3-Bit Destination
This bit when set to 1 stops execution after the current instruction.	MOV 00000 MOV_T1 00001 MOV_T2 00010 PUSH 00011 PULL 00100 ADD 00101 SUB 00110 INTR 00111	Source can be R0, R1, R2, R3, C0, C1, C2.	Destination can be R0, R1, R2, R3, C0, C1, C2. Note that for ADD/SUB instructions, only R0, R1, R2, or R3 can be the destination.

R0, R1, R2, and R3 are four 32-bit general-purpose registers in the HLC. C0, C1, and C2 are three counter registers present in the CLB tile. <Src> is used to indicate the source and <Dest> is used to indicate the destination. Table 9-16 describes the HLC instructions.

Table 9-16. HLC Instruction Description

Instruction	Description
ADD <Src>, <Dest>	This instruction performs an unsigned 32-bit addition. <Dest> = <Dest> + <Src>. The <Src> can be R0, R1, R2, R3, C0, C1, or C2. The <Dest> can only be R0, R1, R2, or R3.
INTR <6-bit constant>	This instruction flags an interrupt through the CPU interface. The 6-bit constant is stored in the interrupt flag register CLB_INTR_TAG_REG. If multiple INTR instructions are called consecutively, only the first one has an effect. When multiple INTR calls are needed, each can be separated by other HLC instructions to make sure the interrupt calls take effect. **Note** Starting with CLB Type 2, NMI can be generated by the CLB. This feature is DISABLED by default and must be enabled (**CLB_LOAD_EN.NMI_EN**).
MOV <Src>, <Dest>	This instruction moves <Src> to <Dest>. Both <Src> and <Dest> can be any of R0, R1, R2, R3, C0, C1, or C2. The COUNT_EVENT_CTRL_x bit must be configured to load (that is, 0) for indirect loads and HLC loads of the counter to take effect.

Table 9-16. HLC Instruction Description (continued)

Instruction	Description
MOV_T1 <Src>, <Dest>	This instruction moves <Src> to the Match1 register of the <Dest> counter. <Src> can be any of the registers R0, R1, R2, R3, or the counter values associated with C0, C1, or C2. <Dest> is the Match1 register of any of the counters C0, C1, or C2. Examples are: • This instruction moves the count value in C1 into register R0: MOV_T1 C1 R0 • This instruction moves the value in R2 into the Match1 register of counter C0: MOV_T1 R2 C0
MOV_T2 <Src>, <Dest>	This instruction is similar to MOV_T1. The instruction moves <Src> to the Match2 register of the <Dest> counter. <Src> can be any of the registers R0, R1, R2, R3, or the counter values associated with C0, C1, or C2. <Dest> is the Match2 register of any of the counters C0, C1, or C2.
PULL <Dest>	This instruction transfers data from the data exchange pull memory buffer in the CPU interface to the <Dest> register. <Dest> can be any of R0, R1, R2, or R3. The PULL instruction is used as seen from the High Level Controller and a PULL operation reads (pulls) data from an internal 4-word FIFO.
PUSH <Src>	This instruction transfers data from <Src> to the data exchange push memory buffer in the CPU interface. <Src> can be any of R0, R1, R2, R3, C0, C1, or C2. The PUSH instruction is used as seen from the High Level Controller and pushes data into an internal 4 word FIFO.
SUB <Src>, <Dest>	This instruction performs an unsigned 32-bit subtraction. <Dest> = <Dest> - <Src>. The <Src> can be R0, R1, R2, R3, C0, C1, or C2. The <Dest> can only be R0, R1, R2, or R3.

MOV, MOV_T1, MOV_T2, ADD, SUB, and INTR instructions take one cycle to execute. PUSH and PULL require two cycles to execute. Note that the PUSH and PULL instructions are pipeline protected, meaning that a register can be used immediately after a PUSH/PULL to that register.

For multiple events triggered simultaneously, if the last instruction in the higher priority event is a PUSH or a PULL, there is an additional cycle delay between the end of the higher priority event and the start of the next event. If the last instruction is not a PUSH or PULL, then there is no cycle delay between the events.

9.4.7.3 <Src> and <Dest>

Three bits are used to encode the <Src> and <Dest> registers as shown in Table 9-17.

Table 9-17. HLC Register Encoding

Bits	Register
000	R0
001	R1
010	R2
011	R3
100	C0
101	C1
110	C2

9.4.7.4 Operation of the PUSH and PULL Instructions (Overflow and Underflow Detection)

The PUSH and PULL operations of the HLC are intended for data exchange with the host system. There are separate FIFO buffers for PUSH and PULL operations. For example, a series of PUSH operations write to successive locations in a linearly mapped-memory buffer. The PUSH buffer and the PULL buffer are mapped at address offsets shown in the *Registers* section.

The CPU can read from and write to the PUSH and PULL buffers, respectively, to exchange data with the HLC. Data pushed by the HLC is read by the CPU from the PUSH buffers. Data sent from the CPU to the HLC is written by the CPU to the PULL buffer and is read by the HLC using the PULL instruction.

Refer to *clb_ex13_push_pull* for guidance on properly using the PUSH and PULL buffers. To make use of one of the CLB inputs as a GPREG, have this input indicate when data is written to the FIFO by the CPU.

There are separate PUSH and PULL address pointers that increment each time the HLC performs a PUSH or PULL operation. These address pointers are also memory-mapped so that the CPU can determine their value. These address pointers are also writable and can be reset by the CPU at any time.

Overflow and underflow detection is done by simply reading the values of the PUSH and PULL address pointers.

In the CLB module of the device, the depth of the PUSH and PULL FIFOs is four 32-bit words each. If the CPU starts a fresh data transfer to the PULL buffers and sees the address pointer greater than four, then an underflow has occurred since the HLC has pulled more data than the number of words written by the CPU into the buffer.

9.5 CPU Interface

9.5.1 Register Description

There are three classes of registers that are used to control and configure the CLB tile. This specification only describes the offset addresses of the registers. The absolute register addresses are different for each CLB tile. The three instances of the various blocks (LUT4, FSM, and Counter Block) are numbered 0, 1, and 2.

- **Logic configuration registers (0x000 – 0x0FF):** These registers control the core reconfiguration logic for the tile. All registers in this group are EALLOW protected and also protected by the LOCK register.
- **Top level control registers (0x100 – 0x1FF):** These registers are used for top level and device related control of the CLB. These registers typically control mux selects for inputs, global enables, and so forth, and are accessible by normal memory mapped access. Some of these registers have EALLOW and LOCK protection.
- **Data exchange registers (0x200 – 0x3FF):** These registers are used to exchange data between the CLB and the rest of the device. The registers are accessible by normal memory-mapped access and no EALLOW or LOCK protection exists.

Note

EALLOW protection means that the write access to the register is enabled only when the EALLOW instruction has been executed prior to the write access. The complementary EDIS instruction disables access to all registers protected in this way. For more information, see Section 9.10.

9.5.2 Non-Memory Mapped Registers

The memory-mapped CLB registers are described later in this chapter; however, many of the CLB resources including counters, the instruction memory of the High Level Controller, and the HLC general-purpose registers (R0 through R3) are only indirectly accessible through a local interface bus and are not memory-mapped. These registers are accessible through the two memory-mapped registers CLB_LOAD_DATA and CLB_LOAD_ADDR. Note that the general-purpose registers R0 through R3 must only be written to during configuration-time and are not intended to be written to during run-time. Writes during run-time can lead to unexpected behavior. If run-time data exchange is desired, refer to Section 9.4.7.4.

Load the data to be written into the CLB_LOAD_DATA register, then load the appropriate address into CLB_LOAD_ADDR to determine where this data is written. Writing a 1 to bit position 0 in the CLB_LOAD_EN register then causes an internal write operation to be triggered. The address allocation for the CLB_LOAD_ADDR register is shown in Table 9-18.

Note

The COUNT_EVENT_CTRL_x bit must be configured to load (that is, 0) for indirect loads and for HLC loads of the counter to take effect.

Table 9-18. Non-Memory Mapped Register Addresses

Address (Binary)	Resource
000000 to 000010	Counter 0 to 2 Load value
000100 to 000110	Counter 0 to 2 Match1 value
001000 to 001010	Counter 0 to 2 Match2 value
001100 to 001111	R0 to R3 of High Level Controller
100000 to 100111	Instructions for Event 0
101000 to 101111	Instructions for Event 1
110000 to 110111	Instructions for Event 2
111000 to 111111	Instructions for Event 3

Use the following steps to load the value 0x11223344 into the general purpose R0 register:

1. Write 0x11223344 to CLB_LOAD_DATA.
2. Write 0xc to CLB_LOAD_ADDR.
3. Write 0x1 to CLB_LOAD_EN.

Note

Even though HLC registers are accessible by the CPU, your application code needs to make sure that no other CLB internal logics are updating the same HLC register at the same time, causing a race condition.

9.6 DMA Access

The DMA does not have access to the CLB memory-mapped registers, including the PUSH and PULL FIFO registers. For more information, refer to Section 9.10.

9.7 CLB Data Export Through SPI RX Buffer

For a continuous export of data from the CLB peripherals, SPI RX buffers can be used. CLB data can be exported through the SPI RX buffers without CPU/CLA interventions.

For the F2838x devices, CLB1 to CLB4 have access to SPIA to SPID, respectively, as shown in Table 9-19.

Table 9-19. CLB to SPI RX Access

CLB Instance	SPI Instance
CLB1	SPIA
CLB2	SPIB
CLB3	SPIC
CLB4	SPID

When the CLB to SPI data exporting is enabled, 16-bit data can be exported from CLB to SPI RX buffers. The 32-bit HLC R0 register is the data that is exported to the SPI RX buffers. The user can select which 16-bit range of the HLC R0 is exported by configuring the **CLB_SPI_DATA_CTRL_HI.SHIFT**. The CLB also controls when HLC R0 data must be transferred to the SPI RX buffer through **CLB_SPI_DATA_CTRL_HI.STRB** that selects one of the HLC event signals from the static switch block.

When CLB to SPI data exporting is required, note:

- The selected SPI transmit functionality is not affected.
- Even though the data is being pushed into the SPI RX buffers by the CLB, the SPI RX interrupt and the DMA trigger for SPI RX in the respective peripherals must be configured.
- The SPI can resume normal operation when the CLB to SPI data exporting is disabled.

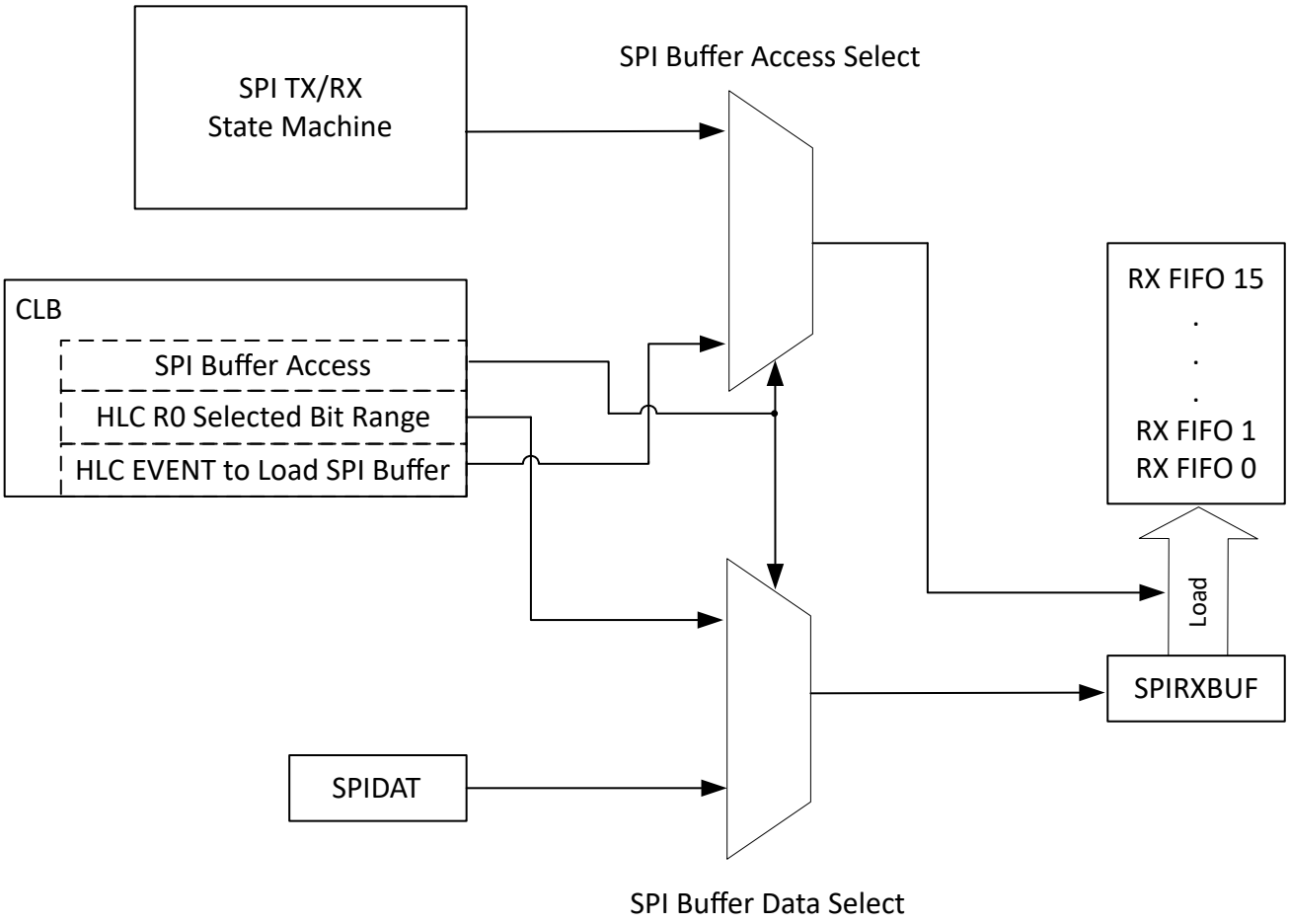


Figure 9-21. CLB Control of SPI RX Buffer

9.8 CLB Pipeline Mode

When operating the CLB TILE at frequencies higher than 100 MHz, the Pipeline mode MUST be enabled. When CLB Pipeline mode is enabled, the operations of the HLC and COUNTER modules are changed.

- When operating at higher frequencies that require Pipeline mode to be enabled, the pipelined versions of the the CLB CELL OUTPUTs are brought into the HLC. The Pipeline mode causes the CELL outputs delayed by 1 clock cycle to be used as the source of the HLC event triggers.
- In Pipeline mode, the COUNTER module's add/sub/shift operations, which are triggered by an event, use the value of the counter in the previous clock cycle (pipelined).

To enable the CLB Pipeline mode, set the CLB_LOAD_EN.PIPELINE_EN.

9.9 Software

9.9.1 CLB Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/clb

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

9.9.1.1 CLB Empty Project

FILE: empty.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

9.9.1.2 CLB Combinational Logic

FILE: clb_ex1_combinatorial_logic.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

The objective of this example is to prevent simultaneous high or low outputs on a PWM pair. PWM modules 1 and 2 are configured to generate identical waveforms based on a fixed frequency up-count mode.

9.9.1.3 CLB GPIO Input Filter

FILE: clb_ex2_gpio_input_filter.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

This example demonstrates use of finite state machines (FSMs) and counters to implement a simple ‘glitch’ filter which might, for example, be applied to an incoming GPIO signal to remove unwanted short duration pulses.

9.9.1.4 CLB Auxilary PWM

FILE: clb_ex3_auxiliary_pwm.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

This example configures a CLB tile as an auxiliary PWM generator. The example uses combinatorial logic (LUTs), state machines (FSMs), counters, and the high level controller (HLC) to demonstrate the PWM output generation capabilities using CLB.

9.9.1.5 CLB PWM Protection

FILE: clb_ex4_pwm_protection.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

This example extends the features of example 1 to ensure an active high complementary pair PWM configuration always operates with a minimum value of dead-band irrespective of how the generating PWM module is configured. The example illustrates the configuration of four separate PWM tiles to implement PWM protection on four PWM modules. The outputs of PWM modules 1 to 4 are operated on by CLB tiles 1 to 4, respectively.

9.9.1.6 CLB Event Window

FILE: clb_ex5_event_window.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

This example uses the counter, FSM, and HLC sub-modules of the CLB to implement an event timing feature which detects whether an interrupt service routine takes too long to respond to an interrupt. The example configures four PWM modules to operate in up-count mode and generate a low-to-high edge on a timer zero match event. The zero match event also triggers a PWM ISR which, for the purposes of this example, contains a dummy payload of variable length. At the end of the ISR, a write operation takes place to a CLB GP register to indicate the ISR has ended.

9.9.1.7 CLB Signal Generator

FILE: clb_ex6_siggen.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

This example uses CLB1 to generate a rectangular wave and CLB2 to check the rectangular wave generated by CLB1 doesn't exceed the defined duty cycle and period limits.

9.9.1.8 CLB State Machine

FILE: clb_ex7_state_machine.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\Designing With the C2000 CLB.pdf This application report describes the process of creating this CLB example and can be used as guidance on desiging custom logic with the CLB. This example uses all submodules inside a CLB TILE in order to implement a complete system.

9.9.1.9 CLB External Signal AND Gate

FILE: clb_ex8_external_signal_AND_gate.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example, two external signals from two GPIOs are passed through the Input X-BAR and the CLB X-BAR to the CLB TILE. Inside the CLB module these two signals are ANDED. The output of the AND gate is then exported to a GPIO, using Output X-BAR.

9.9.1.10 CLB Timer

FILE: clb_ex9_timer.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example, a COUNTER module is used to create timed events. The use of the GP Register is shown. Through setting/clearing the bits in the GP register, the timer is started, stopped or changes direction. The output of the timer event (1-clock cycle) is exported to a GPIO. Interrupts are generated from the timer event using the HLC module. A GPIO is also toggled inside the CLB ISR. The indirect CLB register access is used to update the timer's event match value and the active counter register to modify the frequency of the timer.

9.9.1.11 CLB Timer Two States

FILE: clb_ex10_timer_two_states.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example, the timer is setup the same as the previous example. The difference is the use of the FSM submodule to toggle the output of the CLB which is then exported to a GPIO. The FSM module acts as a single bit memory block. Interrupts are setup in the same format as the previous example. The interrupt delay of the CLB can be seen by comparing the output of the CLB and the GPIO toggled in the ISR.

9.9.1.12 CLB Interrupt Tag

FILE: clb_ex11_interrupt_tag.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example, a timer is setup with two different match values. These two events are used by the HLC submodule to generate interrupts. The interrupt TAG is used to differentiate between the interrupt generated due to the match1 event of the CLB counter and the match2 event of the CLB counter.

9.9.1.13 CLB Output Intersect

FILE: clb_ex12_output_intersect.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example, the CLB module is set up the same as the external_AND_gate example. However, instead of the output being exported to the GPIO using Output X-BAR, the output is exported to the GPIO by replacing the output of ePWM1. This is done by configuring the GPIO for EPWM1A output, followed by enabling output intersection.

9.9.1.14 CLB PUSH PULL

FILE: clb_ex13_push_pull.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example, the use of the PUSH-PULL interface is shown. Multiple COUNTER submodules, HLC submodule, FSM submodules, and OUTLUT submodules are used. The PUSH-PULL interface is used alongside the GP register to update the COUNTER submodules' event frequencies.

9.9.1.15 CLB Multi Tile

FILE: clb_ex14_multi_tile.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example the output of a CLB TILE is passed to the input of another CLB TILE. The output of the second CLB TILE is then exported to a GPIO, showcasing how two CLB TILEs can be used in series.

9.9.1.16 CLB Tile to Tile Delay

FILE: clb_ex15_tile_to_tile_delay.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example the output of a GPIO is taken into the CLB TILE through INPUT XBAR and the CLB XBAR. The signal is forwarded by the TILE to the next TILE. This time the signal only goes through the CLB XBAR and NOT the Input XBAR. This is done to show that delays are added when the signals are passed from TILE to TILE and the delay is NOT characterized. The user should always avoid passing signals with timing requirements between tiles. The COUNTER modules inside the CLBs will count the amount of delay in cycles.

9.9.1.17 CLB based One-shot PWM

FILE: clb_ex17_one_shot_pwm.c

For the detailed description of this example, please refer to :
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

9.9.1.18 CLB AOC Control

FILE: clb_ex18_aoc.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example the Asynchronous Output Conditioning block is used to asynchronously AND gate the input signals to the CLB. This module is only available for CLB types 2 and up.

9.9.1.19 CLB AOC Release Control

FILE: clb_ex19_aoc_release_control.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example the Asynchronous Output Conditioning block is used to asynchronously set/release the input signals to the CLB. This module is only available for CLB types 2 and up.

9.9.1.20 CLB XBARs

FILE: clb_ex20_clbxbars.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example the CLB INPUTXBAR and CLB OUTPUTXBAR are used to take input signals from GPIOs into the CLB TILEs and take output signal from the TILE to GPIOs. The availability of these XBARs are device dependent.

9.9.1.21 CLB AOC Control

FILE: clb_ex21_clockprescalar_nmi.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example the clock prescalar of the CLB module is used to divide down the CLB clock and use it as an input to the TILE logic. Also the HLC module is used to generate NMI interrupts. This module is only available for CLB types 2 and up.

9.9.1.22 CLB Serializer

FILE: clb_ex22_serializer.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example the CLB COUNTER is used in serializer mode to act as a shift register. This module is only available for CLB types 2 and up.

9.9.1.23 CLB LFSR

FILE: clb_ex23_lfsr.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example the CLB COUNTER module is used in Linear Feedback Shift Register (LFSR) mode. This module is only available for CLB types 2 and up.

9.9.1.24 CLB Lock Output Mask

FILE: clb_ex24_lock_output_mask.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example the lock outpt mask feature of the CLB is used to lock the selected output signal override settings. This module is only available for CLB types 3 and up.

9.9.1.25 CLB INPUT Pipeline Mode

FILE: clb_ex25_input_pipeline.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example the CLB Input Pipeline mode is enable to delay the input signal by a clock cycle. This module is only available for CLB types 3 and up.

9.9.1.26 CLB Clocking and PIPELINE Mode

FILE: clb_ex26_clocking_pipeline.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example the CLB pipeline mode is enable and affects the behavior of the CLB COUNTERs and HLC. This module is only available for CLB types 3 and up.

9.9.1.27 CLB SPI Data Export

FILE: clb_ex27_spi_data_export.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example the high speed data export feature of the CLB is used and one of the HLC registers is exported out of the CLB module using the SPI RX buffer. This module is only available for CLB types 3 and up.

9.9.1.28 CLB SPI Data Export DMA

FILE: clb_ex28_spi_data_export_dma.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example the high speed data export feature of the CLB is used and one of the HLC registers is exported out of the CLB module using the SPI RX buffer. The data received in the SPI RX buffer is transferred to memory using DMA. This module is only available for CLB types 3 and up.

9.9.1.29 CLB Trip Zone Timestamp

FILE: clb_ex29_timestamp.c

This example displays how to timestamp interrupts generated by the CLB. An interrupt is generated when ePWM1 is tripped.

ePWM1 is configured to be interrupted by TZ1 and TZ2, both one shot trip sources.

The CLB is configured as follows:
- COUNTER0 and COUNTER1 continually count when the program begins.
- COUNTER0 timestamps TZ1 and COUNTER1 timestamps TZ2.
- COUNTER2 increments once when COUNTER0/COUNTER1 overflows using LUT2.
- FSM0/1 are configured to sync counters and stop COUNTER0/1 when an interrupt is received.
- TZ1 (GPIO12) and TZ2 (GPIO13) are routed as inputs through CLBXBAR.
- BOUNDARY.boundaryInput0 denotes TZ1. On rising edge, HLC issues an interrupt with tag 12.
- BOUDNARY.in1 denotes TZ2. On rising edge, HLC issues an interrupt with tag 13.
- BOUNDARY.boundaryInput7 serves as a simultaneous enable for COUNTER0/1 to begin counting.

TZ1 is tripped when GPIO12 is connected to GND. TZ2 is tripped when GPIO13 is connected to GND. When an interrupt occurs, the interrupt handler determines the initial trip source and stores this value in a variable 'initialTripZone'.

View these variables in Debug Expressions tab:

initialTripZone: stores the first TZ to have been tripped tz1Counter64bit: stores the counter value at the instant that TZ1 is tripped. tz2Counter64bit: stores the counter value at the instant that TZ2 is tripped.

9.9.1.30 CLB CRC

FILE: clb_ex30_cyclic_redundancy_check.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

In this example, the CLB module is used to perform the cyclic redundancy check (C.R.C.) with twelve messages in bits checked with ten different CRC polynomials.

First element passed in is message length, second is the message stored in input_data

This example is only available for CLB types 2 and up.

The known values in the output_data are compared with expected values from the CLB-based CRC calculation. A total of 120 messages are verified, and the number of matching messages are displayed in passCount

Variables to add to Watch Expressions in debug view: passCount - number of messages that match between generated and known CRC values failCount - number of messages that fail the CRC value verification

9.10 CLB Registers

This section describes the Configurable Logic Block Registers.

9.10.1 CLB Base Address Table (C28)

Table 9-20. CLB Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
Clb1LogicCfgRegs	CLB_LOGIC_CONFIG_REGS	CLB1_LOGICCFG_BASE	0x0000_3000	YES	YES	-	YES	-
Clb1LogicCtrlRegs	CLB_LOGIC_CONTROL_REGS	CLB1_LOGICCTL_BASE	0x0000_3100	YES	YES	-	YES	-
Clb1DataExchRegs	CLB_DATA_EXCHANGE_REGS	CLB1_DATAEXCH_BASE	0x0000_3180	YES	YES	-	YES	-
Clb2LogicCfgRegs	CLB_LOGIC_CONFIG_REGS	CLB2_LOGICCFG_BASE	0x0000_3200	YES	YES	-	YES	-
Clb2LogicCtrlRegs	CLB_LOGIC_CONTROL_REGS	CLB2_LOGICCTL_BASE	0x0000_3300	YES	YES	-	YES	-
Clb2DataExchRegs	CLB_DATA_EXCHANGE_REGS	CLB2_DATAEXCH_BASE	0x0000_3380	YES	YES	-	YES	-
Clb3LogicCfgRegs	CLB_LOGIC_CONFIG_REGS	CLB3_LOGICCFG_BASE	0x0000_3400	YES	YES	-	YES	-
Clb3LogicCtrlRegs	CLB_LOGIC_CONTROL_REGS	CLB3_LOGICCTL_BASE	0x0000_3500	YES	YES	-	YES	-
Clb3DataExchRegs	CLB_DATA_EXCHANGE_REGS	CLB3_DATAEXCH_BASE	0x0000_3580	YES	YES	-	YES	-
Clb4LogicCfgRegs	CLB_LOGIC_CONFIG_REGS	CLB4_LOGICCFG_BASE	0x0000_3600	YES	YES	-	YES	-
Clb4LogicCtrlRegs	CLB_LOGIC_CONTROL_REGS	CLB4_LOGICCTL_BASE	0x0000_3700	YES	YES	-	YES	-
Clb4DataExchRegs	CLB_DATA_EXCHANGE_REGS	CLB4_DATAEXCH_BASE	0x0000_3780	YES	YES	-	YES	-
Clb5LogicCfgRegs	CLB_LOGIC_CONFIG_REGS	CLB5_LOGICCFG_BASE	0x0000_3800	YES	YES	-	YES	-
Clb5LogicCtrlRegs	CLB_LOGIC_CONTROL_REGS	CLB5_LOGICCTL_BASE	0x0000_3900	YES	YES	-	YES	-
Clb5DataExchRegs	CLB_DATA_EXCHANGE_REGS	CLB5_DATAEXCH_BASE	0x0000_3980	YES	YES	-	YES	-
Clb6LogicCfgRegs	CLB_LOGIC_CONFIG_REGS	CLB6_LOGICCFG_BASE	0x0000_3A00	YES	YES	-	YES	-
Clb6LogicCtrlRegs	CLB_LOGIC_CONTROL_REGS	CLB6_LOGICCTL_BASE	0x0000_3B00	YES	YES	-	YES	-

Table 9-20. CLB Base Address Table (C28) (continued)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
Clb6DataExchRegs	CLB_DATA_EXCHANGE_REGS	CLB6_DATAEXCH_BASE	0x0000_3B80	YES	YES	-	YES	-
Clb7LogicCfgRegs	CLB_LOGIC_CONFIG_REGS	CLB7_LOGICCFG_BASE	0x0000_3C00	YES	YES	-	YES	-
Clb7LogicCtrlRegs	CLB_LOGIC_CONTROL_REGS	CLB7_LOGICCTL_BASE	0x0000_3D00	YES	YES	-	YES	-
Clb7DataExchRegs	CLB_DATA_EXCHANGE_REGS	CLB7_DATAEXCH_BASE	0x0000_3D80	YES	YES	-	YES	-
Clb8LogicCfgRegs	CLB_LOGIC_CONFIG_REGS	CLB8_LOGICCFG_BASE	0x0000_3E00	YES	YES	-	YES	-
Clb8LogicCtrlRegs	CLB_LOGIC_CONTROL_REGS	CLB8_LOGICCTL_BASE	0x0000_3F00	YES	YES	-	YES	-
Clb8DataExchRegs	CLB_DATA_EXCHANGE_REGS	CLB8_DATAEXCH_BASE	0x0000_3F80	YES	YES	-	YES	-

9.10.2 CLB_LOGIC_CONFIG_REGS Registers

Table 9-21 lists the memory-mapped registers for the CLB_LOGIC_CONFIG_REGS registers. All register offset addresses not listed in Table 9-21 should be considered as reserved locations and the register contents should not be modified.

Table 9-21. CLB_LOGIC_CONFIG_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
2h	CLB_COUNT_RESET	Counter Block RESET	EALLOW, LOCK	Go
4h	CLB_COUNT_MODE_1	Counter Block MODE_1	EALLOW, LOCK	Go
6h	CLB_COUNT_MODE_0	Counter Block MODE_0	EALLOW, LOCK	Go
8h	CLB_COUNT_EVENT	Counter Block EVENT	EALLOW, LOCK	Go
Ah	CLB_FSM_EXTRA_IN0	FSM Extra EXT_IN0	EALLOW, LOCK	Go
Ch	CLB_FSM_EXTERNAL_IN0	FSM EXT_IN0	EALLOW, LOCK	Go
Eh	CLB_FSM_EXTERNAL_IN1	FSM_EXT_IN1	EALLOW, LOCK	Go
10h	CLB_FSM_EXTRA_IN1	FSM Extra_EXT_IN1	EALLOW, LOCK	Go
12h	CLB_LUT4_IN0	LUT4_0/1/2 IN0 input source	EALLOW, LOCK	Go
14h	CLB_LUT4_IN1	LUT4_0/1/2 IN1 input source	EALLOW, LOCK	Go
16h	CLB_LUT4_IN2	LUT4_0/1/2 IN2 input source	EALLOW, LOCK	Go
18h	CLB_LUT4_IN3	LUT4_0/1/2 IN3 input source	EALLOW, LOCK	Go
1Ch	CLB_FSM_LUT_FN1_0	LUT function for FSM Unit 1 and Unit 0	EALLOW, LOCK	Go
1Eh	CLB_FSM_LUT_FN2	LUT function for FSM Unit 2	EALLOW, LOCK	Go
20h	CLB_LUT4_FN1_0	LUT function for LUT4 block of Unit 1 and 0	EALLOW, LOCK	Go
22h	CLB_LUT4_FN2	LUT function for LUT4 block of Unit 2	EALLOW, LOCK	Go
24h	CLB_FSM_NEXT_STATE_0	FSM Next state equations for Unit 0	EALLOW, LOCK	Go
26h	CLB_FSM_NEXT_STATE_1	FSM Next state equations for Unit 1	EALLOW, LOCK	Go
28h	CLB_FSM_NEXT_STATE_2	FSM Next state equations for Unit 2	EALLOW, LOCK	Go
2Ah	CLB_MISC_CONTROL	Static controls for Ctr,FSM	EALLOW, LOCK	Go
2Ch	CLB_OUTPUT_LUT_0	Inp Sel, LUT fns for Out0	EALLOW, LOCK	Go
2Eh	CLB_OUTPUT_LUT_1	Inp Sel, LUT fns for Out1	EALLOW, LOCK	Go
30h	CLB_OUTPUT_LUT_2	Inp Sel, LUT fns for Out2	EALLOW, LOCK	Go
32h	CLB_OUTPUT_LUT_3	Inp Sel, LUT fns for Out3	EALLOW, LOCK	Go
34h	CLB_OUTPUT_LUT_4	Inp Sel, LUT fns for Out4	EALLOW, LOCK	Go
36h	CLB_OUTPUT_LUT_5	Inp Sel, LUT fns for Out5	EALLOW, LOCK	Go
38h	CLB_OUTPUT_LUT_6	Inp Sel, LUT fns for Out6	EALLOW, LOCK	Go
3Ah	CLB_OUTPUT_LUT_7	Inp Sel, LUT fns for Out7	EALLOW, LOCK	Go
3Ch	CLB_HLC_EVENT_SEL	Event Selector register for the High Level controller	EALLOW, LOCK	Go
3Eh	CLB_COUNT_MATCH_TAP_SEL	Counter tap values for match1 and match2 outputs	EALLOW, LOCK	Go
40h	CLB_OUTPUT_COND_CTRL_0	Output conditioning control for output 0	EALLOW, LOCK	Go
42h	CLB_OUTPUT_COND_CTRL_1	Output conditioning control for output 1	EALLOW, LOCK	Go
44h	CLB_OUTPUT_COND_CTRL_2	Output conditioning control for output 2	EALLOW, LOCK	Go
46h	CLB_OUTPUT_COND_CTRL_3	Output conditioning control for output 3	EALLOW, LOCK	Go
48h	CLB_OUTPUT_COND_CTRL_4	Output conditioning control for output 4	EALLOW, LOCK	Go
4Ah	CLB_OUTPUT_COND_CTRL_5	Output conditioning control for output 5	EALLOW, LOCK	Go
4Ch	CLB_OUTPUT_COND_CTRL_6	Output conditioning control for output 6	EALLOW, LOCK	Go
4Eh	CLB_OUTPUT_COND_CTRL_7	Output conditioning control for output 7	EALLOW, LOCK	Go
50h	CLB_MISC_ACCESS_CTRL	Miscellaneous Access and enable control	EALLOW, LOCK	Go

Table 9-21. CLB_LOGIC_CONFIG_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
51h	CLB_SPI_DATA_CTRL_HI	CLB to SPI buffer control High	EALLOW, LOCK	Go

Complex bit access types are encoded to fit into small table cells. Table 9-22 shows the codes that are used for access types in this section.

Table 9-22. CLB_LOGIC_CONFIG_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

9.10.2.1 CLB_COUNT_RESET Register (Offset = 2h) [Reset = 0h]

CLB_COUNT_RESET is shown in Figure 9-22 and described in Table 9-23.

Return to the Summary Table.

Counter Block RESET

Figure 9-22. CLB_COUNT_RESET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R/W1C-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 9-23. CLB_COUNT_RESET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14-10	SEL_2	R/W	0h	Counter reset select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	Counter reset select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	Counter reset select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.2 CLB_COUNT_MODE_1 Register (Offset = 4h) [Reset = 0h]

CLB_COUNT_MODE_1 is shown in Figure 9-23 and described in Table 9-24.

Return to the Summary Table.

Counter Block MODE_1

Figure 9-23. CLB_COUNT_MODE_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R/W1C-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 9-24. CLB_COUNT_MODE_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14-10	SEL_2	R/W	0h	Counter MODE_1 select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	Counter MODE_1 select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	Counter MODE_1 select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.3 CLB_COUNT_MODE_0 Register (Offset = 6h) [Reset = 0h]

CLB_COUNT_MODE_0 is shown in Figure 9-24 and described in Table 9-25.

Return to the Summary Table.

Counter Block MODE_0

Figure 9-24. CLB_COUNT_MODE_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R/W1C-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 9-25. CLB_COUNT_MODE_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14-10	SEL_2	R/W	0h	Counter MODE_0 select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	Counter MODE_0 select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	Counter MODE_0 select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.4 CLB_COUNT_EVENT Register (Offset = 8h) [Reset = 0h]

CLB_COUNT_EVENT is shown in Figure 9-25 and described in Table 9-26.

Return to the Summary Table.

Counter Block EVENT

Figure 9-25. CLB_COUNT_EVENT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R/W1C-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 9-26. CLB_COUNT_EVENT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14-10	SEL_2	R/W	0h	Counter event select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	Counter event select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	Counter event select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.5 CLB_FSM_EXTRA_IN0 Register (Offset = Ah) [Reset = 0h]

CLB_FSM_EXTRA_IN0 is shown in Figure 9-26 and described in Table 9-27.

Return to the Summary Table.

FSM Extra EXT_IN0

Figure 9-26. CLB_FSM_EXTRA_IN0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R/W1C-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 9-27. CLB_FSM_EXTRA_IN0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14-10	SEL_2	R/W	0h	FSM block extra external IN0 select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	FSM block extra external IN0 select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	FSM block extra external IN0 select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.6 CLB_FSM_EXTERNAL_IN0 Register (Offset = Ch) [Reset = 0h]

CLB_FSM_EXTERNAL_IN0 is shown in Figure 9-27 and described in Table 9-28.

Return to the Summary Table.

FSM EXT_IN0

Figure 9-27. CLB_FSM_EXTERNAL_IN0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R/W1C-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 9-28. CLB_FSM_EXTERNAL_IN0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14-10	SEL_2	R/W	0h	FSM block EXT_IN0 select input for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	FSM block EXT_IN0 select input for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	FSM block EXT_IN0 select input for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.7 CLB_FSM_EXTERNAL_IN1 Register (Offset = Eh) [Reset = 0h]

CLB_FSM_EXTERNAL_IN1 is shown in Figure 9-28 and described in Table 9-29.

Return to the Summary Table.

FSM_EXT_IN1

Figure 9-28. CLB_FSM_EXTERNAL_IN1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R/W1C-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 9-29. CLB_FSM_EXTERNAL_IN1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14-10	SEL_2	R/W	0h	FSM block EXT_IN1 select input for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	FSM block EXT_IN1 select input for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	FSM block EXT_IN1 select input for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.8 CLB_FSM_EXTRA_IN1 Register (Offset = 10h) [Reset = 0h]

CLB_FSM_EXTRA_IN1 is shown in Figure 9-29 and described in Table 9-30.

Return to the Summary Table.

FSM Extra_EXT_IN1

Figure 9-29. CLB_FSM_EXTRA_IN1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R/W1C-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 9-30. CLB_FSM_EXTRA_IN1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14-10	SEL_2	R/W	0h	FSM block extra external IN1 select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	FSM block extra external IN1 select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	FSM block extra external IN1 select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.9 CLB_LUT4_IN0 Register (Offset = 12h) [Reset = 0h]

CLB_LUT4_IN0 is shown in Figure 9-30 and described in Table 9-31.

Return to the Summary Table.

LUT4_0/1/2 IN0 input source

Figure 9-30. CLB_LUT4_IN0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R/W1C-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 9-31. CLB_LUT4_IN0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14-10	SEL_2	R/W	0h	LUT4 block IN0 select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	LUT4 block IN0 select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	LUT4 block IN0 select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.10 CLB_LUT4_IN1 Register (Offset = 14h) [Reset = 0h]

CLB_LUT4_IN1 is shown in Figure 9-31 and described in Table 9-32.

Return to the Summary Table.

LUT4_0/1/2 IN1 input source

Figure 9-31. CLB_LUT4_IN1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R/W1C-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 9-32. CLB_LUT4_IN1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14-10	SEL_2	R/W	0h	LUT4 block IN1 select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	LUT4 block IN1 select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	LUT4 block IN1 select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.11 CLB_LUT4_IN2 Register (Offset = 16h) [Reset = 0h]

CLB_LUT4_IN2 is shown in Figure 9-32 and described in Table 9-33.

Return to the Summary Table.

LUT4_0/1/2 IN2 input source

Figure 9-32. CLB_LUT4_IN2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R/W1C-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 9-33. CLB_LUT4_IN2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14-10	SEL_2	R/W	0h	LUT4 block IN2 select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	LUT4 block IN2 select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	LUT4 block IN2 select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.12 CLB_LUT4_IN3 Register (Offset = 18h) [Reset = 0h]

CLB_LUT4_IN3 is shown in Figure 9-33 and described in Table 9-34.

Return to the Summary Table.

LUT4_0/1/2 IN3 input source

Figure 9-33. CLB_LUT4_IN3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R/W1C-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 9-34. CLB_LUT4_IN3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14-10	SEL_2	R/W	0h	LUT4 block IN3 select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	LUT4 block IN3 select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	LUT4 block IN3 select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.13 CLB_FSM_LUT_FN1_0 Register (Offset = 1Ch) [Reset = 0h]

CLB_FSM_LUT_FN1_0 is shown in Figure 9-34 and described in Table 9-35.

Return to the Summary Table.

LUT function for FSM Unit 1 and Unit 0

Figure 9-34. CLB_FSM_LUT_FN1_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FN1																FN0															
R/W-0h																R/W-0h															

Table 9-35. CLB_FSM_LUT_FN1_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	FN1	R/W	0h	FSM block LUT output function for unit 1 Reset type: SYSRSn
15-0	FN0	R/W	0h	FSM block LUT output function for unit 0 Reset type: SYSRSn

9.10.2.14 CLB_FSM_LUT_FN2 Register (Offset = 1Eh) [Reset = 0h]

CLB_FSM_LUT_FN2 is shown in Figure 9-35 and described in Table 9-36.

Return to the Summary Table.

LUT function for FSM Unit 2

Figure 9-35. CLB_FSM_LUT_FN2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																FN1															
R/W1C-0h																R/W-0h															

Table 9-36. CLB_FSM_LUT_FN2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W1C	0h	Reserved
15-0	FN1	R/W	0h	LUT4 output function for unit 2 Reset type: SYSRSn

9.10.2.15 CLB_LUT4_FN1_0 Register (Offset = 20h) [Reset = 0h]

CLB_LUT4_FN1_0 is shown in Figure 9-36 and described in Table 9-37.

Return to the Summary Table.

LUT function for LUT4 block of Unit 1 and 0

Figure 9-36. CLB_LUT4_FN1_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FN1																FN0															
R/W-0h																R/W-0h															

Table 9-37. CLB_LUT4_FN1_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	FN1	R/W	0h	LUT4 output function for unit 1 Reset type: SYSRSn
15-0	FN0	R/W	0h	LUT4 output function for unit 0 Reset type: SYSRSn

9.10.2.16 CLB_LUT4_FN2 Register (Offset = 22h) [Reset = 0h]

CLB_LUT4_FN2 is shown in Figure 9-37 and described in Table 9-38.

Return to the Summary Table.

LUT function for LUT4 block of Unit 2

Figure 9-37. CLB_LUT4_FN2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																FN1															
R/W1C-0h																R/W-0h															

Table 9-38. CLB_LUT4_FN2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W1C	0h	Reserved
15-0	FN1	R/W	0h	LUT4 output function for unit 2 Reset type: SYSRSn

9.10.2.17 CLB_FSM_NEXT_STATE_0 Register (Offset = 24h) [Reset = 0h]

CLB_FSM_NEXT_STATE_0 is shown in Figure 9-38 and described in Table 9-39.

Return to the Summary Table.

FSM Next state equations for Unit 0

Figure 9-38. CLB_FSM_NEXT_STATE_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S1																S0															
R/W-0h																R/W-0h															

Table 9-39. CLB_FSM_NEXT_STATE_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	S1	R/W	0h	FSM next state function for S1, unit0 Reset type: SYSRSn
15-0	S0	R/W	0h	FSM next state function for S0, unit0 Reset type: SYSRSn

9.10.2.18 CLB_FSM_NEXT_STATE_1 Register (Offset = 26h) [Reset = 0h]

CLB_FSM_NEXT_STATE_1 is shown in Figure 9-39 and described in Table 9-40.

Return to the Summary Table.

FSM Next state equations for Unit 1

Figure 9-39. CLB_FSM_NEXT_STATE_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S1																S0															
R/W-0h																R/W-0h															

Table 9-40. CLB_FSM_NEXT_STATE_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	S1	R/W	0h	FSM next state function for S1, unit1 Reset type: SYSRSn
15-0	S0	R/W	0h	FSM next state function for S0, unit1 Reset type: SYSRSn

9.10.2.19 CLB_FSM_NEXT_STATE_2 Register (Offset = 28h) [Reset = 0h]

CLB_FSM_NEXT_STATE_2 is shown in Figure 9-40 and described in Table 9-41.

Return to the Summary Table.

FSM Next state equations for Unit 2

Figure 9-40. CLB_FSM_NEXT_STATE_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S1																S0															
R/W-0h																R/W-0h															

Table 9-41. CLB_FSM_NEXT_STATE_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	S1	R/W	0h	FSM next state function for S1, unit2 Reset type: SYSRSn
15-0	S0	R/W	0h	FSM next state function for S0, unit2 Reset type: SYSRSn

9.10.2.20 CLB_MISC_CONTROL Register (Offset = 2Ah) [Reset = 0h]

CLB_MISC_CONTROL is shown in Figure 9-41 and described in Table 9-42.

Return to the Summary Table.

Static controls for Ctr,FSM

Figure 9-41. CLB_MISC_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED					COUNT2_LFSR_EN	COUNT1_LFSR_EN	COUNT0_LFSR_EN
R-0-0h					R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
COUNT2_MATCH2_TAP_EN	COUNT1_MATCH2_TAP_EN	COUNT0_MATCH2_TAP_EN	COUNT2_MATCH1_TAP_EN	COUNT1_MATCH1_TAP_EN	COUNT0_MATCH1_TAP_EN	FSM_EXTRA_SEL1_2	FSM_EXTRA_SEL0_2
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
FSM_EXTRA_SEL1_1	FSM_EXTRA_SEL0_1	FSM_EXTRA_SEL1_0	FSM_EXTRA_SEL0_0	COUNT_SERIALIZER_2	COUNT_SERIALIZER_1	COUNT_SERIALIZER_0	COUNT_EVENT_CTRL_2
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
COUNT_DIR_2	COUNT_ADD_SHIFT_2	COUNT_EVENT_CTRL_1	COUNT_DIR_1	COUNT_ADD_SHIFT_1	COUNT_EVENT_CTRL_0	COUNT_DIR_0	COUNT_ADD_SHIFT_0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 9-42. CLB_MISC_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R-0	0h	Reserved
26	COUNT2_LFSR_EN	R/W	0h	Defines if Counter 2 should operate in LFSR mode. This should be set to 1 only if it is in the SERIALIZER mode. 0 = Selects normal serializer operation 1 = Selects LFSR mode of operation Reset type: SYSRSn
25	COUNT1_LFSR_EN	R/W	0h	Defines if Counter 1 should operate in LFSR mode. This should be set to 1 only if it is in the SERIALIZER mode. 0 = Selects normal serializer operation 1 = Selects LFSR mode of operation Reset type: SYSRSn
24	COUNT0_LFSR_EN	R/W	0h	Defines if Counter 0 should operate in LFSR mode. This should be set to 1 only if it is in the SERIALIZER mode. 0 = Selects normal serializer operation 1 = Selects LFSR mode of operation Reset type: SYSRSn
23	COUNT2_MATCH2_TAP_EN	R/W	0h	Defines if the Match2 output should come from the match unit or tapped from a bit position of the counter 0 = Selects Match2 comaprison output 1 = Selects Bit position defined by Match2_Tap_val Reset type: SYSRSn
22	COUNT1_MATCH2_TAP_EN	R/W	0h	Defines if the Match2 output should come from the match unit or tapped from a bit position of the counter 0 = Selects Match2 comaprison output 1 = Selects Bit position defined by Match2_Tap_val Reset type: SYSRSn

Table 9-42. CLB_MISC_CONTROL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	COUNT0_MATCH2_TAP_EN	R/W	0h	Defines if the Match2 output should come from the match unit or tapped from a bit position of the counter 0 = Selects Match2 comaprison output 1 = Selects Bit position defined by Match2_Tap_val Reset type: SYSRSn
20	COUNT2_MATCH1_TAP_EN	R/W	0h	Defines if the Match1 output should come from the match unit or tapped from a bit position of the counter 0 = Selects Match1 comaprison output 1 = Selects Bit position defined by Match1_Tap_val Reset type: SYSRSn
19	COUNT1_MATCH1_TAP_EN	R/W	0h	Defines if the Match1 output should come from the match unit or tapped from a bit position of the counter 0 = Selects Match1 comaprison output 1 = Selects Bit position defined by Match1_Tap_val Reset type: SYSRSn
18	COUNT0_MATCH1_TAP_EN	R/W	0h	Defines if the Match1 output should come from the match unit or tapped from a bit position of the counter 0 = Selects Match1 comaprison output 1 = Selects Bit position defined by Match1_Tap_val Reset type: SYSRSn
17	FSM_EXTRA_SEL1_2	R/W	0h	Defines which input should be selected for the FSM LUT of UNIT 2 0 = Selects State S1 for the FSM LUT 1 = Selects EXTRA_EXT_IN1 for the FSM LUT Reset type: SYSRSn
16	FSM_EXTRA_SEL0_2	R/W	0h	Defines which input should be selected for the FSM LUT of UNIT 2 0 = Selects State S0 for the FSM LUT 1 = Selects EXTRA_EXT_IN0 for the FSM LUT Reset type: SYSRSn
15	FSM_EXTRA_SEL1_1	R/W	0h	Defines which input should be selected for the FSM LUT of UNIT 1 0 = Selects State S1 for the FSM LUT 1 = Selects EXTRA_EXT_IN1 for the FSM LUT Reset type: SYSRSn
14	FSM_EXTRA_SEL0_1	R/W	0h	Defines which input should be selected for the FSM LUT of UNIT 1 0 = Selects State S0 for the FSM LUT 1 = Selects EXTRA_EXT_IN0 for the FSM LUT Reset type: SYSRSn
13	FSM_EXTRA_SEL1_0	R/W	0h	Defines which input should be selected for the FSM LUT of UNIT 0 0 = Selects State S1 for the FSM LUT 1 = Selects EXTRA_EXT_IN1 for the FSM LUT Reset type: SYSRSn
12	FSM_EXTRA_SEL0_0	R/W	0h	Defines which input should be selected for the FSM LUT of UNIT 0 0 = Selects State S0 for the FSM LUT 1 = Selects EXTRA_EXT_IN0 for the FSM LUT Reset type: SYSRSn
11	COUNT_SERIALIZER_2	R/W	0h	Controls if the Counter of UNIT 2 is the Serialzer mode or not. 0 = Normal mode 1 = Serializer mode Reset type: SYSRSn
10	COUNT_SERIALIZER_1	R/W	0h	Controls if the Counter of UNIT 1 is the Serialzer mode or not. 0 = Normal mode 1 = Serializer mode Reset type: SYSRSn
9	COUNT_SERIALIZER_0	R/W	0h	Controls if the Counter of UNIT 0 is the Serialzer mode or not. 0 = Normal mode 1 = Serializer mode Reset type: SYSRSn

Table 9-42. CLB_MISC_CONTROL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	COUNT_EVENT_CTRL_2	R/W	0h	Controls the actions on an EVENT for UNIT2. Must be 0 for indirect loads and HLC loads of the counter to take effect. 0 = No add or shift, but load the predefined value 1 = Based on other bits, add/shift with the predefined value Reset type: SYSRSn
7	COUNT_DIR_2	R/W	0h	Controls add/shift direction for UNIT 2 0 = right shift or subtract 1 = left shift or add Reset type: SYSRSn
6	COUNT_ADD_SHIFT_2	R/W	0h	Controls whether the UNIT 2 counter will do an ADD or a SHIFT on an EVENT. 0 =Shift 1 = ADD Reset type: SYSRSn
5	COUNT_EVENT_CTRL_1	R/W	0h	Controls the actions on an EVENT for UNIT1. Must be 0 for indirect loads and HLC loads of the counter to take effect. 0 = No add or shift, but load the predefined value 1 = Based on other bits, add/shift with the predefined value Reset type: SYSRSn
4	COUNT_DIR_1	R/W	0h	Controls add/shift direction for UNIT 1 0 = right shift or subtract 1 = left shift or add Reset type: SYSRSn
3	COUNT_ADD_SHIFT_1	R/W	0h	Controls whether the UNIT 1 counter will do an ADD or a SHIFT on an EVENT. 0 = Shift 1 = ADD Reset type: SYSRSn
2	COUNT_EVENT_CTRL_0	R/W	0h	Controls the actions on an EVENT for UNIT1. Must be 0 for indirect loads and HLC loads of the counter to take effect. 0 = No add or shift, but load the predefined value 1 = Based on other bits, add/shift with the predefined value Reset type: SYSRSn
1	COUNT_DIR_0	R/W	0h	Controls add/shift direction for UNIT 0 0 = right shift or subtract 1 = left shift or add Reset type: SYSRSn
0	COUNT_ADD_SHIFT_0	R/W	0h	Controls whether the UNIT 0 counter will do an ADD or a SHIFT on an EVENT. 0 = Shift 1 = ADD Reset type: SYSRSn

9.10.2.21 CLB_OUTPUT_LUT_0 Register (Offset = 2Ch) [Reset = 0h]

CLB_OUTPUT_LUT_0 is shown in Figure 9-42 and described in Table 9-43.

Return to the Summary Table.

Inp Sel, LUT fns for Out0

Figure 9-42. CLB_OUTPUT_LUT_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									FN								IN2					IN1					IN0				
R/W1C-0h									R/W-0h								R/W-0h					R/W-0h					R/W-0h				

Table 9-43. CLB_OUTPUT_LUT_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R/W1C	0h	Reserved
22-15	FN	R/W	0h	Output function for output LUT Reset type: SYSRSn
14-10	IN2	R/W	0h	Select value for IN2 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	IN1	R/W	0h	Select value for IN1 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	IN0	R/W	0h	Select value for IN0 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.22 CLB_OUTPUT_LUT_1 Register (Offset = 2Eh) [Reset = 0h]

CLB_OUTPUT_LUT_1 is shown in Figure 9-43 and described in Table 9-44.

Return to the Summary Table.

Inp Sel, LUT fns for Out1

Figure 9-43. CLB_OUTPUT_LUT_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									FN								IN2					IN1					IN0				
R/W1C-0h									R/W-0h								R/W-0h					R/W-0h					R/W-0h				

Table 9-44. CLB_OUTPUT_LUT_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R/W1C	0h	Reserved
22-15	FN	R/W	0h	Output function for output LUT Reset type: SYSRSn
14-10	IN2	R/W	0h	Select value for IN2 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	IN1	R/W	0h	Select value for IN1 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	IN0	R/W	0h	Select value for IN0 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.23 CLB_OUTPUT_LUT_2 Register (Offset = 30h) [Reset = 0h]

CLB_OUTPUT_LUT_2 is shown in Figure 9-44 and described in Table 9-45.

Return to the Summary Table.

Inp Sel, LUT fns for Out2

Figure 9-44. CLB_OUTPUT_LUT_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									FN								IN2					IN1					IN0				
R/W1C-0h									R/W-0h								R/W-0h					R/W-0h					R/W-0h				

Table 9-45. CLB_OUTPUT_LUT_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R/W1C	0h	Reserved
22-15	FN	R/W	0h	Output function for output LUT Reset type: SYSRSn
14-10	IN2	R/W	0h	Select value for IN2 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	IN1	R/W	0h	Select value for IN1 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	IN0	R/W	0h	Select value for IN0 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.24 CLB_OUTPUT_LUT_3 Register (Offset = 32h) [Reset = 0h]

CLB_OUTPUT_LUT_3 is shown in Figure 9-45 and described in Table 9-46.

Return to the Summary Table.

Inp Sel, LUT fns for Out3

Figure 9-45. CLB_OUTPUT_LUT_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									FN								IN2					IN1					IN0				
R/W1C-0h									R/W-0h								R/W-0h					R/W-0h					R/W-0h				

Table 9-46. CLB_OUTPUT_LUT_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R/W1C	0h	Reserved
22-15	FN	R/W	0h	Output function for output LUT Reset type: SYSRSn
14-10	IN2	R/W	0h	Select value for IN2 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	IN1	R/W	0h	Select value for IN1 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	IN0	R/W	0h	Select value for IN0 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.25 CLB_OUTPUT_LUT_4 Register (Offset = 34h) [Reset = 0h]

CLB_OUTPUT_LUT_4 is shown in Figure 9-46 and described in Table 9-47.

Return to the Summary Table.

Inp Sel, LUT fns for Out4

Figure 9-46. CLB_OUTPUT_LUT_4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									FN								IN2					IN1					IN0				
R/W1C-0h									R/W-0h								R/W-0h					R/W-0h					R/W-0h				

Table 9-47. CLB_OUTPUT_LUT_4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R/W1C	0h	Reserved
22-15	FN	R/W	0h	Output function for output LUT Reset type: SYSRSn
14-10	IN2	R/W	0h	Select value for IN2 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	IN1	R/W	0h	Select value for IN1 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	IN0	R/W	0h	Select value for IN0 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.26 CLB_OUTPUT_LUT_5 Register (Offset = 36h) [Reset = 0h]

CLB_OUTPUT_LUT_5 is shown in Figure 9-47 and described in Table 9-48.

Return to the Summary Table.

Inp Sel, LUT fns for Out5

Figure 9-47. CLB_OUTPUT_LUT_5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									FN								IN2					IN1					IN0				
R/W1C-0h									R/W-0h								R/W-0h					R/W-0h					R/W-0h				

Table 9-48. CLB_OUTPUT_LUT_5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R/W1C	0h	Reserved
22-15	FN	R/W	0h	Output function for output LUT Reset type: SYSRSn
14-10	IN2	R/W	0h	Select value for IN2 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	IN1	R/W	0h	Select value for IN1 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	IN0	R/W	0h	Select value for IN0 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.27 CLB_OUTPUT_LUT_6 Register (Offset = 38h) [Reset = 0h]

CLB_OUTPUT_LUT_6 is shown in Figure 9-48 and described in Table 9-49.

Return to the Summary Table.

Inp Sel, LUT fns for Out6

Figure 9-48. CLB_OUTPUT_LUT_6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									FN								IN2					IN1					IN0				
R/W1C-0h									R/W-0h								R/W-0h					R/W-0h					R/W-0h				

Table 9-49. CLB_OUTPUT_LUT_6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R/W1C	0h	Reserved
22-15	FN	R/W	0h	Output function for output LUT Reset type: SYSRSn
14-10	IN2	R/W	0h	Select value for IN2 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	IN1	R/W	0h	Select value for IN1 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	IN0	R/W	0h	Select value for IN0 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.28 CLB_OUTPUT_LUT_7 Register (Offset = 3Ah) [Reset = 0h]

CLB_OUTPUT_LUT_7 is shown in Figure 9-49 and described in Table 9-50.

Return to the Summary Table.

Inp Sel, LUT fns for Out7

Figure 9-49. CLB_OUTPUT_LUT_7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									FN								IN2					IN1					IN0				
R/W1C-0h									R/W-0h								R/W-0h					R/W-0h					R/W-0h				

Table 9-50. CLB_OUTPUT_LUT_7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R/W1C	0h	Reserved
22-15	FN	R/W	0h	Output function for output LUT Reset type: SYSRSn
14-10	IN2	R/W	0h	Select value for IN2 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	IN1	R/W	0h	Select value for IN1 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	IN0	R/W	0h	Select value for IN0 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.29 CLB_HLC_EVENT_SEL Register (Offset = 3Ch) [Reset = 0h]

CLB_HLC_EVENT_SEL is shown in Figure 9-50 and described in Table 9-51.

Return to the Summary Table.

Event Selector register for the High Level controller

Figure 9-50. CLB_HLC_EVENT_SEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
ALT_EVENT3_SEL	ALT_EVENT2_SEL	ALT_EVENT1_SEL	ALT_EVENT0_SEL	EVENT3_SEL			
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h			
15	14	13	12	11	10	9	8
EVENT3_SEL	EVENT2_SEL					EVENT1_SEL	
R/W-0h	R/W-0h					R/W-0h	
7	6	5	4	3	2	1	0
EVENT1_SEL			EVENT0_SEL				
R/W-0h			R/W-0h				

Table 9-51. CLB_HLC_EVENT_SEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R-0	0h	Reserved
23	ALT_EVENT3_SEL	R/W	0h	Defines selection of alternate inputs for EVENT3 Reset type: SYSRSn
22	ALT_EVENT2_SEL	R/W	0h	Defines selection of alternate inputs for EVENT2 Reset type: SYSRSn
21	ALT_EVENT1_SEL	R/W	0h	Defines selection of alternate inputs for EVENT1 Reset type: SYSRSn
20	ALT_EVENT0_SEL	R/W	0h	Defines selection of alternate inputs for EVENT0 Reset type: SYSRSn
19-15	EVENT3_SEL	R/W	0h	5 bit select value for EVENT3 of the High Level Controller. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
14-10	EVENT2_SEL	R/W	0h	5 bit select value for EVENT2 of the High Level Controller. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	EVENT1_SEL	R/W	0h	5 bit select value for EVENT1 of the High Level Controller. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	EVENT0_SEL	R/W	0h	5 bit select value for EVENT0 of the High Level Controller. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

9.10.2.30 CLB_COUNT_MATCH_TAP_SEL Register (Offset = 3Eh) [Reset = 0h]

CLB_COUNT_MATCH_TAP_SEL is shown in Figure 9-51 and described in Table 9-52.

Return to the Summary Table.

Counter tap values for match1 and match2 outputs

Figure 9-51. CLB_COUNT_MATCH_TAP_SEL Register

31	30	29	28	27	26	25	24
RESERVED	COUNT2_MATCH2					COUNT1_MATCH2	
R-0-0h	R/W-0h					R/W-0h	
23	22	21	20	19	18	17	16
COUNT1_MATCH2			COUNT0_MATCH2				
R/W-0h			R/W-0h				
15	14	13	12	11	10	9	8
RESERVED	COUNT2_MATCH1					COUNT1_MATCH1	
R-0-0h	R/W-0h					R/W-0h	
7	6	5	4	3	2	1	0
COUNT1_MATCH1			COUNT0_MATCH1				
R/W-0h			R/W-0h				

Table 9-52. CLB_COUNT_MATCH_TAP_SEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R-0	0h	Reserved
30-26	COUNT2_MATCH2	R/W	0h	5 bit MUX Select for Match2 Tap for Counter Unit 2 Reset type: SYSRSn
25-21	COUNT1_MATCH2	R/W	0h	5 bit MUX Select for Match2 Tap for Counter Unit 1 Reset type: SYSRSn
20-16	COUNT0_MATCH2	R/W	0h	5 bit MUX Select for Match2 Tap for Counter Unit 0 Reset type: SYSRSn
15	RESERVED	R-0	0h	Reserved
14-10	COUNT2_MATCH1	R/W	0h	5 bit MUX Select for Match1 Tap for Counter Unit 2 Reset type: SYSRSn
9-5	COUNT1_MATCH1	R/W	0h	5 bit MUX Select for Match1 Tap for Counter Unit 1 Reset type: SYSRSn
4-0	COUNT0_MATCH1	R/W	0h	5 bit MUX Select for Match1 Tap for Counter Unit 0 Reset type: SYSRSn

9.10.2.31 CLB_OUTPUT_COND_CTRL_0 Register (Offset = 40h) [Reset = 0h]

CLB_OUTPUT_COND_CTRL_0 is shown in Figure 9-52 and described in Table 9-53.

Return to the Summary Table.

Output conditioning control for output 0

Figure 9-52. CLB_OUTPUT_COND_CTRL_0 Register

31	30	29	28	27	26	25	24
RESERVED							
R/W1C-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W1C-0h							
15	14	13	12	11	10	9	8
RESERVED	ASYNC_COND_EN	SEL_RAW_IN	HW_RLS_CTRL_SEL	HW_GATING_CTRL_SEL	SEL_RELEASE_CTRL		
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h		
7	6	5	4	3	2	1	0
SEL_GATING_CTRL			LEVEL_3_SEL		LEVEL_2_SEL		LEVEL_1_SEL
R/W1C-0h			R/W1C-0h		R/W1C-0h		R/W1C-0h

Table 9-53. CLB_OUTPUT_COND_CTRL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14	ASYNC_COND_EN	R/W1C	0h	Controls whether the output will pass through the asynchronous conditioning block or bypass it. 0 Bypass the asynchronous conditioning block 1 Enable the asynchronous conditioning path Reset type: SYSRSn
13	SEL_RAW_IN	R/W1C	0h	Controls whether the CELL outputs or inputs are sent to the output conditioning block logic. 0 = CELL output (internally delayed by 1 cycle) is used. 1 = CELL input (raw) is used. Reset type: SYSRSn
12	HW_RLS_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the release control 0 SW register value will act as release control 1 Selected CELL output will act as release control Reset type: SYSRSn
11	HW_GATING_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the gating control 0 SW register value will act as gating control 1 Selected CELL output will act as gating control Reset type: SYSRSn
10-8	SEL_RELEASE_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Release control. Reset type: SYSRSn
7-5	SEL_GATING_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Gating control. Reset type: SYSRSn

Table 9-53. CLB_OUTPUT_COND_CTRL_0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	LEVEL_3_SEL	R/W1C	0h	Controls Third level Mux select 00 Input Signal will be sent as is to the output 01 Rising edge of Input signal will cause asynchronous CLEAR of the output 10 Rising edge of Input signal will cause asynchronous SET of the output 11 Input Signal delayed by 1 clock cycle will be sent to the output Reset type: SYSRSn
2-1	LEVEL_2_SEL	R/W1C	0h	Controls Second level Mux select 00 Input Signal sent as output to next level 01 Input Signal AND Gating_control sent as output to next level 10 Input Signal OR Gating_control sent as output to next level 11 Input Signal XOR Gating_control sent as output to next level Reset type: SYSRSn
0	LEVEL_1_SEL	R/W1C	0h	First level MUX select value 0 Direct signal sent as output to next level 1 Inverted signal sent as output to the next level Reset type: SYSRSn

9.10.2.32 CLB_OUTPUT_COND_CTRL_1 Register (Offset = 42h) [Reset = 0h]

CLB_OUTPUT_COND_CTRL_1 is shown in Figure 9-53 and described in Table 9-54.

Return to the Summary Table.

Output conditioning control for output 1

Figure 9-53. CLB_OUTPUT_COND_CTRL_1 Register

31	30	29	28	27	26	25	24
RESERVED							
R/W1C-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W1C-0h							
15	14	13	12	11	10	9	8
RESERVED	ASYNC_COND_EN	SEL_RAW_IN	HW_RLS_CTRL_SEL	HW_GATING_CTRL_SEL	SEL_RELEASE_CTRL		
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h		
7	6	5	4	3	2	1	0
SEL_GATING_CTRL			LEVEL_3_SEL		LEVEL_2_SEL		LEVEL_1_SEL
R/W1C-0h			R/W1C-0h		R/W1C-0h		R/W1C-0h

Table 9-54. CLB_OUTPUT_COND_CTRL_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14	ASYNC_COND_EN	R/W1C	0h	Controls whether the output will pass through the asynchronous conditioning block or bypass it. 0 Bypass the asynchronous conditioning block 1 Enable the asynchronous conditioning path Reset type: SYSRSn
13	SEL_RAW_IN	R/W1C	0h	Controls whether the CELL outputs or inputs are sent to the output conditioning block logic. 0 = CELL output (internally delayed by 1 cycle) is used. 1 = CELL input (raw) is used. Reset type: SYSRSn
12	HW_RLS_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the release control 0 SW register value will act as release control 1 Selected CELL output will act as release control Reset type: SYSRSn
11	HW_GATING_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the gating control 0 SW register value will act as gating control 1 Selected CELL output will act as gating control Reset type: SYSRSn
10-8	SEL_RELEASE_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Release control. Reset type: SYSRSn
7-5	SEL_GATING_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Gating control. Reset type: SYSRSn

Table 9-54. CLB_OUTPUT_COND_CTRL_1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	LEVEL_3_SEL	R/W1C	0h	Controls Third level Mux select 00 Input Signal will be sent as is to the output 01 Rising edge of Input signal will cause asynchronous CLEAR of the output 10 Rising edge of Input signal will cause asynchronous SET of the output 11 Input Signal delayed by 1 clock cycle will be sent to the output Reset type: SYSRSn
2-1	LEVEL_2_SEL	R/W1C	0h	Controls Second level Mux select 00 Input Signal sent as output to next level 01 Input Signal AND Gating_control sent as output to next level 10 Input Signal OR Gating_control sent as output to next level 11 Input Signal XOR Gating_control sent as output to next level Reset type: SYSRSn
0	LEVEL_1_SEL	R/W1C	0h	First level MUX select value 0 Direct signal sent as output to next level 1 Inverted signal sent as output to the next level Reset type: SYSRSn

9.10.2.33 CLB_OUTPUT_COND_CTRL_2 Register (Offset = 44h) [Reset = 0h]

CLB_OUTPUT_COND_CTRL_2 is shown in Figure 9-54 and described in Table 9-55.

Return to the Summary Table.

Output conditioning control for output 2

Figure 9-54. CLB_OUTPUT_COND_CTRL_2 Register

31	30	29	28	27	26	25	24
RESERVED							
R/W1C-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W1C-0h							
15	14	13	12	11	10	9	8
RESERVED	ASYNC_COND_EN	SEL_RAW_IN	HW_RLS_CTRL_SEL	HW_GATING_CTRL_SEL	SEL_RELEASE_CTRL		
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h		
7	6	5	4	3	2	1	0
SEL_GATING_CTRL			LEVEL_3_SEL		LEVEL_2_SEL		LEVEL_1_SEL
R/W1C-0h			R/W1C-0h		R/W1C-0h		R/W1C-0h

Table 9-55. CLB_OUTPUT_COND_CTRL_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14	ASYNC_COND_EN	R/W1C	0h	Controls whether the output will pass through the asynchronous conditioning block or bypass it. 0 Bypass the asynchronous conditioning block 1 Enable the asynchronous conditioning path Reset type: SYSRSn
13	SEL_RAW_IN	R/W1C	0h	Controls whether the CELL outputs or inputs are sent to the output conditioning block logic. 0 = CELL output (internally delayed by 1 cycle) is used. 1 = CELL input (raw) is used. Reset type: SYSRSn
12	HW_RLS_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the release control 0 SW register value will act as release control 1 Selected CELL output will act as release control Reset type: SYSRSn
11	HW_GATING_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the gating control 0 SW register value will act as gating control 1 Selected CELL output will act as gating control Reset type: SYSRSn
10-8	SEL_RELEASE_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Release control. Reset type: SYSRSn
7-5	SEL_GATING_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Gating control. Reset type: SYSRSn

Table 9-55. CLB_OUTPUT_COND_CTRL_2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	LEVEL_3_SEL	R/W1C	0h	Controls Third level Mux select 00 Input Signal will be sent as is to the output 01 Rising edge of Input signal will cause asynchronous CLEAR of the output 10 Rising edge of Input signal will cause asynchronous SET of the output 11 Input Signal delayed by 1 clock cycle will be sent to the output Reset type: SYSRSn
2-1	LEVEL_2_SEL	R/W1C	0h	Controls Second level Mux select 00 Input Signal sent as output to next level 01 Input Signal AND Gating_control sent as output to next level 10 Input Signal OR Gating_control sent as output to next level 11 Input Signal XOR Gating_control sent as output to next level Reset type: SYSRSn
0	LEVEL_1_SEL	R/W1C	0h	First level MUX select value 0 Direct signal sent as output to next level 1 Inverted signal sent as output to the next level Reset type: SYSRSn

9.10.2.34 CLB_OUTPUT_COND_CTRL_3 Register (Offset = 46h) [Reset = 0h]

CLB_OUTPUT_COND_CTRL_3 is shown in Figure 9-55 and described in Table 9-56.

Return to the Summary Table.

Output conditioning control for output 3

Figure 9-55. CLB_OUTPUT_COND_CTRL_3 Register

31	30	29	28	27	26	25	24
RESERVED							
R/W1C-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W1C-0h							
15	14	13	12	11	10	9	8
RESERVED	ASYNC_COND_EN	SEL_RAW_IN	HW_RLS_CTRL_SEL	HW_GATING_CTRL_SEL	SEL_RELEASE_CTRL		
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h		
7	6	5	4	3	2	1	0
SEL_GATING_CTRL			LEVEL_3_SEL		LEVEL_2_SEL		LEVEL_1_SEL
R/W1C-0h			R/W1C-0h		R/W1C-0h		R/W1C-0h

Table 9-56. CLB_OUTPUT_COND_CTRL_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14	ASYNC_COND_EN	R/W1C	0h	Controls whether the output will pass through the asynchronous conditioning block or bypass it. 0 Bypass the asynchronous conditioning block 1 Enable the asynchronous conditioning path Reset type: SYSRSn
13	SEL_RAW_IN	R/W1C	0h	Controls whether the CELL outputs or inputs are sent to the output conditioning block logic. 0 = CELL output (internally delayed by 1 cycle) is used. 1 = CELL input (raw) is used. Reset type: SYSRSn
12	HW_RLS_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the release control 0 SW register value will act as release control 1 Selected CELL output will act as release control Reset type: SYSRSn
11	HW_GATING_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the gating control 0 SW register value will act as gating control 1 Selected CELL output will act as gating control Reset type: SYSRSn
10-8	SEL_RELEASE_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Release control. Reset type: SYSRSn
7-5	SEL_GATING_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Gating control. Reset type: SYSRSn

Table 9-56. CLB_OUTPUT_COND_CTRL_3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	LEVEL_3_SEL	R/W1C	0h	Controls Third level Mux select 00 Input Signal will be sent as is to the output 01 Rising edge of Input signal will cause asynchronous CLEAR of the output 10 Rising edge of Input signal will cause asynchronous SET of the output 11 Input Signal delayed by 1 clock cycle will be sent to the output Reset type: SYSRSn
2-1	LEVEL_2_SEL	R/W1C	0h	Controls Second level Mux select 00 Input Signal sent as output to next level 01 Input Signal AND Gating_control sent as output to next level 10 Input Signal OR Gating_control sent as output to next level 11 Input Signal XOR Gating_control sent as output to next level Reset type: SYSRSn
0	LEVEL_1_SEL	R/W1C	0h	First level MUX select value 0 Direct signal sent as output to next level 1 Inverted signal sent as output to the next level Reset type: SYSRSn

9.10.2.35 CLB_OUTPUT_COND_CTRL_4 Register (Offset = 48h) [Reset = 0h]

CLB_OUTPUT_COND_CTRL_4 is shown in Figure 9-56 and described in Table 9-57.

Return to the Summary Table.

Output conditioning control for output 4

Figure 9-56. CLB_OUTPUT_COND_CTRL_4 Register

31	30	29	28	27	26	25	24
RESERVED							
R/W1C-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W1C-0h							
15	14	13	12	11	10	9	8
RESERVED	ASYNC_COND_EN	SEL_RAW_IN	HW_RLS_CTRL_SEL	HW_GATING_CTRL_SEL	SEL_RELEASE_CTRL		
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h		
7	6	5	4	3	2	1	0
SEL_GATING_CTRL			LEVEL_3_SEL		LEVEL_2_SEL		LEVEL_1_SEL
R/W1C-0h			R/W1C-0h		R/W1C-0h		R/W1C-0h

Table 9-57. CLB_OUTPUT_COND_CTRL_4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14	ASYNC_COND_EN	R/W1C	0h	Controls whether the output will pass through the asynchronous conditioning block or bypass it. 0 Bypass the asynchronous conditioning block 1 Enable the asynchronous conditioning path Reset type: SYSRSn
13	SEL_RAW_IN	R/W1C	0h	Controls whether the CELL outputs or inputs are sent to the output conditioning block logic. 0 = CELL output (internally delayed by 1 cycle) is used. 1 = CELL input (raw) is used. Reset type: SYSRSn
12	HW_RLS_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the release control 0 SW register value will act as release control 1 Selected CELL output will act as release control Reset type: SYSRSn
11	HW_GATING_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the gating control 0 SW register value will act as gating control 1 Selected CELL output will act as gating control Reset type: SYSRSn
10-8	SEL_RELEASE_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Release control. Reset type: SYSRSn
7-5	SEL_GATING_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Gating control. Reset type: SYSRSn

Table 9-57. CLB_OUTPUT_COND_CTRL_4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	LEVEL_3_SEL	R/W1C	0h	Controls Third level Mux select 00 Input Signal will be sent as is to the output 01 Rising edge of Input signal will cause asynchronous CLEAR of the output 10 Rising edge of Input signal will cause asynchronous SET of the output 11 Input Signal delayed by 1 clock cycle will be sent to the output Reset type: SYSRSn
2-1	LEVEL_2_SEL	R/W1C	0h	Controls Second level Mux select 00 Input Signal sent as output to next level 01 Input Signal AND Gating_control sent as output to next level 10 Input Signal OR Gating_control sent as output to next level 11 Input Signal XOR Gating_control sent as output to next level Reset type: SYSRSn
0	LEVEL_1_SEL	R/W1C	0h	First level MUX select value 0 Direct signal sent as output to next level 1 Inverted signal sent as output to the next level Reset type: SYSRSn

9.10.2.36 CLB_OUTPUT_COND_CTRL_5 Register (Offset = 4Ah) [Reset = 0h]

CLB_OUTPUT_COND_CTRL_5 is shown in Figure 9-57 and described in Table 9-58.

Return to the Summary Table.

Output conditioning control for output 5

Figure 9-57. CLB_OUTPUT_COND_CTRL_5 Register

31	30	29	28	27	26	25	24
RESERVED							
R/W1C-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W1C-0h							
15	14	13	12	11	10	9	8
RESERVED	ASYNC_COND_EN	SEL_RAW_IN	HW_RLS_CTRL_SEL	HW_GATING_CTRL_SEL	SEL_RELEASE_CTRL		
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h		
7	6	5	4	3	2	1	0
SEL_GATING_CTRL			LEVEL_3_SEL		LEVEL_2_SEL		LEVEL_1_SEL
R/W1C-0h			R/W1C-0h		R/W1C-0h		R/W1C-0h

Table 9-58. CLB_OUTPUT_COND_CTRL_5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14	ASYNC_COND_EN	R/W1C	0h	Controls whether the output will pass through the asynchronous conditioning block or bypass it. 0 Bypass the asynchronous conditioning block 1 Enable the asynchronous conditioning path Reset type: SYSRSn
13	SEL_RAW_IN	R/W1C	0h	Controls whether the CELL outputs or inputs are sent to the output conditioning block logic. 0 = CELL output (internally delayed by 1 cycle) is used. 1 = CELL input (raw) is used. Reset type: SYSRSn
12	HW_RLS_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the release control 0 SW register value will act as release control 1 Selected CELL output will act as release control Reset type: SYSRSn
11	HW_GATING_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the gating control 0 SW register value will act as gating control 1 Selected CELL output will act as gating control Reset type: SYSRSn
10-8	SEL_RELEASE_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Release control. Reset type: SYSRSn
7-5	SEL_GATING_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Gating control. Reset type: SYSRSn

Table 9-58. CLB_OUTPUT_COND_CTRL_5 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	LEVEL_3_SEL	R/W1C	0h	Controls Third level Mux select 00 Input Signal will be sent as is to the output 01 Rising edge of Input signal will cause asynchronous CLEAR of the output 10 Rising edge of Input signal will cause asynchronous SET of the output 11 Input Signal delayed by 1 clock cycle will be sent to the output Reset type: SYSRSn
2-1	LEVEL_2_SEL	R/W1C	0h	Controls Second level Mux select 00 Input Signal sent as output to next level 01 Input Signal AND Gating_control sent as output to next level 10 Input Signal OR Gating_control sent as output to next level 11 Input Signal XOR Gating_control sent as output to next level Reset type: SYSRSn
0	LEVEL_1_SEL	R/W1C	0h	First level MUX select value 0 Direct signal sent as output to next level 1 Inverted signal sent as output to the next level Reset type: SYSRSn

9.10.2.37 CLB_OUTPUT_COND_CTRL_6 Register (Offset = 4Ch) [Reset = 0h]

CLB_OUTPUT_COND_CTRL_6 is shown in Figure 9-58 and described in Table 9-59.

Return to the Summary Table.

Output conditioning control for output 6

Figure 9-58. CLB_OUTPUT_COND_CTRL_6 Register

31	30	29	28	27	26	25	24
RESERVED							
R/W1C-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W1C-0h							
15	14	13	12	11	10	9	8
RESERVED	ASYNC_COND_EN	SEL_RAW_IN	HW_RLS_CTRL_SEL	HW_GATING_CTRL_SEL	SEL_RELEASE_CTRL		
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h		
7	6	5	4	3	2	1	0
SEL_GATING_CTRL			LEVEL_3_SEL		LEVEL_2_SEL		LEVEL_1_SEL
R/W1C-0h			R/W1C-0h		R/W1C-0h		R/W1C-0h

Table 9-59. CLB_OUTPUT_COND_CTRL_6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14	ASYNC_COND_EN	R/W1C	0h	Controls whether the output will pass through the asynchronous conditioning block or bypass it. 0 Bypass the asynchronous conditioning block 1 Enable the asynchronous conditioning path Reset type: SYSRSn
13	SEL_RAW_IN	R/W1C	0h	Controls whether the CELL outputs or inputs are sent to the output conditioning block logic. 0 = CELL output (internally delayed by 1 cycle) is used. 1 = CELL input (raw) is used. Reset type: SYSRSn
12	HW_RLS_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the release control 0 SW register value will act as release control 1 Selected CELL output will act as release control Reset type: SYSRSn
11	HW_GATING_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the gating control 0 SW register value will act as gating control 1 Selected CELL output will act as gating control Reset type: SYSRSn
10-8	SEL_RELEASE_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Release control. Reset type: SYSRSn
7-5	SEL_GATING_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Gating control. Reset type: SYSRSn

Table 9-59. CLB_OUTPUT_COND_CTRL_6 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	LEVEL_3_SEL	R/W1C	0h	Controls Third level Mux select 00 Input Signal will be sent as is to the output 01 Rising edge of Input signal will cause asynchronous CLEAR of the output 10 Rising edge of Input signal will cause asynchronous SET of the output 11 Input Signal delayed by 1 clock cycle will be sent to the output Reset type: SYSRSn
2-1	LEVEL_2_SEL	R/W1C	0h	Controls Second level Mux select 00 Input Signal sent as output to next level 01 Input Signal AND Gating_control sent as output to next level 10 Input Signal OR Gating_control sent as output to next level 11 Input Signal XOR Gating_control sent as output to next level Reset type: SYSRSn
0	LEVEL_1_SEL	R/W1C	0h	First level MUX select value 0 Direct signal sent as output to next level 1 Inverted signal sent as output to the next level Reset type: SYSRSn

9.10.2.38 CLB_OUTPUT_COND_CTRL_7 Register (Offset = 4Eh) [Reset = 0h]

CLB_OUTPUT_COND_CTRL_7 is shown in Figure 9-59 and described in Table 9-60.

Return to the Summary Table.

Output conditioning control for output 7

Figure 9-59. CLB_OUTPUT_COND_CTRL_7 Register

31	30	29	28	27	26	25	24
RESERVED							
R/W1C-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W1C-0h							
15	14	13	12	11	10	9	8
RESERVED	ASYNC_COND_EN	SEL_RAW_IN	HW_RLS_CTRL_SEL	HW_GATING_CTRL_SEL	SEL_RELEASE_CTRL		
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h		
7	6	5	4	3	2	1	0
SEL_GATING_CTRL			LEVEL_3_SEL		LEVEL_2_SEL		LEVEL_1_SEL
R/W1C-0h			R/W1C-0h		R/W1C-0h		R/W1C-0h

Table 9-60. CLB_OUTPUT_COND_CTRL_7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W1C	0h	Reserved
14	ASYNC_COND_EN	R/W1C	0h	Controls whether the output will pass through the asynchronous conditioning block or bypass it. 0 Bypass the asynchronous conditioning block 1 Enable the asynchronous conditioning path Reset type: SYSRSn
13	SEL_RAW_IN	R/W1C	0h	Controls whether the CELL outputs or inputs are sent to the output conditioning block logic. 0 = CELL output (internally delayed by 1 cycle) is used. 1 = CELL input (raw) is used. Reset type: SYSRSn
12	HW_RLS_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the release control 0 SW register value will act as release control 1 Selected CELL output will act as release control Reset type: SYSRSn
11	HW_GATING_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the gating control 0 SW register value will act as gating control 1 Selected CELL output will act as gating control Reset type: SYSRSn
10-8	SEL_RELEASE_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Release control. Reset type: SYSRSn
7-5	SEL_GATING_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Gating control. Reset type: SYSRSn

Table 9-60. CLB_OUTPUT_COND_CTRL_7 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	LEVEL_3_SEL	R/W1C	0h	Controls Third level Mux select 00 Input Signal will be sent as is to the output 01 Rising edge of Input signal will cause asynchronous CLEAR of the output 10 Rising edge of Input signal will cause asynchronous SET of the output 11 Input Signal delayed by 1 clock cycle will be sent to the output Reset type: SYSRSn
2-1	LEVEL_2_SEL	R/W1C	0h	Controls Second level Mux select 00 Input Signal sent as output to next level 01 Input Signal AND Gating_control sent as output to next level 10 Input Signal OR Gating_control sent as output to next level 11 Input Signal XOR Gating_control sent as output to next level Reset type: SYSRSn
0	LEVEL_1_SEL	R/W1C	0h	First level MUX select value 0 Direct signal sent as output to next level 1 Inverted signal sent as output to the next level Reset type: SYSRSn

9.10.2.39 CLB_MISC_ACCESS_CTRL Register (Offset = 50h) [Reset = 0h]

CLB_MISC_ACCESS_CTRL is shown in Figure 9-60 and described in Table 9-61.

Return to the Summary Table.

Miscellaneous Access and enable control

Figure 9-60. CLB_MISC_ACCESS_CTRL Register

15	14	13	12	11	10	9	8
RESERVED							
R/W1C-0h							
7	6	5	4	3	2	1	0
RESERVED						BLKEN	SPIEN
R/W1C-0h						R/W1C-0h	R/W1C-0h

Table 9-61. CLB_MISC_ACCESS_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R/W1C	0h	Reserved
1	BLKEN	R/W1C	0h	This bit is used to block writes to CLB_OUT_EN 0 Writes to CLB_OUT_EN are allowed 1 Writes to CLB_OUT_EN are blocked Reset type: SYSRSn
0	SPIEN	R/W1C	0h	This bit indicates the status of the SPI buffers ability to export CLB output data. 0 Feature Disabled 1 Feature Enabled Reset type: SYSRSn

9.10.2.40 CLB_SPI_DATA_CTRL_HI Register (Offset = 51h) [Reset = 0h]

CLB_SPI_DATA_CTRL_HI is shown in Figure 9-61 and described in Table 9-62.

Return to the Summary Table.

CLB to SPI buffer control High

Figure 9-61. CLB_SPI_DATA_CTRL_HI Register

15	14	13	12	11	10	9	8
RESERVED			SHIFT				
R/W1C-0h			R/W1C-0h				
7	6	5	4	3	2	1	0
RESERVED			STRB				
R/W1C-0h			R/W1C-0h				

Table 9-62. CLB_SPI_DATA_CTRL_HI Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R/W1C	0h	Reserved
12-8	SHIFT	R/W1C	0h	This is a 5 bit value which denotes the first bit position of register R0 from which the Least Significant Bit of the output data should start. 00000 Output data is R0[15:0] 00001 Output data is R0[16:1] 00010 Output data is R0[17:2] 00011 Output data is R0[18:3] ... ... 10000 Output data is R0[31:16] Reset type: SYSRSn
7-5	RESERVED	R/W1C	0h	Reserved
4-0	STRB	R/W1C	0h	This is a 5 bit value which selects one of the HLC_EVENT inputs to be treated as the data_valid strobe Reset type: SYSRSn

9.10.3 CLB_LOGIC_CONTROL_REGS Registers

Table 9-63 lists the memory-mapped registers for the CLB_LOGIC_CONTROL_REGS registers. All register offset addresses not listed in Table 9-63 should be considered as reserved locations and the register contents should not be modified.

Table 9-63. CLB_LOGIC_CONTROL_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CLB_LOAD_EN	Global enable & indirect load enable control	LOCK (only Global Enable Bit is LOCK protected)	Go
2h	CLB_LOAD_ADDR	Indirect address		Go
4h	CLB_LOAD_DATA	Data for indirect loads		Go
6h	CLB_INPUT_FILTER	Input filter selection for both edge detection and synchronizers	LOCK	Go
8h	CLB_IN_MUX_SEL_0	Input selection to decide between Signals and GP register	LOCK	Go
Ah	CLB_LCL_MUX_SEL_1	Input Mux selection for local mux	LOCK	Go
Ch	CLB_LCL_MUX_SEL_2	Input Mux selection for local mux	LOCK	Go
Eh	CLB_BUF_PTR	PUSH and PULL pointers		Go
10h	CLB_GP_REG	General purpose register for CELL inputs		Go
12h	CLB_OUT_EN	CELL output enable register		Go
14h	CLB_GLBL_MUX_SEL_1	Global Mux select for CELL inputs	LOCK	Go
16h	CLB_GLBL_MUX_SEL_2	Global Mux select for CELL inputs	LOCK	Go
18h	CLB_PRESCALE_CTRL	Prescaler register control	LOCK	Go
20h	CLB_INTR_TAG_REG	Interrupt Tag register		Go
22h	CLB_LOCK	Lock control register	EALLOW	Go
24h	CLB_HLC_INSTR_READ_PTR	HLC instruction read pointer		Go
26h	CLB_HLC_INSTR_VALUE	HLC instruction read value		Go
2Eh	CLB_DBG_OUT_2	Visibility for CLB inputs and final asynchronous outputs		Go
30h	CLB_DBG_R0	R0 of High level Controller		Go
32h	CLB_DBG_R1	R1 of High level Controller		Go
34h	CLB_DBG_R2	R2 of High level Controller		Go
36h	CLB_DBG_R3	R3 of High level Controller		Go
38h	CLB_DBG_C0	Count of Unit 0		Go
3Ah	CLB_DBG_C1	Count of Unit 1		Go
3Ch	CLB_DBG_C2	Count of Unit 2		Go
3Eh	CLB_DBG_OUT	Outputs of various units in the Cell		Go

Complex bit access types are encoded to fit into small table cells. Table 9-64 shows the codes that are used for access types in this section.

Table 9-64. CLB_LOGIC_CONTROL_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s

Table 9-64. CLB_LOGIC_CONTROL_REGS Access Type Codes (continued)

Access Type	Code	Description
R-1	R -1	Read Returns 1s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

9.10.3.1 CLB_LOAD_EN Register (Offset = 0h) [Reset = 0h]

CLB_LOAD_EN is shown in Figure 9-62 and described in Table 9-65.

Return to the Summary Table.

Global enable & indirect load enable control

Figure 9-62. CLB_LOAD_EN Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED			PIPELINE_EN	NMI_EN	STOP	GLOBAL_EN	LOAD_EN
R-0-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 9-65. CLB_LOAD_EN Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R-0	0h	Reserved
4	PIPELINE_EN	R/W	0h	This bit controls pipelining of all the CLB operations in HLC and Counter blocks. Pipelined operation is enabled when this bit is set to 1. Reset type: SYSRSn
3	NMI_EN	R/W	0h	This bit controls the generation of NMI along with the interrupt whenever a INTR operation is executed by the HLC. NMI generation is disabled by default. It will be enabled when this bit is set to 1. Reset type: SYSRSn
2	STOP	R/W	0h	This bit defines the behaviour of the sequential elements in the CELL during debug HALTs of the CPU. If this bit is set to 0, the debug HALT condition is ignored. Reset type: SYSRSn
1	GLOBAL_EN	R/W	0h	This bit is a global enable signal for the logic in the CELL. This also acts as a soft reset for the CELL logic. CLB outputs (including LUTs and OUTLUTs) will be gated when this bit is cleared from 1 to 0, i.e., the CLB outputs will be low when GLOBAL_EN is low. Additionally, the FSM and AOC blocks will also be reset. Note that when this bit goes low, the COUNTER blocks and HLC are simply halted, but they will NOT be reset internally. This allows the ability to preload these submodules when GLOBAL_EN is 0. This bit is normally set after all the other configuration settings are completed. This bit is LOCK protected. Reset type: SYSRSn
0	LOAD_EN	R/W	0h	A write with this bit set to 1 will pulse the Load Enable signal for the indirect register loads in the CELL. Reset type: SYSRSn

9.10.3.2 CLB_LOAD_ADDR Register (Offset = 2h) [Reset = 0h]

CLB_LOAD_ADDR is shown in Figure 9-63 and described in Table 9-66.

Return to the Summary Table.

Indirect address

Figure 9-63. CLB_LOAD_ADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										ADDR					
R-0-0h																										R/W-0h					

Table 9-66. CLB_LOAD_ADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-0	ADDR	R/W	0h	These are the address bits used for writing to the indirect address space of the CELL. Reset type: SYSRSn

9.10.3.3 CLB_LOAD_DATA Register (Offset = 4h) [Reset = 0h]

CLB_LOAD_DATA is shown in Figure 9-64 and described in Table 9-67.

Return to the Summary Table.

Data for indirect loads

Figure 9-64. CLB_LOAD_DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 9-67. CLB_LOAD_DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	This register holds the 32-bit data for writing to the indirect address space of the CELL. Reset type: SYSRSn

9.10.3.4 CLB_INPUT_FILTER Register (Offset = 6h) [Reset = 0h]

CLB_INPUT_FILTER is shown in Figure 9-65 and described in Table 9-68.

Return to the Summary Table.

Input filter selection for both edge detection and synchronizers

Figure 9-65. CLB_INPUT_FILTER Register

31	30	29	28	27	26	25	24
PIPE7	PIPE6	PIPE5	PIPE4	PIPE3	PIPE2	PIPE1	PIPE0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
SYNC7	SYNC6	SYNC5	SYNC4	SYNC3	SYNC2	SYNC1	SYNC0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
FIN7		FIN6		FIN5		FIN4	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
FIN3		FIN2		FIN1		FIN0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 9-68. CLB_INPUT_FILTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31	PIPE7	R/W	0h	Enable pipelining for Input 7 Reset type: SYSRSn
30	PIPE6	R/W	0h	Enable pipelining for Input 6 Reset type: SYSRSn
29	PIPE5	R/W	0h	Enable pipelining for Input 5 Reset type: SYSRSn
28	PIPE4	R/W	0h	Enable pipelining for Input 4 Reset type: SYSRSn
27	PIPE3	R/W	0h	Enable pipelining for Input 3 Reset type: SYSRSn
26	PIPE2	R/W	0h	Enable pipelining for Input 2 Reset type: SYSRSn
25	PIPE1	R/W	0h	Enable pipelining for Input 1 Reset type: SYSRSn
24	PIPE0	R/W	0h	Enable pipelining for Input 0 Reset type: SYSRSn
23	SYNC7	R/W	0h	Synchronizer Select Control for Input 7 Reset type: SYSRSn
22	SYNC6	R/W	0h	Synchronizer Select Control for Input 6 Reset type: SYSRSn
21	SYNC5	R/W	0h	Synchronizer Select Control for Input 5 Reset type: SYSRSn
20	SYNC4	R/W	0h	Synchronizer Select Control for Input 4 Reset type: SYSRSn
19	SYNC3	R/W	0h	Synchronizer Select Control for Input 3 Reset type: SYSRSn
18	SYNC2	R/W	0h	Synchronizer Select Control for Input 2 Reset type: SYSRSn
17	SYNC1	R/W	0h	Synchronizer Select Control for Input 1 Reset type: SYSRSn

Table 9-68. CLB_INPUT_FILTER Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	SYNC0	R/W	0h	Synchronizer Select Control for Input 0 Reset type: SYSRSn
15-14	FIN7	R/W	0h	Input filter selection for CELL Input 7 2 bits are used to define the edge filtering . 00 : No filtering 01 : Rising edge detect 10 : Falling edge detect 11 : Any edge detect Reset type: SYSRSn
13-12	FIN6	R/W	0h	Input filter selection for CELL Input 6 2 bits are used to define the edge filtering . 00 : No filtering 01 : Rising edge detect 10 : Falling edge detect 11 : Any edge detect Reset type: SYSRSn
11-10	FIN5	R/W	0h	Input filter selection for CELL Input 5 2 bits are used to define the edge filtering . 00 : No filtering 01 : Rising edge detect 10 : Falling edge detect 11 : Any edge detect Reset type: SYSRSn
9-8	FIN4	R/W	0h	Input filter selection for CELL Input 4 2 bits are used to define the edge filtering . 00 : No filtering 01 : Rising edge detect 10 : Falling edge detect 11 : Any edge detect Reset type: SYSRSn
7-6	FIN3	R/W	0h	Input filter selection for CELL Input 3 2 bits are used to define the edge filtering . 00 : No filtering 01 : Rising edge detect 10 : Falling edge detect 11 : Any edge detect Reset type: SYSRSn
5-4	FIN2	R/W	0h	Input filter selection for CELL Input 2 2 bits are used to define the edge filtering . 00 : No filtering 01 : Rising edge detect 10 : Falling edge detect 11 : Any edge detect Reset type: SYSRSn
3-2	FIN1	R/W	0h	Input filter selection for CELL Input 1 2 bits are used to define the edge filtering . 00 : No filtering 01 : Rising edge detect 10 : Falling edge detect 11 : Any edge detect Reset type: SYSRSn
1-0	FIN0	R/W	0h	Input filter selection for CELL Input 0 2 bits are used to define the edge filtering . 00 : No filtering 01 : Rising edge detect 10 : Falling edge detect 11 : Any edge detect Reset type: SYSRSn

9.10.3.5 CLB_IN_MUX_SEL_0 Register (Offset = 8h) [Reset = 0h]

CLB_IN_MUX_SEL_0 is shown in Figure 9-66 and described in Table 9-69.

Return to the Summary Table.

Input selection to decide between Signals and GP register

Figure 9-66. CLB_IN_MUX_SEL_0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
SEL_GP_IN_7	SEL_GP_IN_6	SEL_GP_IN_5	SEL_GP_IN_4	SEL_GP_IN_3	SEL_GP_IN_2	SEL_GP_IN_1	SEL_GP_IN_0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 9-69. CLB_IN_MUX_SEL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R-0	0h	Reserved
7	SEL_GP_IN_7	R/W	0h	Select control for Input 7 to decide between external input and CLB_GP_REG[7] 0 : Input comes from selected external input 1 : Input comes from CLB_GP_REG[7] Reset type: SYSRSn
6	SEL_GP_IN_6	R/W	0h	Select control for Input 6 to decide between external input and CLB_GP_REG[6] 0 : Input comes from selected external input 1 : Input comes from CLB_GP_REG[6] Reset type: SYSRSn
5	SEL_GP_IN_5	R/W	0h	Select control for Input 5 to decide between external input and CLB_GP_REG[5] 0 : Input comes from selected external input 1 : Input comes from CLB_GP_REG[5] Reset type: SYSRSn
4	SEL_GP_IN_4	R/W	0h	Select control for Input 4 to decide between external input and CLB_GP_REG[4] 0 : Input comes from selected external input 1 : Input comes from CLB_GP_REG[4] Reset type: SYSRSn
3	SEL_GP_IN_3	R/W	0h	Select control for Input 3 to decide between external input and CLB_GP_REG[3] 0 : Input comes from selected external input 1 : Input comes from CLB_GP_REG[3] Reset type: SYSRSn
2	SEL_GP_IN_2	R/W	0h	Select control for Input 2 to decide between external input and CLB_GP_REG[2] 0 : Input comes from selected external input 1 : Input comes from CLB_GP_REG[2] Reset type: SYSRSn

Table 9-69. CLB_IN_MUX_SEL_0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	SEL_GP_IN_1	R/W	0h	Select control for Input 1 to decide between external input and CLB_GP_REG[1] 0 : Input comes from selected external input 1 : Input comes from CLB_GP_REG[1] Reset type: SYSRSn
0	SEL_GP_IN_0	R/W	0h	Select control for Input 0 to decide between external input and CLB_GP_REG[0] 0 : Input comes from selected external input 1 : Input comes from CLB_GP_REG[0] Reset type: SYSRSn

9.10.3.6 CLB_LCL_MUX_SEL_1 Register (Offset = Ah) [Reset = 0h]

CLB_LCL_MUX_SEL_1 is shown in Figure 9-67 and described in Table 9-70.

Return to the Summary Table.

Input Mux selection for local mux

Figure 9-67. CLB_LCL_MUX_SEL_1 Register

31	30	29	28	27	26	25	24
MISC_INPUT_SEL_3	MISC_INPUT_SEL_2	MISC_INPUT_SEL_1	MISC_INPUT_SEL_0	RESERVED			
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0-0h			
23	22	21	20	19	18	17	16
RESERVED				LCL_MUX_SEL_IN_3			
R-0-0h				R/W-0h			
15	14	13	12	11	10	9	8
LCL_MUX_SEL_IN_3	LCL_MUX_SEL_IN_2					LCL_MUX_SEL_IN_1	
R/W-0h	R/W-0h					R/W-0h	
7	6	5	4	3	2	1	0
LCL_MUX_SEL_IN_1			LCL_MUX_SEL_IN_0				
R/W-0h			R/W-0h				

Table 9-70. CLB_LCL_MUX_SEL_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MISC_INPUT_SEL_3	R/W	0h	Setting this bit to 1 will select MISC_INPUT instead of Local Mux for Input 3 Reset type: SYSRSn
30	MISC_INPUT_SEL_2	R/W	0h	Setting this bit to 1 will select MISC_INPUT instead of Local Mux for Input 2 Reset type: SYSRSn
29	MISC_INPUT_SEL_1	R/W	0h	Setting this bit to 1 will select MISC_INPUT instead of Local Mux for Input 1 Reset type: SYSRSn
28	MISC_INPUT_SEL_0	R/W	0h	Setting this bit to 1 will select MISC_INPUT instead of Local Mux for Input 0 Reset type: SYSRSn
27-20	RESERVED	R-0	0h	Reserved
19-15	LCL_MUX_SEL_IN_3	R/W	0h	5 bit MUX Select for Local MUX control for Input 3 See Local Signals and Mux Selection Table Reset type: SYSRSn
14-10	LCL_MUX_SEL_IN_2	R/W	0h	5 bit MUX Select for Local MUX control for Input 2 See Local Signals and Mux Selection Table Reset type: SYSRSn
9-5	LCL_MUX_SEL_IN_1	R/W	0h	5 bit MUX Select for Local MUX control for Input 1 See Local Signals and Mux Selection Table Reset type: SYSRSn
4-0	LCL_MUX_SEL_IN_0	R/W	0h	5 bit MUX Select for Local MUX control for Input 0 See Local Signals and Mux Selection Table Reset type: SYSRSn

9.10.3.7 CLB_LCL_MUX_SEL_2 Register (Offset = Ch) [Reset = 0h]

CLB_LCL_MUX_SEL_2 is shown in Figure 9-68 and described in Table 9-71.

Return to the Summary Table.

Input Mux selection for local mux

Figure 9-68. CLB_LCL_MUX_SEL_2 Register

31	30	29	28	27	26	25	24
MISC_INPUT_SEL_7	MISC_INPUT_SEL_6	MISC_INPUT_SEL_5	MISC_INPUT_SEL_4	RESERVED			
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0-0h			
23	22	21	20	19	18	17	16
RESERVED				LCL_MUX_SEL_IN_7			
R-0-0h				R/W-0h			
15	14	13	12	11	10	9	8
LCL_MUX_SEL_IN_7	LCL_MUX_SEL_IN_6					LCL_MUX_SEL_IN_5	
R/W-0h	R/W-0h					R/W-0h	
7	6	5	4	3	2	1	0
LCL_MUX_SEL_IN_5			LCL_MUX_SEL_IN_4				
R/W-0h			R/W-0h				

Table 9-71. CLB_LCL_MUX_SEL_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MISC_INPUT_SEL_7	R/W	0h	Setting this bit to 1 will select MISC_INPUT instead of Local Mux for Input 7 Reset type: SYSRSn
30	MISC_INPUT_SEL_6	R/W	0h	Setting this bit to 1 will select MISC_INPUT instead of Local Mux for Input 6 Reset type: SYSRSn
29	MISC_INPUT_SEL_5	R/W	0h	Setting this bit to 1 will select MISC_INPUT instead of Local Mux for Input 5 Reset type: SYSRSn
28	MISC_INPUT_SEL_4	R/W	0h	Setting this bit to 1 will select MISC_INPUT instead of Local Mux for Input 4 Reset type: SYSRSn
27-20	RESERVED	R-0	0h	Reserved
19-15	LCL_MUX_SEL_IN_7	R/W	0h	5 bit MUX Select for Local MUX control for Input 7 See Local Signals and Mux Selection Table Reset type: SYSRSn
14-10	LCL_MUX_SEL_IN_6	R/W	0h	5 bit MUX Select for Local MUX control for Input 6 See Local Signals and Mux Selection Table Reset type: SYSRSn
9-5	LCL_MUX_SEL_IN_5	R/W	0h	5 bit MUX Select for Local MUX control for Input 5 See Local Signals and Mux Selection Table Reset type: SYSRSn
4-0	LCL_MUX_SEL_IN_4	R/W	0h	5 bit MUX Select for Local MUX control for Input 4 See Local Signals and Mux Selection Table Reset type: SYSRSn

9.10.3.8 CLB_BUF_PTR Register (Offset = Eh) [Reset = 0h]

CLB_BUF_PTR is shown in Figure 9-69 and described in Table 9-72.

Return to the Summary Table.

PUSH and PULL pointers

Figure 9-69. CLB_BUF_PTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								PUSH								RESERVED								PULL							
R-0-0h								R/W-0h								R-0-0h								R/W-0h							

Table 9-72. CLB_BUF_PTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R-0	0h	Reserved
23-16	PUSH	R/W	0h	8 bit pointer which indicates the number of data values which have been pulled from the buffer by the High Level Controller. This counter will wrap around after 0xff. The Least significant 2 bits are used as the actual pointer for the operation. Reset type: SYSRSn
15-8	RESERVED	R-0	0h	Reserved
7-0	PULL	R/W	0h	8 bit pointer which indicates the number of data values that have been written by the High Level controller into the buffer. The Least significant 2 bits are used as the actual pointer for the operation. Reset type: SYSRSn

9.10.3.9 CLB_GP_REG Register (Offset = 10h) [Reset = 0h]

CLB_GP_REG is shown in Figure 9-70 and described in Table 9-73.

Return to the Summary Table.

General purpose register for CELL inputs

Figure 9-70. CLB_GP_REG Register

31	30	29	28	27	26	25	24
SW_RLS_CTRL_7	SW_RLS_CTRL_6	SW_RLS_CTRL_5	SW_RLS_CTRL_4	SW_RLS_CTRL_3	SW_RLS_CTRL_2	SW_RLS_CTRL_1	SW_RLS_CTRL_0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
SW_GATING_CTRL_7	SW_GATING_CTRL_6	SW_GATING_CTRL_5	SW_GATING_CTRL_4	SW_GATING_CTRL_3	SW_GATING_CTRL_2	SW_GATING_CTRL_1	SW_GATING_CTRL_0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
REG							
R/W-0h							

Table 9-73. CLB_GP_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SW_RLS_CTRL_7	R/W	0h	Software release control for output 7 of the asynchronous output conditioning block Reset type: SYSRSn
30	SW_RLS_CTRL_6	R/W	0h	Software release control for output 6 of the asynchronous output conditioning block Reset type: SYSRSn
29	SW_RLS_CTRL_5	R/W	0h	Software release control for output 5 of the asynchronous output conditioning block Reset type: SYSRSn
28	SW_RLS_CTRL_4	R/W	0h	Software release control for output 4 of the asynchronous output conditioning block Reset type: SYSRSn
27	SW_RLS_CTRL_3	R/W	0h	Software release control for output 3 of the asynchronous output conditioning block Reset type: SYSRSn
26	SW_RLS_CTRL_2	R/W	0h	Software release control for output 2 of the asynchronous output conditioning block Reset type: SYSRSn
25	SW_RLS_CTRL_1	R/W	0h	Software release control for output 1 of the asynchronous output conditioning block Reset type: SYSRSn
24	SW_RLS_CTRL_0	R/W	0h	Software release control for output 0 of the asynchronous output conditioning block Reset type: SYSRSn
23	SW_GATING_CTRL_7	R/W	0h	Software gating control for output 7 of the asynchronous output conditioning block Reset type: SYSRSn
22	SW_GATING_CTRL_6	R/W	0h	Software gating control for output 6 of the asynchronous output conditioning block Reset type: SYSRSn

Table 9-73. CLB_GP_REG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	SW_GATING_CTRL_5	R/W	0h	Software gating control for output 5 of the asynchronous output conditioning block Reset type: SYSRSn
20	SW_GATING_CTRL_4	R/W	0h	Software gating control for output 4 of the asynchronous output conditioning block Reset type: SYSRSn
19	SW_GATING_CTRL_3	R/W	0h	Software gating control for output 3 of the asynchronous output conditioning block Reset type: SYSRSn
18	SW_GATING_CTRL_2	R/W	0h	Software gating control for output 2 of the asynchronous output conditioning block Reset type: SYSRSn
17	SW_GATING_CTRL_1	R/W	0h	Software gating control for output 1 of the asynchronous output conditioning block Reset type: SYSRSn
16	SW_GATING_CTRL_0	R/W	0h	Software gating control for output 0 of the asynchronous output conditioning block Reset type: SYSRSn
15-8	RESERVED	R-0	0h	Reserved
7-0	REG	R/W	0h	8 bits which are directly connected to the 8 inputs of the CELL if that corresponding bit is selected in the CLB_IN_MUX_SEL_0 register Reset type: SYSRSn

9.10.3.10 CLB_OUT_EN Register (Offset = 12h) [Reset = 0h]

CLB_OUT_EN is shown in Figure 9-71 and described in Table 9-74.

Return to the Summary Table.

CELL output enable register

Figure 9-71. CLB_OUT_EN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUT0																															
R/W-0h																															

Table 9-74. CLB_OUT_EN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	OUT0	R/W	0h	32 bits which are directly driven out as OUTPUT_EN signals. Enabling bit x (x = 0:31) will override the corresponding peripheral signal muxed on the CLB OUTx. Reset type: SYSRSn

9.10.3.11 CLB_GLBL_MUX_SEL_1 Register (Offset = 14h) [Reset = 0h]

CLB_GLBL_MUX_SEL_1 is shown in Figure 9-72 and described in Table 9-75.

Return to the Summary Table.

Global Mux select for CELL inputs

Figure 9-72. CLB_GLBL_MUX_SEL_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				GLBL_MUX_SEL_IN_3							GLBL_MUX_SEL_IN_2				
R-0-0h				R/W-0h							R/W-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GLBL_MUX_SEL_IN_2				GLBL_MUX_SEL_IN_1							GLBL_MUX_SEL_IN_0				
R/W-0h				R/W-0h							R/W-0h				

Table 9-75. CLB_GLBL_MUX_SEL_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R-0	0h	Reserved
27-21	GLBL_MUX_SEL_IN_3	R/W	0h	7 bit MUX Select for Global MUX control for Input 3 See Global Signals and Mux Selection Table Reset type: SYSRSn
20-14	GLBL_MUX_SEL_IN_2	R/W	0h	7 bit MUX Select for Global MUX control for Input 2 See Global Signals and Mux Selection Table Reset type: SYSRSn
13-7	GLBL_MUX_SEL_IN_1	R/W	0h	7 bit MUX Select for Global MUX control for Input 1 See Global Signals and Mux Selection Table Reset type: SYSRSn
6-0	GLBL_MUX_SEL_IN_0	R/W	0h	7 bit MUX Select for Global MUX control for Input 0 See Global Signals and Mux Selection Table Reset type: SYSRSn

9.10.3.12 CLB_GLBL_MUX_SEL_2 Register (Offset = 16h) [Reset = 0h]

CLB_GLBL_MUX_SEL_2 is shown in Figure 9-73 and described in Table 9-76.

Return to the Summary Table.

Global Mux select for CELL inputs

Figure 9-73. CLB_GLBL_MUX_SEL_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				GLBL_MUX_SEL_IN_7							GLBL_MUX_SEL_IN_6				
R-0-0h				R/W-0h							R/W-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GLBL_MUX_SEL_IN_6			GLBL_MUX_SEL_IN_5						GLBL_MUX_SEL_IN_4						
R/W-0h			R/W-0h						R/W-0h						

Table 9-76. CLB_GLBL_MUX_SEL_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R-0	0h	Reserved
27-21	GLBL_MUX_SEL_IN_7	R/W	0h	7 bit MUX Select for Global MUX control for Input 7 See Global Signals and Mux Selection Table Reset type: SYSRSn
20-14	GLBL_MUX_SEL_IN_6	R/W	0h	7 bit MUX Select for Global MUX control for Input 6 See Global Signals and Mux Selection Table Reset type: SYSRSn
13-7	GLBL_MUX_SEL_IN_5	R/W	0h	7 bit MUX Select for Global MUX control for Input 5 See Global Signals and Mux Selection Table Reset type: SYSRSn
6-0	GLBL_MUX_SEL_IN_4	R/W	0h	7 bit MUX Select for Global MUX control for Input 4 See Global Signals and Mux Selection Table Reset type: SYSRSn

9.10.3.13 CLB_PRESCALE_CTRL Register (Offset = 18h) [Reset = 0h]

CLB_PRESCALE_CTRL is shown in Figure 9-74 and described in Table 9-77.

Return to the Summary Table.

Prescaler register control

Figure 9-74. CLB_PRESCALE_CTRL Register

31	30	29	28	27	26	25	24
PRESCALE							
R/W-0h							
23	22	21	20	19	18	17	16
PRESCALE							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		TAP				STRB	CLKEN
R-0-0h		R/W-0h				R/W-0h	R/W-0h

Table 9-77. CLB_PRESCALE_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	PRESCALE	R/W	0h	16-bit Value of prescaler to be used for the counter as reference to reset when reaching this value. The counter is a simple incrementing counter which will count up to the reference value and reset to 0 and this cycle will continue as long as the counter is enabled. This 16-bit register value is used as a reference for the 16-bit counter to reset to zero whenever count reaches this value. Reset type: SYSRSn
15-6	RESERVED	R-0	0h	Reserved
5-2	TAP	R/W	0h	TAP Select value. These 4 bits will be used as a select to tap one of the 16 register bit position of the counter as the output. 0000 selects Counter Bit position 0 0001 selects Counter Bit position 1 .... 1111 selects Counter Bit position 15 Reset type: SYSRSn
1	STRB	R/W	0h	When set to 0, a strobe output will be sent out whenever the counter value matches the PRESCALE_VALUE. When set to 1, the output of the counter register bit position as selected by TAP_SELECT_VALUE will be sent out. Reset type: SYSRSn
0	CLKEN	R/W	0h	Enable the prescale clock/strobe generator. A 16-bit counter is used to either generate a strobe or send out a selected counter bit position to the CLB CELL. This is meant to be a general purpose strobe/prescaled clock which can be used by the CELL logic if needed. This will be sent to the CELL through one of the LCL_IN MUX ports. Reset type: SYSRSn

9.10.3.14 CLB_INTR_TAG_REG Register (Offset = 20h) [Reset = 0h]

CLB_INTR_TAG_REG is shown in Figure 9-75 and described in Table 9-78.

Return to the Summary Table.

Interrupt Tag register

Figure 9-75. CLB_INTR_TAG_REG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		TAG					
R-0-0h		R/W-0h					

Table 9-78. CLB_INTR_TAG_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-6	RESERVED	R-0	0h	Reserved
5-0	TAG	R/W	0h	6 bits which are used by the High Level Controller to set a tag value on flagging interrupts. . This can be cleared through the VBUS interface since it is writeable through the VBUS. Reset type: SYSRSn

9.10.3.15 CLB_LOCK Register (Offset = 22h) [Reset = 0h]

CLB_LOCK is shown in Figure 9-76 and described in Table 9-79.

Return to the Summary Table.

Lock control register

Figure 9-76. CLB_LOCK Register

31	30	29	28	27	26	25	24
KEY							
WSonce-0h							
23	22	21	20	19	18	17	16
KEY							
WSonce-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK
R-0-0h							R/W-0h

Table 9-79. CLB_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	WSonce	0h	These 16 bits act as a key to enable writes to Bit 0 of this register. The only time a '1' can be written to Bit 0 is by a single 32-bit write where bits 31:16 equal 0x5a5a and bit 0 is '1'. All other writes are ignored including separate 16-bit writes. This is EALLOW protected. Reset type: SYSRSn
15-1	RESERVED	R-0	0h	Reserved
0	LOCK	R/W	0h	This bit is used as a one-time write bit (Set Once). Once it is set to '1', only a reset (SYSRSN 0) will clear this bit back to 0. Reset type: SYSRSn

9.10.3.16 CLB_HLC_INSTR_READ_PTR Register (Offset = 24h) [Reset = 0h]

CLB_HLC_INSTR_READ_PTR is shown in Figure 9-77 and described in Table 9-80.

Return to the Summary Table.

HLC instruction read pointer

Figure 9-77. CLB_HLC_INSTR_READ_PTR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED			READ_PTR				
R-0-0h			R/W-0h				

Table 9-80. CLB_HLC_INSTR_READ_PTR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R-0	0h	Reserved
4-0	READ_PTR	R/W	0h	This is a 5 bit value which will be used as an address pointer to read out HLC instruction memory. Reset type: SYSRSn

9.10.3.17 CLB_HLC_INSTR_VALUE Register (Offset = 26h) [Reset = 0h]

CLB_HLC_INSTR_VALUE is shown in Figure 9-78 and described in Table 9-81.

Return to the Summary Table.

HLC instruction read value

Figure 9-78. CLB_HLC_INSTR_VALUE Register

15	14	13	12	11	10	9	8
RESERVED				INSTR			
R-0-0h				R-0h			
7	6	5	4	3	2	1	0
INSTR							
R-0h							

Table 9-81. CLB_HLC_INSTR_VALUE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R-0	0h	Reserved
11-0	INSTR	R	0h	This is a 12 bit value which will read the content of the HLC instruction memory address pointed by CLB_HLC_INSTR_READ_PTR register. Reset type: SYSRSn

9.10.3.18 CLB_DBG_OUT_2 Register (Offset = 2Eh) [Reset = 0h]

CLB_DBG_OUT_2 is shown in Figure 9-79 and described in Table 9-82.

Return to the Summary Table.

Visibility for CLB inputs and final asynchronous outputs

Figure 9-79. CLB_DBG_OUT_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																IN								OUT							
R/W1C-0h																R/W-0h								R/W-0h							

Table 9-82. CLB_DBG_OUT_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W1C	0h	Reserved
15-8	IN	R/W	0h	These bits reflect the state of the 8 inputs finally going to the CELL after selection and input conditioning. Reset type: SYSRSn
7-0	OUT	R/W	0h	These bits reflect the state of the 8 outputs of the Output Conditioning Block. Reset type: SYSRSn

9.10.3.19 CLB_DBG_R0 Register (Offset = 30h) [Reset = 0h]

CLB_DBG_R0 is shown in Figure 9-80 and described in Table 9-83.

Return to the Summary Table.

R0 of High level Controller

Figure 9-80. CLB_DBG_R0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG																															
R-0h																															

Table 9-83. CLB_DBG_R0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBG	R	0h	CLB_DBG_R0 Reset type: SYSRSn

9.10.3.20 CLB_DBG_R1 Register (Offset = 32h) [Reset = 0h]

CLB_DBG_R1 is shown in Figure 9-81 and described in Table 9-84.

Return to the Summary Table.

R1 of High level Controller

Figure 9-81. CLB_DBG_R1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG																															
R-0h																															

Table 9-84. CLB_DBG_R1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBG	R	0h	CLB_DBG_R1 Reset type: SYSRSn

9.10.3.21 CLB_DBG_R2 Register (Offset = 34h) [Reset = 0h]

CLB_DBG_R2 is shown in Figure 9-82 and described in Table 9-85.

Return to the Summary Table.

R2 of High level Controller

Figure 9-82. CLB_DBG_R2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG																															
R-0h																															

Table 9-85. CLB_DBG_R2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBG	R	0h	CLB_DBG_R2 Reset type: SYSRSn

9.10.3.22 CLB_DBG_R3 Register (Offset = 36h) [Reset = 0h]

CLB_DBG_R3 is shown in Figure 9-83 and described in Table 9-86.

Return to the Summary Table.

R3 of High level Controller

Figure 9-83. CLB_DBG_R3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG																															
R-0h																															

Table 9-86. CLB_DBG_R3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBG	R	0h	CLB_DBG_R3 Reset type: SYSRSn

9.10.3.23 CLB_DBG_C0 Register (Offset = 38h) [Reset = 0h]

CLB_DBG_C0 is shown in Figure 9-84 and described in Table 9-87.

Return to the Summary Table.

Count of Unit 0

Figure 9-84. CLB_DBG_C0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG																															
R-0h																															

Table 9-87. CLB_DBG_C0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBG	R	0h	CLB_DBG_C0 Reset type: SYSRSn

9.10.3.24 CLB_DBG_C1 Register (Offset = 3Ah) [Reset = 0h]

CLB_DBG_C1 is shown in Figure 9-85 and described in Table 9-88.

Return to the Summary Table.

Count of Unit 1

Figure 9-85. CLB_DBG_C1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG																															
R-0h																															

Table 9-88. CLB_DBG_C1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBG	R	0h	CLB_DBG_C1 Reset type: SYSRSn

9.10.3.25 CLB_DBG_C2 Register (Offset = 3Ch) [Reset = 0h]

CLB_DBG_C2 is shown in Figure 9-86 and described in Table 9-89.

Return to the Summary Table.

Count of Unit 2

Figure 9-86. CLB_DBG_C2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG																															
R-0h																															

Table 9-89. CLB_DBG_C2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBG	R	0h	CLB_DBG_C2 Reset type: SYSRSn

9.10.3.26 CLB_DBG_OUT Register (Offset = 3Eh) [Reset = 00010100h]

CLB_DBG_OUT is shown in Figure 9-87 and described in Table 9-90.

Return to the Summary Table.

Outputs of various units in the Cell

Figure 9-87. CLB_DBG_OUT Register

31	30	29	28	27	26	25	24
OUT7	OUT6	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
LUT42_OUT	FSM2_LUTOUT	FSM2_S1	FSM2_S0	COUNT2_MATCH1	COUNT2_ZERO	COUNT2_MATCH2	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-1-1h
15	14	13	12	11	10	9	8
LUT41_OUT	FSM1_LUTOUT	FSM1_S1	FSM1_S0	COUNT1_MATCH1	COUNT1_ZERO	COUNT1_MATCH2	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-1-1h
7	6	5	4	3	2	1	0
LUT40_OUT	FSM0_LUTOUT	FSM0_S1	FSM0_S0	COUNT0_MATCH1	COUNT0_ZERO	COUNT0_MATCH2	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0-0h

Table 9-90. CLB_DBG_OUT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	OUT7	R	0h	CELL Output 7 Reset type: SYSRSn
30	OUT6	R	0h	CELL Output 6 Reset type: SYSRSn
29	OUT5	R	0h	CELL Output 5 Reset type: SYSRSn
28	OUT4	R	0h	CELL Output 4 Reset type: SYSRSn
27	OUT3	R	0h	CELL Output 3 Reset type: SYSRSn
26	OUT2	R	0h	CELL Output 2 Reset type: SYSRSn
25	OUT1	R	0h	CELL Output 1 Reset type: SYSRSn
24	OUT0	R	0h	CELL Output 0 Reset type: SYSRSn
23	LUT42_OUT	R	0h	LUT4_OUT UNIT 2 Reset type: SYSRSn
22	FSM2_LUTOUT	R	0h	FSM_LUT_OUT UNIT 2 Reset type: SYSRSn
21	FSM2_S1	R	0h	FSM_S1 UNIT 2 Reset type: SYSRSn
20	FSM2_S0	R	0h	FSM_S0 UNIT 2 Reset type: SYSRSn
19	COUNT2_MATCH1	R	0h	COUNT_MATCH1 UNIT 2 Reset type: SYSRSn
18	COUNT2_ZERO	R	0h	COUNT_ZERO UNIT 2 Reset type: SYSRSn

Table 9-90. CLB_DBG_OUT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	COUNT2_MATCH2	R	0h	COUNT_MATCH2 UNIT 2 Reset type: SYSRSn
16	RESERVED	R-1	1h	Reserved
15	LUT41_OUT	R	0h	LUT4_OUT UNIT 1 Reset type: SYSRSn
14	FSM1_LUTOUT	R	0h	FSM_LUT_OUT UNIT 1 Reset type: SYSRSn
13	FSM1_S1	R	0h	FSM_S1 UNIT 1 Reset type: SYSRSn
12	FSM1_S0	R	0h	FSM_S0 UNIT 1 Reset type: SYSRSn
11	COUNT1_MATCH1	R	0h	COUNT_MATCH1 UNIT 1 Reset type: SYSRSn
10	COUNT1_ZERO	R	0h	COUNT_ZERO UNIT 1 Reset type: SYSRSn
9	COUNT1_MATCH2	R	0h	COUNT_MATCH2 UNIT 1 Reset type: SYSRSn
8	RESERVED	R-1	1h	Reserved
7	LUT40_OUT	R	0h	LUT4_OUT UNIT 0 Reset type: SYSRSn
6	FSM0_LUTOUT	R	0h	FSM_LUT_OUT UNIT 0 Reset type: SYSRSn
5	FSM0_S1	R	0h	FSM_S1 UNIT 0 Reset type: SYSRSn
4	FSM0_S0	R	0h	FSM_S0 UNIT 0 Reset type: SYSRSn
3	COUNT0_MATCH1	R	0h	COUNT_MATCH1 UNIT 0 Reset type: SYSRSn
2	COUNT0_ZERO	R	0h	COUNT_ZERO UNIT 0 Reset type: SYSRSn
1	COUNT0_MATCH2	R	0h	COUNT_MATCH2 UNIT 0 Reset type: SYSRSn
0	RESERVED	R-0	0h	Reserved

9.10.4 CLB_DATA_EXCHANGE_REGS Registers

Table 9-91 lists the memory-mapped registers for the CLB_DATA_EXCHANGE_REGS registers. All register offset addresses not listed in Table 9-91 should be considered as reserved locations and the register contents should not be modified.

Table 9-91. CLB_DATA_EXCHANGE_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CLB_PUSH	CLB_PUSH FIFO Registers (from HLC)		Go
40h	CLB_PULL	CLB_PULL FIFO Registers (TO HLC)		Go

Complex bit access types are encoded to fit into small table cells. Table 9-92 shows the codes that are used for access types in this section.

Table 9-92. CLB_DATA_EXCHANGE_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

9.10.4.1 CLB_PUSH Register (Offset = 0h) [Reset = 0h]

CLB_PUSH is shown in Figure 9-88 and described in Table 9-93.

Return to the Summary Table.

CLB_PUSH FIFO Registers (from HLC)

Figure 9-88. CLB_PUSH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PUSH																															
R-0h																															

Table 9-93. CLB_PUSH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PUSH	R	0h	FIFO TO System From CLB Reset type: SYSRSn

9.10.4.2 CLB_PULL Register (Offset = 40h) [Reset = 0h]

CLB_PULL is shown in Figure 9-89 and described in Table 9-94.

Return to the Summary Table.

CLB_PULL FIFO Registers (TO HLC)

Figure 9-89. CLB_PULL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PULL																															
R/W-0h																															

Table 9-94. CLB_PULL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PULL	R/W	0h	FIFO From system TO CLB Reset type: SYSRSn

9.10.5 CLB Registers to Driverlib Functions

Table 9-95. CLB Registers to Driverlib Functions

File	Driverlib Function
COUNT_RESET	
clb.h	CLB_selectCounterInputs
COUNT_MODE_1	
clb.h	CLB_selectCounterInputs
COUNT_MODE_0	
clb.h	CLB_selectCounterInputs
COUNT_EVENT	
clb.h	CLB_selectCounterInputs
FSM_EXTRA_IN0	
clb.h	CLB_selectFSMInputs
FSM_EXTERNAL_IN0	
clb.h	CLB_selectFSMInputs
FSM_EXTERNAL_IN1	
clb.h	CLB_selectFSMInputs
FSM_EXTRA_IN1	
clb.h	CLB_selectFSMInputs
LUT4_IN0	
clb.h	CLB_selectLUT4Inputs
LUT4_IN1	
clb.h	CLB_selectLUT4Inputs
LUT4_IN2	
clb.h	CLB_selectLUT4Inputs
LUT4_IN3	
clb.h	CLB_selectLUT4Inputs
FSM_LUT_FN1_0	
clb.h	CLB_configFSMLUTFunction
FSM_LUT_FN2	
clb.h	CLB_configFSMLUTFunction
LUT4_FN1_0	

Table 9-95. CLB Registers to Driverlib Functions (continued)

File	Driverlib Function
clb.h	CLB_configLUT4Function
LUT4_FN2	
clb.h	CLB_configLUT4Function
FSM_NEXT_STATE_0	
clb.h	CLB_configFSMNextState
FSM_NEXT_STATE_1	
clb.h	CLB_configFSMNextState
FSM_NEXT_STATE_2	
clb.h	CLB_configFSMNextState
MISC_CONTROL	
clb.h	CLB_configMiscCtrlModes
OUTPUT_LUT_0	
clb.h	CLB_configOutputLUT
OUTPUT_LUT_1	
-	See OUTPUT_LUT_0
OUTPUT_LUT_2	
-	See OUTPUT_LUT_0
OUTPUT_LUT_3	
-	See OUTPUT_LUT_0
OUTPUT_LUT_4	
-	See OUTPUT_LUT_0
OUTPUT_LUT_5	
-	See OUTPUT_LUT_0
OUTPUT_LUT_6	
-	See OUTPUT_LUT_0
OUTPUT_LUT_7	
-	See OUTPUT_LUT_0
HLC_EVENT_SEL	
clb.h	CLB_configHLCEventSelect
COUNT_MATCH_TAP_SEL	
clb.h	CLB_configCounterTapSelects
OUTPUT_COND_CTRL_0	
clb.h	CLB_configAOC
OUTPUT_COND_CTRL_1	
-	
OUTPUT_COND_CTRL_2	
-	
OUTPUT_COND_CTRL_3	
-	
OUTPUT_COND_CTRL_4	
-	
OUTPUT_COND_CTRL_5	
-	
OUTPUT_COND_CTRL_6	
-	

Table 9-95. CLB Registers to Driverlib Functions (continued)

File	Driverlib Function
OUTPUT_COND_CTRL_7	
-	
MISC_ACCESS_CTRL	
clb.h	CLB_disableOutputMaskUpdates
clb.h	CLB_enableOutputMaskUpdates
clb.h	CLB_disableSPIBufferAccess
clb.h	CLB_enableSPIBufferAccess
SPI_DATA_CTRL_HI	
clb.h	CLB_configSPIBufferLoadSignal
clb.h	CLB_configSPIBufferShift
LOAD_EN	
clb.h	CLB_enableCLB
clb.h	CLB_disableCLB
clb.h	CLB_enableNMI
clb.h	CLB_disableNMI
clb.h	CLB_writeInterface
clb.h	CLB_enablePipelineMode
clb.h	CLB_disablePipelineMode
LOAD_ADDR	
clb.h	CLB_writeInterface
LOAD_DATA	
clb.h	CLB_writeInterface
INPUT_FILTER	
clb.h	CLB_selectInputFilter
clb.h	CLB_enableSynchronization
clb.h	CLB_disableSynchronization
clb.h	CLB_enableInputPipelineMode
clb.h	CLB_disableInputPipelineMode
IN_MUX_SEL_0	
clb.h	CLB_configGPInputMux
LCL_MUX_SEL_1	
clb.h	CLB_configLocalInputMux
LCL_MUX_SEL_2	
clb.h	CLB_configLocalInputMux
BUF_PTR	
clb.c	CLB_clearFIFOs
GP_REG	
clb.h	CLB_writeSWReleaseControl
clb.h	CLB_writeSWGateControl
clb.h	CLB_setGPREG
clb.h	CLB_getGPREG
OUT_EN	
clb.h	CLB_setOutputMask
GLBL_MUX_SEL_1	
clb.h	CLB_configGlobalInputMux

Table 9-95. CLB Registers to Driverlib Functions (continued)

File	Driverlib Function
GLBL_MUX_SEL_2	
clb.h	CLB_configGlobalInputMux
PRESCALE_CTRL	
clb.h	CLB_configureClockPrescalar
clb.h	CLB_configureStrobeMode
INTR_TAG_REG	
clb.h	CLB_getInterruptTag
clb.h	CLB_clearInterruptTag
LOCK	
clb.h	CLB_enableLock
HLC_INSTR_READ_PTR	
-	
HLC_INSTR_VALUE	
-	
DBG_OUT_2	
-	
DBG_R0	
-	
DBG_R1	
-	
DBG_R2	
-	
DBG_R3	
-	
DBG_C0	
-	
DBG_C1	
-	
DBG_C2	
-	
DBG_OUT	
clb.h	CLB_getOutputStatus
PUSH	
clb.c	CLB_readFIFOs
PULL	
clb.c	CLB_clearFIFOs
clb.c	CLB_writeFIFOs

This page intentionally left blank.

Chapter 10

Dual-Clock Comparator (DCC)

This chapter describes the Dual-Clock Comparator (DCC) module.

10.1 Introduction

The dual-clock comparator module is used for evaluating and monitoring the clock input based on a second clock, which can be a more accurate and reliable version. This instrumentation is used to detect faults in clock source or clock structures, thereby enhancing the system's safety metrics.

10.1.1 Features

The main features of each of the DCC modules are:

- Allows the application to make sure that a fixed ratio is maintained between frequencies of two clock signals.
- Supports the definition of a programmable tolerance window in terms of the number of reference clock cycles.
- Supports continuous monitoring without requiring application intervention.
- Supports a single-sequence mode for spot measurements.
- Allows the selection of a clock source for each of the counters, resulting in several specific use cases.

10.1.2 Block Diagram

Figure 10-1 shows how the DCC connects to the rest of the system. Figure 10-2 shows the main concept of the DCC module.

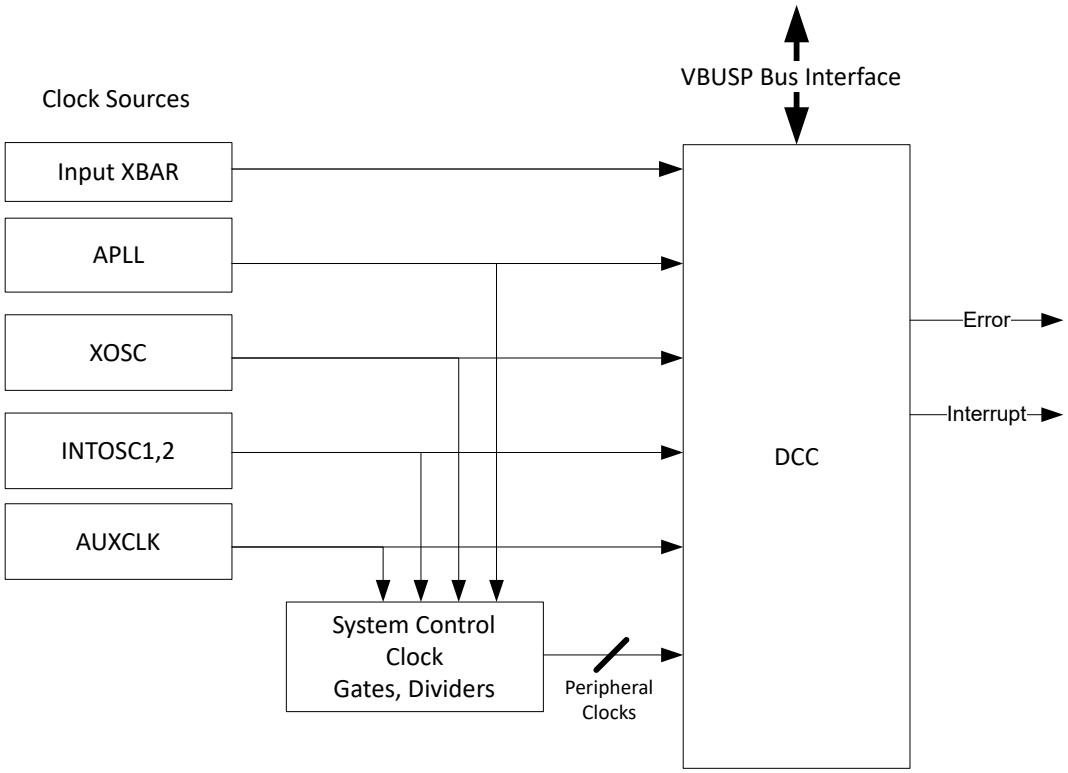


Figure 10-1. DCC Module Overview

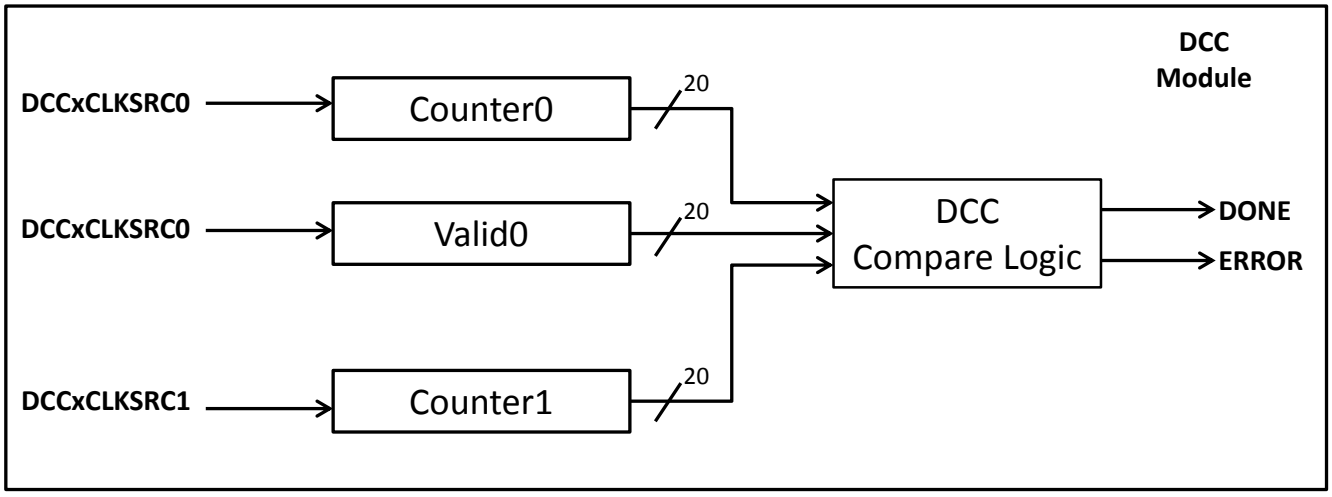


Figure 10-2. DCC Operation

10.2 Module Operation

As shown in Figure 10-2, DCC contains three counters – Counter0, Valid0 and Counter1. Initially, all counters are loaded with their user-defined, pre-load value. Counter0 and Counter1 start decrementing once the DCC is enabled at rates determined by the frequencies of Clock0 and Clock1, respectively. When Counter0 equals 0 (expires), the Valid0 counter decrements at a rate determined by Clock0. If Counter1 decrements to 0 in the valid window, then no error is generated and Clock1 is considered to be good within allowable tolerance as configured by the user.

10.2.1 Configuring DCC Counters

Counter0 and Counter1 are configured based on the ratio between the frequencies of Clock0 and Clock1 (Fclk1×Counter0 = Fclk0×Counter1). The Valid0 counter provides tolerance and is configured based on the error in DCC. Since Clock0 and Clock1 are asynchronous, the start and stop of the counters do not occur synchronously. Hence, while configuring the counters, two different sources of errors must be accounted for:

- DCC Errors due to the asynchronous timing of Clock0 and Clock1: this depends on the frequency of Clock0 and Clock1:
  - If Fclk1 > Fclk0, then Async. Error (in Clock0 cycles) = 2 + 2×(max(Fsysclk/Fclk0))
  - If Fclk1 < Fclk0, then Async. Error (in Clock0 cycles) = 2×(Fclk0/Fclk1) + 2×(max(Fsysclk/Fclk0))
  - If Fclk1 is unknown, then Async. Error (in Clock0 cycles) = 2 + 2×(Fsysclk/Fclk0)
- Digitization Error = 8 Clock0 cycles

DCC Error (in Clock0 Cycles) = Async. Error + Digitization Error

DCC error shows up as a frequency error for clock under measurement. This error is DCC induced and does not represent error in frequency of clock under measurement. The application needs to take this into consideration while configuring the counters, and determine a desirable tolerance for DCC error that defines the window of measurement. To illustrate:

Window (in Clock0 Cycles) = (DCC Error) / (0.01×Tolerance)

For example, if DCC Error is 10 and the tolerance desired is +/-0.1%, then:

Window (in Clock0 Cycles) = 10/(0.01×0.1) = 10000

Based on above formula for Window, if the desired tolerance is low, then the counter values are large and increase the window of measurement. This means that counter values for a tolerance of 0.1% are larger than that of 0.2%. So, based on the application defined tolerance, define the window of measurement in terms of Clock0 cycles.

The clock under measurement can have an allowed frequency error. If this error is expected, then the error can also be accounted while configuring counters. For example, if measuring INTOSC1/2 frequency using an external crystal as a reference clock, the allowable tolerance of INTOSC1/2 (for example, +/-1%) can be accounted for and factored into the counter configuration. The formula is:

Frequency Error Allowed (in Clock0 Cycles) = Window × (Allowable Frequency Tolerance (in %) / 100)

Total Error (in Clock0 Cycles) = DCC Error + Frequency Error Allowed

The following equations are used to configure counter values:

Counter0 (DCCCNTSEED0) = Window - Total Error

Valid0 (DCCVALIDSEED0) = 2 × Total Error

Counter1 (DCCCNTSEED1) = Window × (Fclk1/Fclk0)

Note

Counter1 is a 20-bit counter, so the maximum possible value cannot exceed 1048575. If the value does exceed, then increase the desired Tolerance for DCC error, so that Window of measurement is lowered. The following formula can be used to compute minimum tolerance possible:

Tolerance (%) = (100 × DCC Error × (Fclk1/Fclk0)) / 1048575

10.2.2 Single-Shot Measurement Mode

The DCC module can be programmed to count down one time by enabling the single-shot mode. In this mode, the DCC stops operating when the down counter0 and the valid counter0 reach 0.

At the end of one sequence of counting down in this single-shot mode, the DCC gets disabled automatically, which prevents further counting. This mode is typically used for spot-checking the frequency of a signal.

Example-1: Validating PLLRAWCLK frequency

A practical example of the usage is to validate the PLL output clock frequency using the XTAL as the reference clock. Assume XTAL is 10 MHz, PLL output frequency is 100 MHz, SYSCLK is 100 MHz, allowable Frequency Tolerance is 0.1%, and DCC Tolerance required is 0.1%. The measurement sequence proceeds as follows:

- Set Clock0 source for Counter0 and Valid0 as XTAL, and Clock1 source for Counter1 as PLL output clock.
- Based on the equations defined in Section 10.2.1, calculated seed values for Counters can be Counter0 = 29940; Valid0 = 120; Counter1 = 300000
- Once the DCC is enabled, the counters Counter0 and Counter1 both start counting down from their seed values.
- When Counter0 reaches zero, Counter0 automatically triggers the Valid0 counter.
- When Valid0 reaches zero and Counter1 is not zero, an ERROR status flag is set and a "DCC error" is sent to the PIE. Counter1 is frozen so that the counter stops counting down any further. The application can enable an interrupt to be generated from the PIE whenever this DCC error is indicated. Refer to Table 3-2 to know the channel mapping of DCC Interrupt.
- The application then needs to clear the ERROR status flag and restart the DCC module so that the module is ready for the next spot measurement.

If there is no error generated at the end of the sequence, then the DONE status flag is set and a DONE interrupt is generated. The application must clear the DONE flag before restarting the DCC.

Error Conditions:

An error condition is generated by any one of the following:

1. Counter1 counts down to 0 before Counter0 reaches 0. This means that Clock1 is faster than expected, or Clock0 is slower than expected. This error includes the case when Clock0 is stuck at 1 or 0.
2. Counter1 does not reach 0 even when Counter0 and Valid0 have both reached 0. This means that Clock1 is slower than expected. This error includes the case when Clock1 is stuck at 1 or 0.

Any error freezes the counters from counting. An application can then read out the counter values to help determine what caused the error.

Another example of single-shot mode is to measure the frequency of AUXCLKIN (unknown frequency) using INTOSC1 (10 MHz) as the reference clock and SYSCLK is 10 MHz. The measurement sequence proceeds as follows:

- Set Clock0 source for Counter0 and Valid0 as INTOSC1 (10 MHz), and Clock1 source for Counter1 as AUXCLKIN.
- Now configure counter values using equations in Section 10.2.1. For tolerance = ±0.1%, Total Error = 10 clock0 cycles; Window = 10000 clock0 cycles; Counter0 = 9990; Valid0 = 20. Since Clock1 frequency (Fclk1) is unknown, the Counter1 value can be set to the maximum value, 1048575 (0xFFFFF).
- Once the DCC is enabled, the counters Counter0 and Counter1 both start counting down from their seed values.
- Since Counter1 is set to the maximum value, 1048575, the counter does not expire when Counter0 and Valid0 have expired. This generates an error that is expected and the application ignores this error and uses Counter1 values to compute the frequency of Clock1 (Fclk1).
- Knowing the frequency of Clock0 (INTOSC1), Fclk0 = 10 MHz, and using Equation 1, the frequency of AUXCLKIN, Fclk1, can be measured:

$$Fclk1 = \frac{Fclk0 \times (1048575 - Meas.Counter1)}{(Counter0 + Valid0)} = \frac{10 \times (1048575 - Meas.Counter1)}{(9990 + 20)} \quad (1)$$

10.2.3 Continuous Monitoring Mode

In this mode, the DCC is used by the application to make sure that two clock signals maintain the correct frequency ratio. Suppose the application wants to make sure that the PLL output signal always maintains a fixed frequency relationship with the XTAL:

- In this case, the application can use the XTAL as the Clock0 signal (for Counter0 and Valid0) and the PLL output as the Clock1 (for Counter1).
- The seed values of Counter0, Valid0 and Counter1 are selected based on the equations defined in Section 10.2.1 such that if the actual frequencies of Clock0 and Clock1 are equal to their expected frequencies, then the Counter1 reaches zero during the count down of the Valid0 counter.
- If the Counter1 reaches zero during the count down of the Valid0 counter, then all the counters (Counter0, Valid0, Counter1) are reloaded with their initial seed values.
- This sequence of counting down and checking then continues as long as there is no error, or until the DCC module is disabled.
- The counters must get reloaded if the application resets and restarts the DCC module.

Error Conditions:

An error condition is generated by one of the following:

1. Counter1 counts down to 0 before Counter0 reaches 0. This means that Clock1 is faster than expected or Clock0 is slower than expected. This condition includes the case when Clock0 is stuck at 1 or 0.
2. Counter1 does not reach 0 even when Counter0 and Valid0 have both reached 0. This means that Clock1 is slower than expected. This condition includes the case when Clock1 is stuck at 1 or 0.

Any error freezes the counters from counting. An application can then read out the counter values to help determine what caused the error.

10.2.4 Error Conditions

While operating in continuous mode, the counters get reloaded with the seed values and continue counting down under the following conditions:

- The module is reset or restarted by the application, OR
- Counter0, Valid 0, and Counter1 all reach 0 without any error.

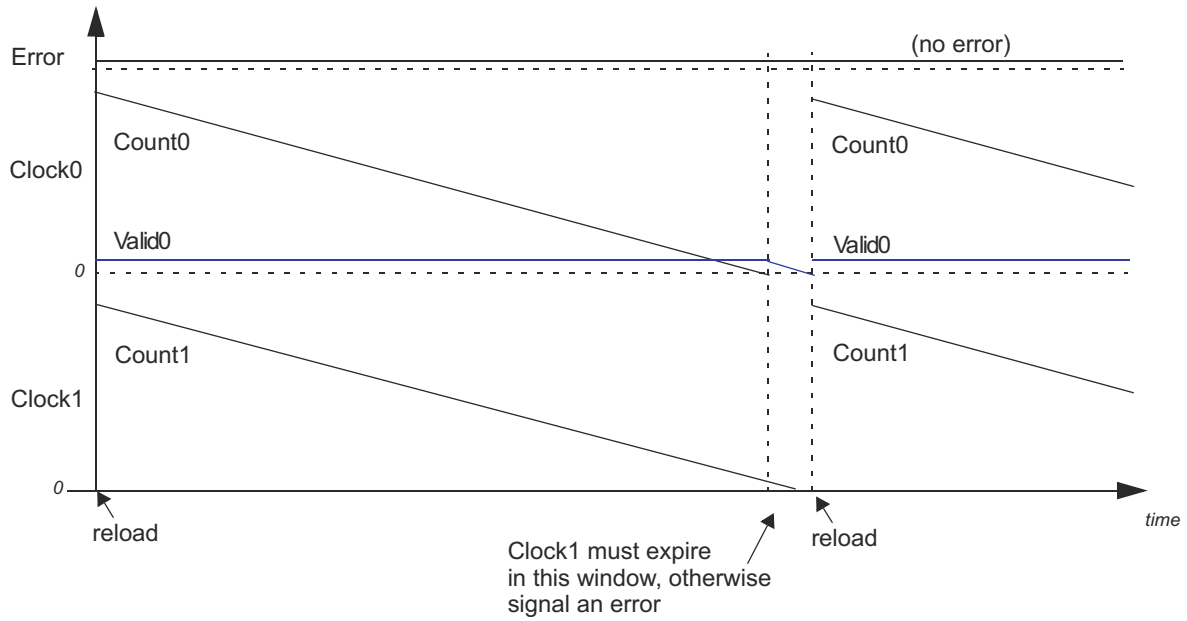


Figure 10-3. Counter Relationship

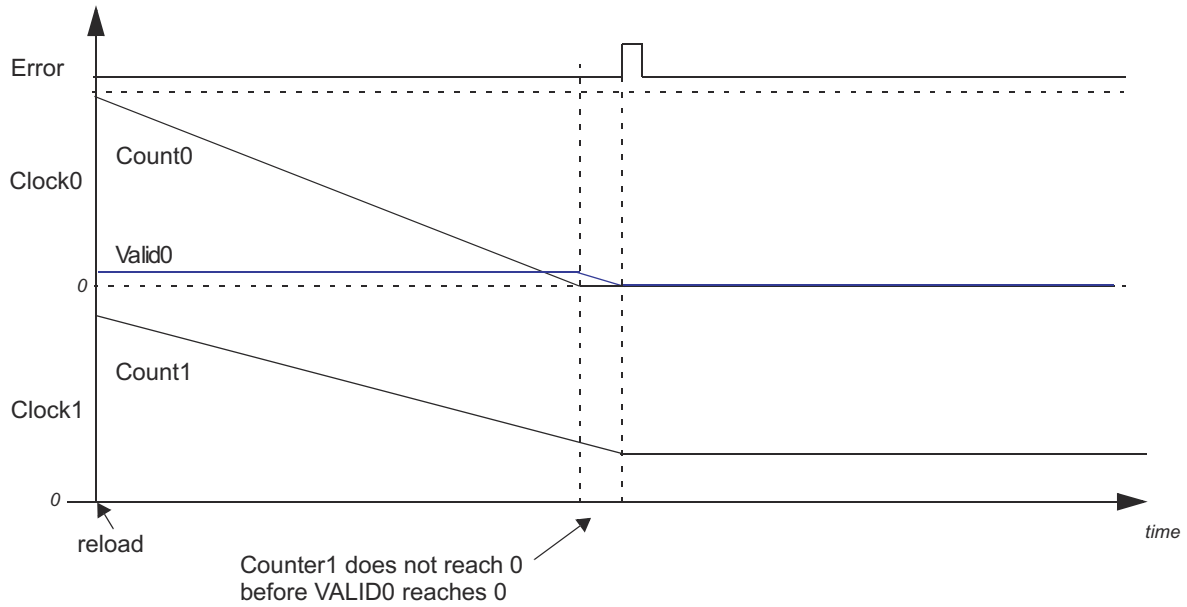


Figure 10-4. Clock1 Slower Than Clock0 - Results in an Error and Stops Counting

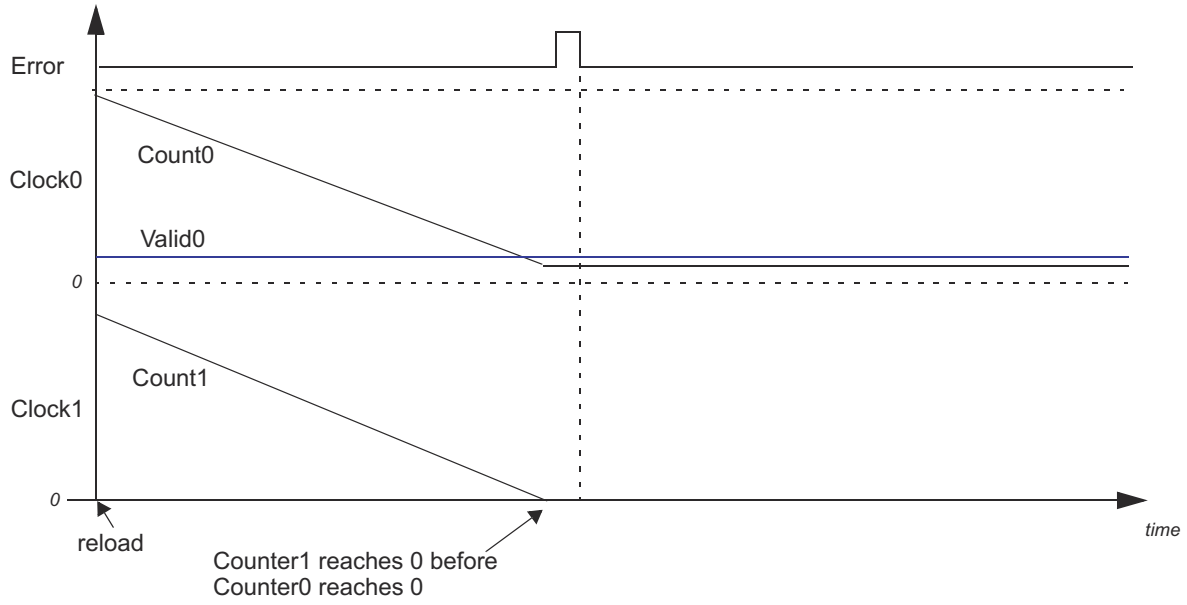


Figure 10-5. Clock1 Faster Than Clock0 - Results in an Error and Stops Counting

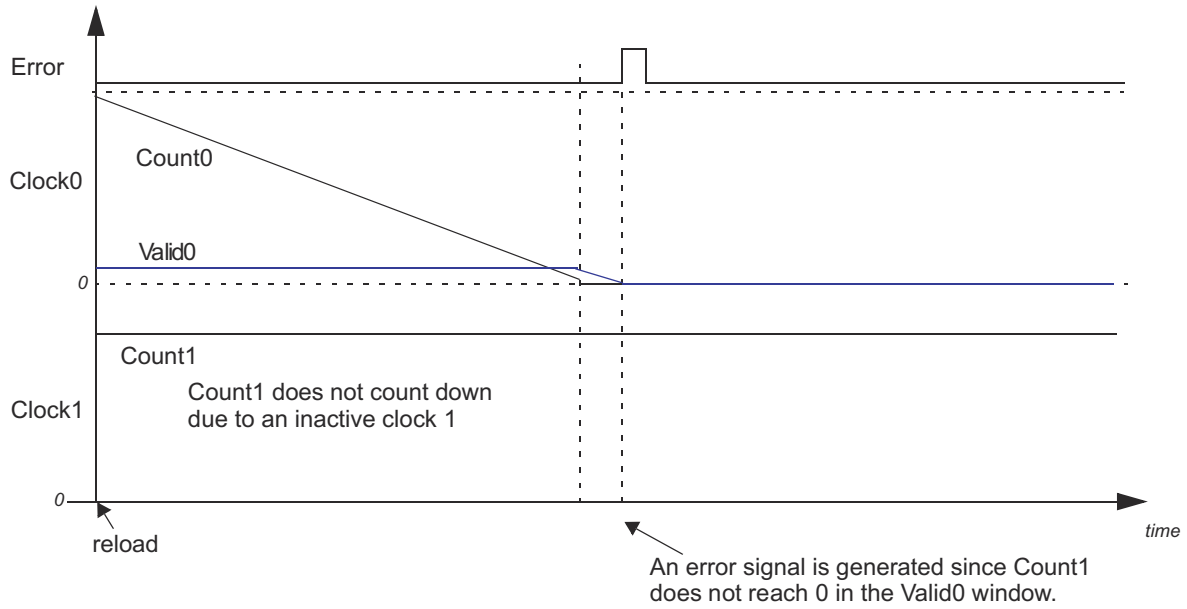


Figure 10-6. Clock1 Not Present - Results in an Error and Stops Counting

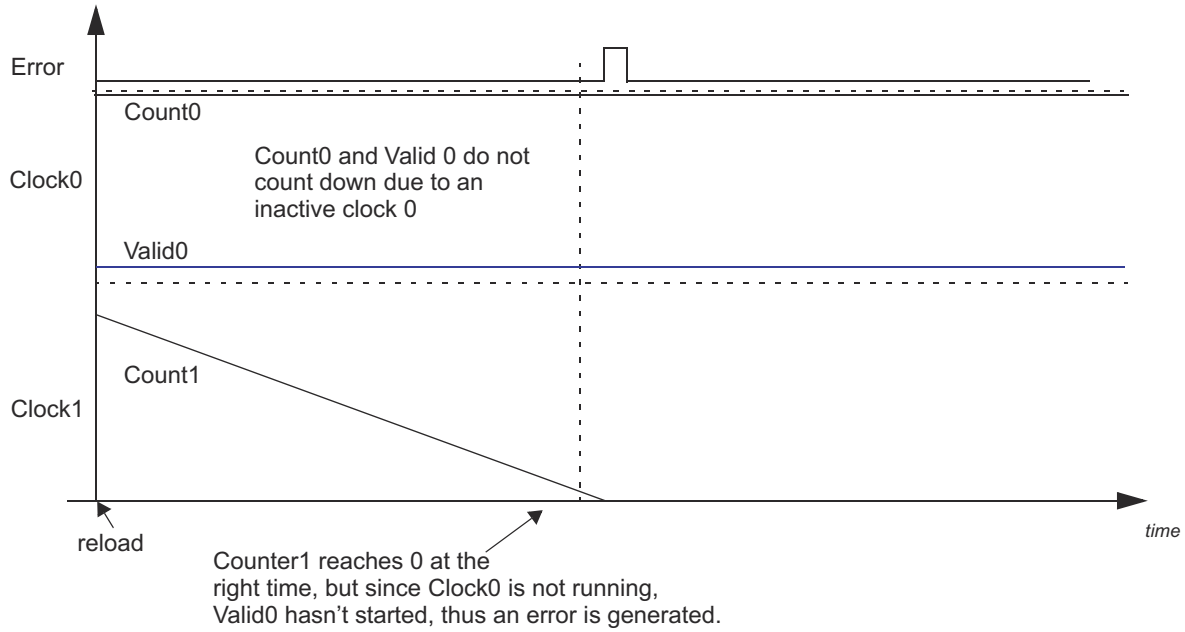


Figure 10-7. Clock0 Not Present - Results in an Error and Stops Counting

10.3 Interrupts

DCC generates an interrupt on either of two events:

- DCC finishes counting and all the counters expire within a defined window indicating DONE operation, provided DCCGCTRL.DONENA=1.
- DCC finishes counting with error where counters do not expire in a defined window. This indicates an ERROR event, and sets an interrupt provided DCCGCTRL.ERRENA=1.

Interrupts generated by DONE or ERROR events are ORed and flagged as SYS_ERR_INT, which goes to INT1.10 in Table 3-2. The application interrupt service routine needs to check the status flag inside the DCCSTATUS register to determine whether the interrupt is due to ERROR or DONE.

10.4 Software

10.4.1 DCC Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/dcc

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

10.4.1.1 DCC Single shot Clock verification

FILE: dcc_ex1_single_shot_verification.c

This program uses the XTAL clock as a reference clock to verify the frequency of the PLLRAW clock.

The Dual-Clock Comparator Module 0 is used for the clock verification. The clocksource0 is the reference clock (Fclk0 = 25Mhz) and the clocksource1 is the clock that needs to be verified (Fclk1 = 200Mhz). Seed is the value that gets loaded into the Counter.

Please refer to the TRM for details on counter seed values to be set.

External Connections

- None

Watch Variables

- *status/result* - Status of the PLLRAW clock verification

10.4.1.2 DCC Single shot Clock measurement

FILE: dcc_ex2_single_shot_measurement.c

This program demonstrates Single Shot measurement of the INTOSC2 clock post trim using XTAL as the reference clock.

The Dual-Clock Comparator Module 0 is used for the clock measurement. The clocksource0 is the reference clock (Fclk0 = 25Mhz) and the clocksource1 is the clock that needs to be measured (Fclk1 = 10Mhz). Since the frequency of the clock1 needs to be measured an initial seed is set to the max value of the counter.

Please refer to the TRM for details on counter seed values to be set.

External Connections

- None

Watch Variables

- *result* - Status if the INTOSC2 clock measurement completed successfully.
- *meas_freq1* - measured clock frequency, in this case for INTOSC2.

10.4.1.3 DCC Continuous clock monitoring

FILE: dcc_ex3_continuous_monitoring_of_clock.c

This program demonstrates continuous monitoring of PLL Clock in the system using INTOSC2 as the reference clock. This would trigger an interrupt on any error, causing the decrement/ reload of counters to stop.

The Dual-Clock Comparator Module 0 is used for the clock monitoring. The clocksource0 is the reference clock (Fclk0 = 10Mhz) and the clocksource1 is the clock that needs to be monitored (Fclk1 = 200Mhz). The clock0 and clock1 seed are set to achieve a window of 500us. Seed is the value that gets loaded into the Counter. For the sake of demo a slight variance is given to clock1 seed value to generate an error on continuous monitoring.

Please refer to the TRM for details on counter seed values to be set. Note : When running in flash configuration it is good to do a reset & restart after loading the example to remove any stale flags/states.

External Connections

- None

Watch Variables

- *status/result* - Status of the PLLRAW clock monitoring
- *cnt0* - Counter0 Value measure when error is generated
- *cnt1* - Counter1 Value measure when error is generated
- *valid* - Valid0 Value measure when error is generated

10.4.1.4 DCC Continuous clock monitoring

FILE: dcc_ex3_continuous_monitoring_of_clock_syscfg.c

This program demonstrates continuous monitoring of PLL Clock in the system using INTOSC2 as the reference clock. This would trigger an interrupt on any error, causing the decrement/ reload of counters to stop. The Dual-Clock Comparator Module 0 is used for the clock monitoring. The clocksource0 is the reference clock (Fclk0 = 10Mhz) and the clocksource1 is the clock that needs to be monitored (Fclk1 = 200Mhz). The clock0 and clock1 seed are set automatically by the error tolerances defined in the sysconfig file included this project. For the sake of demo an un-realistic tolerance is assumed to generate an error on continuous monitoring.

Please refer to the TRM for details on counter seed values to be set. Note : When running in flash configuration it is good to do a reset & restart after loading the example to remove any stale flags/states.

External Connections

- None

Watch Variables

- *status/result* - Status of the PLLRAW clock monitoring
- *cnt0* - Counter0 Value measure when error is generated
- *cnt1* - Counter1 Value measure when error is generated
- *valid* - Valid0 Value measure when error is generated

10.4.1.5 DCC Detection of clock failure

FILE: dcc_ex4_clock_fail_detect.c

This program demonstrates clock failure detection on continuous monitoring of the PLL Clock in the system using XTAL as the osc clock source. Once the oscillator clock fails, it would trigger a DCC error interrupt, causing the decrement/ reload of counters to stop. In this examples, the clock failure is simulated by turning off the XTAL oscillator. Once the ISR is serviced, the osc source is changed to INTOSC1 and the PLL is turned off.

The Dual-Clock Comparator Module 0 is used for the clock monitoring. The clocksource0 is the reference clock (Fclk0 = 25Mhz) and the clocksource1 is the clock that needs to be monitored (Fclk1 = 200Mhz). Seed is the value that gets loaded into the Counter.

In the current example, the XTAL is expected to be a Resonator running in Crystal mode which is later switched off to simulate the clock failure. If an SE Crystal is used, you will need to physically disconnect the clock on the board. Please refer to the TRM for details on counter seed values to be set. Note : When running in flash configuration it is good to do a reset & restart after loading the example to remove any stale flags/states.

External Connections

- None

Watch Variables

- *status/result* - Status of the clock failure detection

10.5 DCC Registers

This section describes the Dual Clock Comparator Registers.

Note

The DCC is used by Boot ROM; hence, the register values could be different than the hardware reset value. You need to make sure to configure the values of these registers to the desired value before enabling the DCC.

10.5.1 DCC Base Address Table (C28)

Table 10-1. DCC Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
Dcc0Regs	DCC_REGS	DCC0_BASE	0x0005_E700	YES	-	-	-	YES
Dcc1Regs	DCC_REGS	DCC1_BASE	0x0005_E740	YES	-	-	-	YES
Dcc2Regs	DCC_REGS	DCC2_BASE	0x0005_E780	YES	-	-	-	YES

10.5.2 DCC_REGS Registers

Table 10-2 lists the memory-mapped registers for the DCC_REGS registers. All register offset addresses not listed in Table 10-2 should be considered as reserved locations and the register contents should not be modified.

Table 10-2. DCC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	DCCGCTRL	Starts / stops the counters. Clears the error signal.		Go
8h	DCCCNTSEED0	Seed value for the counter attached to Clock Source 0.		Go
Ch	DCCVALIDSEED0	Seed value for the timeout counter attached to Clock Source 0.		Go
10h	DCCCNTSEED1	Seed value for the counter attached to Clock Source 1.		Go
14h	DCCSTATUS	Specifies the status of the DCC Module.		Go
18h	DCCCNT0	Value of the counter attached to Clock Source 0.		Go
1Ch	DCCVALID0	Value of the valid counter attached to Clock Source 0.		Go
20h	DCCCNT1	Value of the counter attached to Clock Source 1.		Go
24h	DCCCLKSRC1	Selects the clock source for Counter 1.		Go
28h	DCCCLKSRC0	Selects the clock source for Counter 0.		Go

Complex bit access types are encoded to fit into small table cells. Table 10-3 shows the codes that are used for access types in this section.

Table 10-3. DCC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
R-1	R -1	Read Returns 1s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

10.5.2.1 DCCGCTRL Register (Offset = 0h) [Reset = 5555h]

DCCGCTRL is shown in Figure 10-8 and described in Table 10-4.

Return to the Summary Table.

Starts / stops the counters. Clears the error signal.

Figure 10-8. DCCGCTRL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DONEENA				SINGLESHOT				ERRENA				DCCENA			
R/W-5h				R/W-5h				R/W-5h				R/W-5h			

Table 10-4. DCCGCTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-12	DONEENA	R/W	5h	DONE Enable Enables/disables the done interrupt signal, but has no effect on the done status flag in DCCSTAT register. 0101 The done signal is disabled Others The done signal is enabled Reset type: SYSRSn
11-8	SINGLESHOT	R/W	5h	Single-Shot Enable Enables/disables repetitive operation of the DCC. 1010: Stop counting when COUNTER0 and VALID0 both reach zero 1011: Reserved Others: Continuously repeat (until error) Reset type: SYSRSn
7-4	ERRENA	R/W	5h	Error Enable Enables/disables the error signal. 0101 The error signal is disabled Others The error signal is enabled Reset type: SYSRSn
3-0	DCCENA	R/W	5h	DCC Enable Starts and stops the operation of the DCC. 0101 Counters are stopped Others Counters are running Reset type: SYSRSn

10.5.2.2 DCCCNTSEED0 Register (Offset = 8h) [Reset = 0h]

DCCCNTSEED0 is shown in Figure 10-9 and described in Table 10-5.

Return to the Summary Table.

Seed value for the counter attached to Clock Source 0.

Figure 10-9. DCCCNTSEED0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												COUNTSEED0																			
R-0h												R/W-0h																			

Table 10-5. DCCCNTSEED0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-0	COUNTSEED0	R/W	0h	Seed Value for Counter 0 Contains the seed value that gets loaded into Counter 0 (Clock Source 0). NOTE: Operating the DCC with '0' in the COUNTSEED0 register will result in undefined operation. Reset type: SYSRSn

10.5.2.3 DCCVALIDSEED0 Register (Offset = Ch) [Reset = 0h]

DCCVALIDSEED0 is shown in Figure 10-10 and described in Table 10-6.

Return to the Summary Table.

Seed value for the timeout counter attached to Clock Source 0.

Figure 10-10. DCCVALIDSEED0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																VALIDSEED															
R-0h																R/W-0h															

Table 10-6. DCCVALIDSEED0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	VALIDSEED	R/W	0h	Seed Value for Valid Duration Counter 0 Contains the seed value that gets loaded into the valid duration counter for Clock Source 0. NOTE: Operating the DCC with '0' in the VALIDSEED0 register will result in undefined operation. VALID0 defines a window in which COUNT1 expires. This window is meant to be at least four cycles wide. Do not program a value less than '4' into the VALID0 register. Reset type: SYSRSn

10.5.2.4 DCCCNTSEED1 Register (Offset = 10h) [Reset = 0h]

DCCCNTSEED1 is shown in Figure 10-11 and described in Table 10-7.

Return to the Summary Table.

Seed value for the counter attached to Clock Source 1.

Figure 10-11. DCCCNTSEED1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												COUNTSEED1																			
R-0h												R/W-0h																			

Table 10-7. DCCCNTSEED1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-0	COUNTSEED1	R/W	0h	Seed Value for Counter 1 Contains the seed value that gets loaded into Counter 1 (Clock Source 1). NOTE: Operating the DCC with '0' in the COUNTSEED1 register will result in undefined operation. Reset type: SYSRSn

10.5.2.5 DCCSTATUS Register (Offset = 14h) [Reset = 0h]

DCCSTATUS is shown in Figure 10-12 and described in Table 10-8.

Return to the Summary Table.

Specifies the status of the DCC Module.

Figure 10-12. DCCSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						DONE	ERR
R-0h						R/W-0h	R/W-0h

Table 10-8. DCCSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	DONE	R/W	0h	Single-Shot Done Flag Indicates when single-shot mode is complete without error. Writing a '1' to this bit clears the flag. 0 Single-shot mode has not completed. 1 Single-shot mode has completed. Reset type: SYSRSn
0	ERR	R/W	0h	Error Flag Indicates whether or not an error has occurred. Writing a '1' to this bit clears the flag. 0 No errors have occurred. 1 An error has occurred. Reset type: SYSRSn

10.5.2.6 DCCCNT0 Register (Offset = 18h) [Reset = 0h]

DCCCNT0 is shown in Figure 10-13 and described in Table 10-9.

Return to the Summary Table.

Value of the counter attached to Clock Source 0.

Figure 10-13. DCCCNT0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												COUNT0																			
R-0h												R-0h																			

Table 10-9. DCCCNT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-0	COUNT0	R	0h	Current Value of Counter 0 Reset type: SYSRSn

10.5.2.7 DCCVALID0 Register (Offset = 1Ch) [Reset = 0h]

DCCVALID0 is shown in Figure 10-14 and described in Table 10-10.

Return to the Summary Table.

Value of the valid counter attached to Clock Source 0.

Figure 10-14. DCCVALID0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																VALID0															
R-0h																R-0h															

Table 10-10. DCCVALID0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	VALID0	R	0h	Current Value of Valid 0 Reset type: SYSRSn

10.5.2.8 DCCCNT1 Register (Offset = 20h) [Reset = 0h]

DCCCNT1 is shown in Figure 10-15 and described in Table 10-11.

Return to the Summary Table.

Value of the counter attached to Clock Source 1.

Figure 10-15. DCCCNT1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												COUNT1																			
R-0h												R-0h																			

Table 10-11. DCCCNT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-0	COUNT1	R	0h	Current Value of Counter 1 Reset type: SYSRSn

10.5.2.9 DCCCLKSRC1 Register (Offset = 24h) [Reset = 0h]

DCCCLKSRC1 is shown in Figure 10-16 and described in Table 10-12.

Return to the Summary Table.

Selects the clock source for Counter 1.

Figure 10-16. DCCCLKSRC1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY				RESERVED							CLKSRC1				
R-0/W-0h				R-0h							R/W-0h				

Table 10-12. DCCCLKSRC1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-12	KEY	R-0/W	0h	Enables or Disables Clock Source Write for COUNT1 1010 The CLKSRC field selects the clock source for COUNT1. Others: Previous values retained new writes on register fields has no impact. Reset type: SYSRSn
11-5	RESERVED	R	0h	Reserved
4-0	CLKSRC1	R/W	0h	Clock Source Select for Counter 1 Specifies the clock source for COUNT1, when the KEY field enables this feature. 00000: PLLRAWCLK 00001: AUXPLLRAWCLK 00010: INTOSC1 00011: INTOSC2 00101: CMCLK 00110: CPU1SYSCLK 00111: ETHERNET RXCLK 01000: CPU2SYSCLK 01001: INPUTXBAR (Output15 of the input-xbar) 01010: AUXCLKIN 01011: EPWMCLK 01100: LSPCLK 01101: MII0 RXCLK (etherCAT) 01110: WDCLK 01111: CAN0BITCLK 10111: MII1 RXCLK (etherCAT) Note: All other values are "RSVD" Reset type: SYSRSn

10.5.2.10 DCCCLKSRC0 Register (Offset = 28h) [Reset = 0h]

DCCCLKSRC0 is shown in Figure 10-17 and described in Table 10-13.

Return to the Summary Table.

Selects the clock source for Counter 0.

Figure 10-17. DCCCLKSRC0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY				RESERVED								CLKSRC0			
R-0/W-0h				R-0h								R/W-0h			

Table 10-13. DCCCLKSRC0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-12	KEY	R-0/W	0h	Enables or Disables Clock Source Write for COUNT0 1010: The CLKSRC0 field written with key gets updated to with new selection to clock COUNT0. Others: Previous values retained new writes on register fields has no impact. Reset type: SYSRSn
11-4	RESERVED	R	0h	Reserved
3-0	CLKSRC0	R/W	0h	Clock Source Select for Counter 0 Specifies the clock source for COUNT0, when the KEY field enables this feature. 0000: XTAL/X1 0001: INTOSC1 0010: INTOSC2 0101: CPU1SYSCLK 0110: CPU2SYSCLK 1100: INPUTXBAR (Output16 of the input-xbar) Note: All other values are Reserved Reset type: SYSRSn

10.5.3 DCC Registers to Driverlib Functions

Table 10-14. DCC Registers to Driverlib Functions

File	Driverlib Function
DCCGCTRL	
dcc.h	DCC_enableModule
dcc.h	DCC_disableModule
dcc.h	DCC_enableErrorSignal
dcc.h	DCC_enableDoneSignal
dcc.h	DCC_disableErrorSignal
dcc.h	DCC_disableDoneSignal
dcc.h	DCC_enableSingleShotMode
dcc.h	DCC_disableSingleShotMode
DCCCNTSEED0	
dcc.h	DCC_setCounterSeeds
DCCVALIDSEED0	
dcc.h	DCC_setCounterSeeds
DCCCNTSEED1	

Table 10-14. DCC Registers to Driverlib Functions (continued)

File	Driverlib Function
dcc.h	DCC_setCounterSeeds
DCCSTATUS	
dcc.h	DCC_getErrorStatus
dcc.h	DCC_getSingleShotStatus
dcc.h	DCC_clearErrorFlag
dcc.h	DCC_clearDoneFlag
sysctl.c	SysCtl_isPLLValid
DCCCNT0	
dcc.h	DCC_getCounter0Value
DCCVALID0	
dcc.h	DCC_getValidCounter0Value
DCCCNT1	
dcc.h	DCC_getCounter1Value
DCCCLKSRC1	
dcc.h	DCC_setCounter1ClkSource
dcc.h	DCC_getCounter1ClkSource
DCCCLKSRC0	
dcc.h	DCC_setCounter0ClkSource
dcc.h	DCC_getCounter0ClkSource

Chapter 11

Direct Memory Access (DMA)

The direct memory access (DMA) module provides a hardware method of transferring data between peripherals and memory without intervention from the CPU; thereby, freeing up bandwidth for other system functions. Additionally, the DMA has the capability to orthogonally rearrange the data as the data is transferred as well as “ping-pong” data between buffers. These features are useful for structuring data into blocks for CPU processing.

11.1 Introduction

The strength of a controller is not measured purely in processor speed, but in total system capabilities. As a part of the equation, any time the CPU bandwidth for a given function can be reduced, the greater the system capabilities. Many times applications spend a significant amount of their bandwidth moving data, whether moving data from off-chip memory to on-chip memory, from a peripheral such as an analog-to-digital converter (ADC) to RAM, or from one peripheral to another. Furthermore, many times this data comes in a format that is not conducive to the optimum processing powers of the CPU. The DMA module described in this chapter has the ability to free up CPU bandwidth and rearrange the data into a pattern for more streamlined processing.

The DMA module is an event-based machine, meaning the DMA module requires a peripheral or software trigger to start a DMA transfer. Although the DMA module can be made into a periodic time-driven machine by configuring a timer as the DMA trigger source, there is no mechanism within the module itself to start memory transfers periodically. The DMA module has six independent DMA channels that can be configured separately and each channel contains their own independent PIE interrupt to let the CPU know when a DMA transfer has either started or completed. Five of the six channels are exactly the same, while Channel 1 has the ability to be configured at a higher priority than the others. At the heart of the DMA is a state machine and tightly coupled address control logic. This address control logic allows for rearrangement of the block of data during the transfer as well as the process of ping-ponging data between buffers. Each of these features is discussed in detail in this chapter.

11.1.1 Features

DMA features include:

- Six channels with independent PIE interrupts
- Each DMA channel can be triggered from multiple peripheral trigger sources independently
- Word Size: 16-bit or 32-bit (SPI limited to 16-bit)
- Throughput: 3 cycles/word without arbitration

11.1.2 Block Diagram

Figure 11-1 shows a device-level block diagram of the DMA.

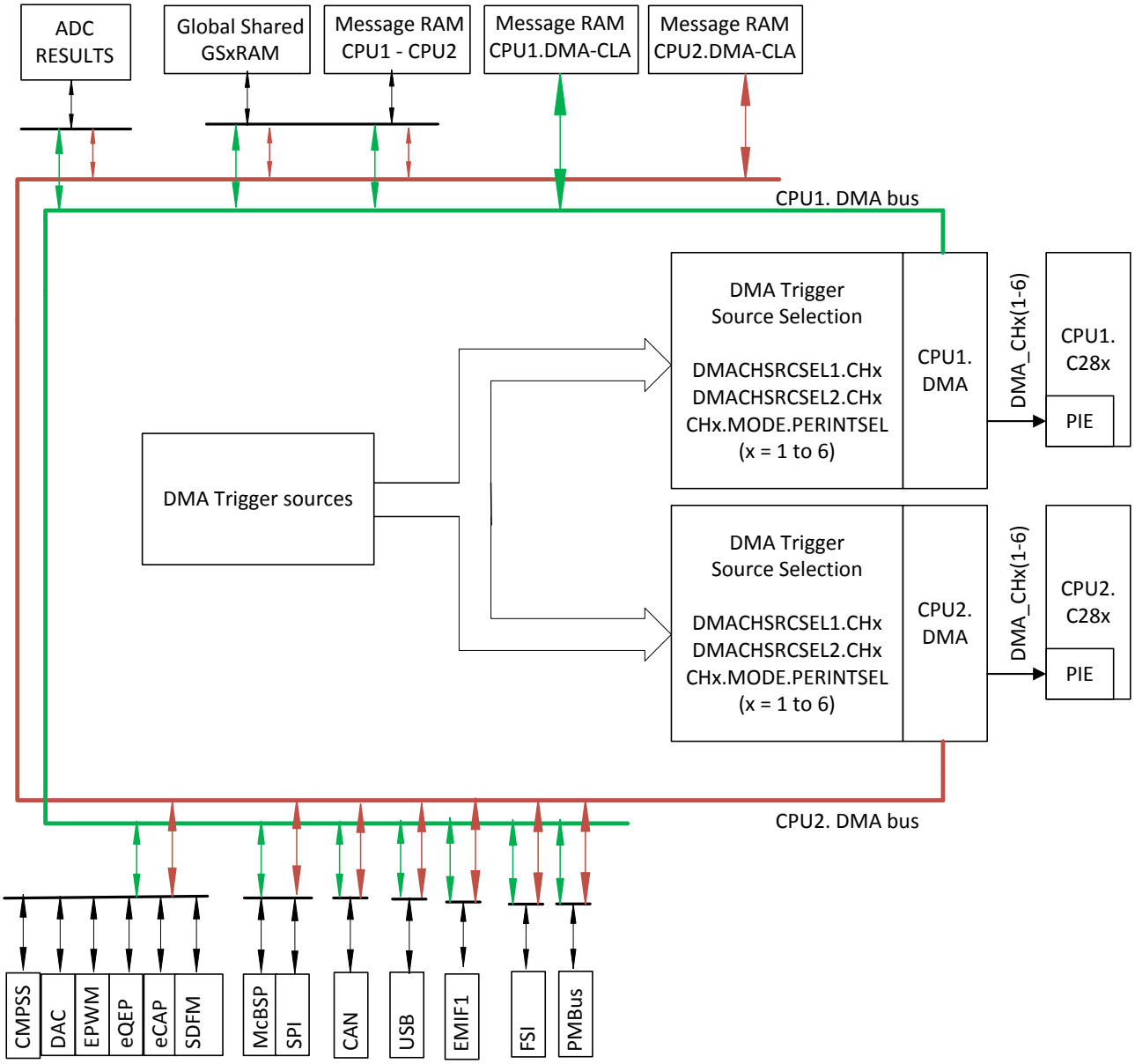


Figure 11-1. DMA Block Diagram

11.2 Architecture

11.2.1 Peripheral Interrupt Event Trigger Sources

Each DMA Channel can be configured to trigger by software and other peripheral triggers events. DMACHSRCSELx register can be used to configure DMA Trigger sources for each DMA channel. CHx.MODE.PERINTSEL register bit field can be set to channel number (CHx.MODE.PERINTSEL = x) as shown in Figure 11-2. Included in these DMA Trigger sources are five external interrupt signals that can be connected to most of the general-purpose input/output (GPIO) pins on the device. This adds significant flexibility to the event trigger capabilities. Upon receipt of a peripheral interrupt event signal, the DMA automatically sends a clear signal to the interrupt source so that subsequent interrupt events occur.

Note

To use the system-level DMA Trigger source selection, the DMA internal trigger source selection configuration for each channel can be done using the DMACHSRCSELx register and the CHx.MODE.PERINTSEL register. See Table 11-1 or the DMACHSRCSELx register definition for a complete list of DMA trigger sources.

Regardless of the value of the MODE.CHx[PERINTSEL] bit field, software can always force a trigger by using the CONTROL.CHx[PERINTFRC] bit. Likewise, software can always clear a pending DMA trigger using the CONTROL.CHx[PERINTCLR] bit.

Once a particular peripheral trigger event sets a channel's PERINTFLG bit, the bit remains pending until the priority logic of the state machine starts the burst transfer for that channel. Once the burst transfer starts, the flag is cleared. If a new peripheral trigger event is generated while a burst is in progress, the burst completes before responding to the new peripheral trigger event (after proper prioritization). If a third peripheral trigger event occurs before the pending event is serviced, an error flag is set in the CONTROL.CHx[OVRFLG] bit. If a peripheral trigger event occurs at the same time as the latched flag is being cleared, the trigger event has priority and the PERINTFLG remains set.

Figure 11-3 shows a diagram of the trigger select circuit.

Table 11-1 shows the peripheral trigger source options that are available for each channel.

CAUTION

See the Device Errata "ADC:DMA Read of Stale Result" Advisory regarding the potential for the DMA to read the ADC result registers before the result is ready

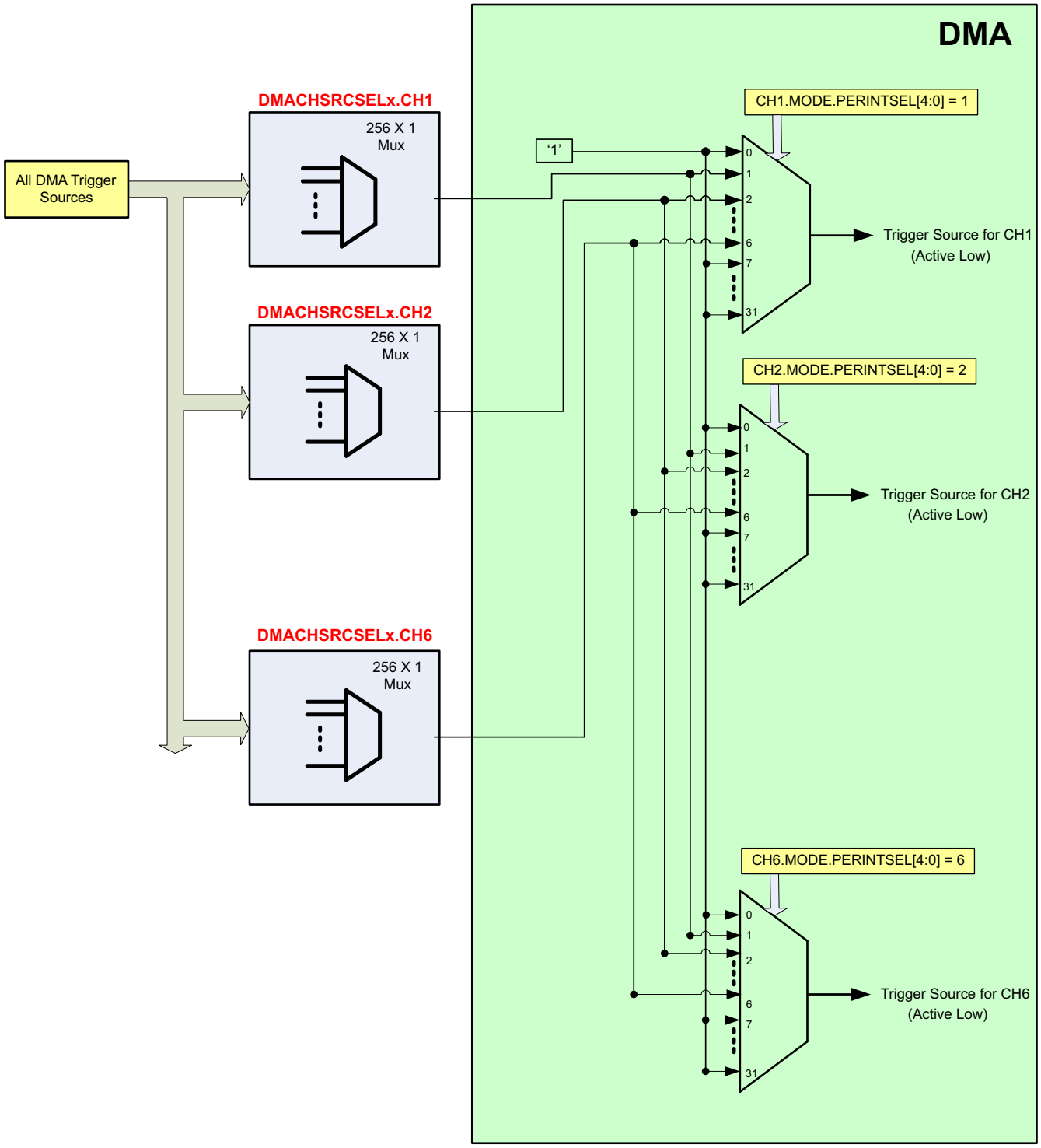
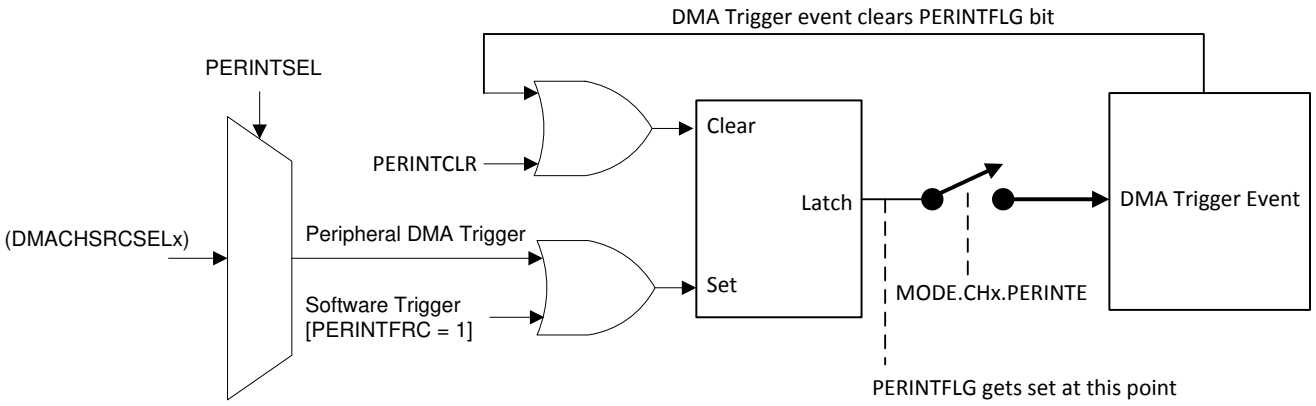


Figure 11-2. DMA Trigger Architecture



Note: See Figure 11-2.

Figure 11-3. Peripheral Interrupt Trigger Input Diagram

Table 11-1. DMA Trigger Source Options

Select Index	Trigger Source
0	DMA_SOFTWARE_TRIGGER
1	ADCAINT1_DMA
2	ADCAINT2_DMA
3	ADCAINT3_DMA
4	ADCAINT4_DMA
5	ADCAEVT
6	ADCBINT1_DMA
7	ADCBINT2_DMA
8	ADCBINT3_DMA
9	ADCBINT4_DMA
10	ADCBEVT
11	ADCCINT1_DMA
12	ADCCINT2_DMA
13	ADCCINT3_DMA
14	ADCCINT4_DMA
15	ADCCEVT
16	ADCDINT1_DMA
17	ADCDINT2_DMA
18	ADCDINT3_DMA
19	ADCDINT4_DMA
20	ADCDEVT
21-28	Reserved
29	XINT1
30	XINT2
31	XINT3
32	XINT4
33	XINT5
34-35	Reserved
36	EPWM1_SOCA
37	EPWM1_SOCB
38	EPWM2_SOCA

Table 11-1. DMA Trigger Source Options (continued)

Select Index	Trigger Source
39	EPWM2_SOCB
40	EPWM3_SOCA
41	EPWM3_SOCB
42	EPWM4_SOCA
43	EPWM4_SOCB
44	EPWM5_SOCA
45	EPWM5_SOCB
46	EPWM6_SOCA
47	EPWM6_SOCB
48	EPWM7_SOCA
49	EPWM7_SOCB
50	EPWM8_SOCA
51	EPWM8_SOCB
52	EPWM9_SOCA
53	EPWM9_SOCB
54	EPWM10_SOCA
55	EPWM10_SOCB
56	EPWM11_SOCA
57	EPWM11_SOCB
58	EPWM12_SOCA
59	EPWM12_SOCB
60	EPWM13_SOCA
61	EPWM13_SOCB
62	EPWM14_SOCA
63	EPWM14_SOCB
64	EPWM15_SOCA
65	EPWM15_SOCB
66	EPWM16_SOCA
67	EPWM16_SOCB
68	CPU1_TINT0
69	CPU1_TINT1
70	CPU1_TINT2
71	MCBSPA_XEVT
72	MCBSPA_REVT
73	MCBSPB_XEVT
74	MCBSPB_REVT
75	ECAP1_DMA
76	ECAP2_DMA
77	ECAP3_DMA
78	ECAP4_DMA
79	ECAP5_DMA
80	ECAP6_DMA
81	ECAP7_DMA
82-94	Reserved
95	SD1FLT1_DRINT

Table 11-1. DMA Trigger Source Options (continued)

Select Index	Trigger Source
96	SD1FLT2_DRINT
97	SD1FLT3_DRINT
98	SD1FLT4_DRINT
99	SD2FLT1_DRINT
100	SD2FLT2_DRINT
101	SD2FLT3_DRINT
102	SD2FLT4_DRINT
103	SYNC_DMA_TRIG
104-108	Reserved
109	SPIA_TXDMA
110	SPIA_RXDMA
111	SPIB_TXDMA
112	SPIB_RXDMA
113	SPIC_TXDMA
114	SPIC_RXDMA
115	SPID_TXDMA
116	SPID_RXDMA
117	CLB5_INT
118	CLB6_INT
119	CLB7_INT
120	CLB8_INT
121-122	Reserved
123	FSITXA_DMA
124	Reserved
125	FSIRXA_DMA
126	Reserved
127	CLB1_INT
128	CLB2_INT
129	CLB3_INT
130	CLB4_INT
131	USBA_EPX_RX1
132	USBA_EPX_TX1
133	USBA_EPX_RX2
134	USBA_EPX_TX2
135	USBA_EPX_RX3
136	USBA_EPX_TX3
137-142	Reserved
143	FSIRXC_DMA
144	FSIRXD_DMA
145	FSIRXE_DMA
146	FSIRXF_DMA
147	FSIRXG_DMA
148	FSIRXH_DMA
149-154	Reserved
155	FSITXB_DMA

Table 11-1. DMA Trigger Source Options (continued)

Select Index	Trigger Source
156	Reserved
157	FSIRXB_DMA
158-166	Reserved
167	CANA_IF1
168	CANA_IF2
169	CANA_IF3
170	CANA_IF1
171	CANA_IF2
172	CANA_IF3
173-255	Reserved

11.2.2 DMA Bus

The DMA bus architecture consists of a 32-bit address bus, a 32-bit data read bus, and a 32-bit data write bus. Memories and register locations connected to the DMA bus by way of interfaces that sometimes share resources with the CPU memory or peripheral bus. Arbitration rules are defined in Section 11.5.

11.3 Address Pointer and Transfer Control

The DMA state machine is, at the most basic level, two nested loops.

Burst (Inner) Loop:

The burst (inner) loop transfers a programmable number of words set by (BURST_SIZE + 1) register when a DMA channel trigger (Peripheral or Software trigger) is received. The BURST_SIZE register allows a maximum of 32 sixteen-bit words to be transferred in one burst. Each DMA channel supports both 16-bit or 32-bit word burst that can be controlled by MODE.DATASIZE bit field. Each DMA channel contains a shadowed address pointer for the source (SRC_ADDR_SHADOW) and the destination (DST_ADDR_SHADOW) address. At the beginning of each transfer, the shadowed version of each pointer is copied into the respective active (SRC_ADDR_ACTIVE or DST_ADDR_ACTIVE) register. During the burst loop, after each word is transferred, the signed value contained in the appropriate source or destination BURST_STEP register is added to the active register:

SRC_ADDR_ACTIVE = SRC_ADDR_ACTIVE + SRC_BURST_STEP

DST_ADDR_ACTIVE = DST_ADDR_ACTIVE + DST_BURST_STEP

The burst (inner) loop transfers a burst of data when a DMA Channel Trigger (Peripheral or Software trigger) is received.

Transfer (Outer) Loop:

The Transfer (outer) loop transfers a programmable number of bursts set by (TRANSFER_SIZE + 1) register for each channel. Since TRANSFER_SIZE is a 16-bit register, the total size of a transfer allowed is well beyond any practical requirement. During the transfer loop, after each burst is complete, there are two methods that can be used to modify the active address pointer:

Method 1 (Default): When address wrapping is disabled (SRC_WRAP_SIZE or DST_WRAP_SIZE is greater than TRANSFER_SIZE), active address pointer is updated as shown below

SRC_ADDR_ACTIVE = SRC_ADDR_ACTIVE + SRC_TRANSFER_STEP

DST_ADDR_ACTIVE = DST_ADDR_ACTIVE + DST_TRANSFER_STEP

Method 2: Address wrapping gets enabled when SRC_WRAP_SIZE or DST_WRAP_SIZE is less than TRANSFER_SIZE. This allows the channel to wrap multiple times within a single transfer. When the number of

bursts is equal to (SRC/DST_WRAP_SIZE + 1) register, the state machine modifies the active address pointers as:

SRC_BEG_ADDR_ACTIVE = SRC_BEG_ADDR_ACTIVE + SRC_WRAP_STEP

DST_BEG_ADDR_ACTIVE = DST_BEG_ADDR_ACTIVE + DST_WRAP_STEP

SRC_ADDR_ACTIVE = SRC_BEG_ADDR_ACTIVE

DST_ADDR_ACTIVE = DST_BEG_ADDR_ACTIVE

At the end of DMA transfer, DMA can have transferred (BURST_SIZE + 1) x (TRANSFER_SIZE + 1) words.

OneShot Mode:

OneShot mode is disabled by default.

When OneShot mode is disabled (MODE.CHx[ONESHOT] = 0), DMA transfers one burst [(BURST_SIZE + 1) words] of data each time a DMA Channel Trigger is received. After the burst is completed, the state machine moves on to the next pending channel in the priority scheme, even if another trigger for the channel just completed is pending. This feature keeps any single channel from monopolizing the DMA bus.

When OneShot mode is enabled (MODE.CHx[ONESHOT] = 1), DMA transfers all the bursts [(BURST_SIZE + 1) x (TRANSFER_SIZE + 1) words] on a single DMA channel trigger. Be careful when using this mode, since this can create a condition where one trigger uses up the majority of the DMA bandwidth.

Continuous Mode:

Continuous mode is disabled by default.

When Continuous mode is disabled (MODE.CHx[CONTINUOUS] = 0), DMA state machine disables channel after all bursts in a transfer loop (TRANSFER_COUNT = 0) are complete. The channel must be re-enabled by setting the RUN bit in the CONTROL register before another transfer can be started on that channel.

When Continuous mode is enabled (MODE.CHx[CONTINUOUS] = 1), DMA state machine keep channel active even after all bursts in a transfer loop (TRANSFER_COUNT = 0) are complete.

Each DMA channel can trigger their own EPIE interrupt for each DMA transfer either at start of DMA transfer or end of DMA transfer using MODE.CHx[CHINTMODE] bit.

Source/Destination Address Pointers (SRC/DST_ADDR)	The value written into the shadow register is the start address of the first location where data is read or written to. At the beginning of a transfer the shadow register (SRC/DST_ADDR_SHADOW) is copied into the active register (SRC/DST_ADDR_ACTIVE). The active register performs as the current address pointer.
Source/Destination Begin Address Pointers (SRC/DST_BEG_ADDR)	This is the wrap pointer. The value written into the shadow register (SRC/DST_BEG_ADDR_SHADOW) is loaded into the active register (SRC/DST_BEG_ADDR_ACTIVE) at the start of a transfer. On a wrap condition, the active register (SRC/DST_BEG_ADDR_ACTIVE) is incremented by the signed value in the appropriate SRC/DST_WRAP_STEP register prior to being loaded into the active register (SRC/DST_ADDR_ACTIVE).

For each channel, the transfer process can be controlled with the following size values:

Source and Destination Burst Size (BURST_SIZE)

This specifies the number of words to be transferred in a burst.

This value is loaded into the BURST_COUNT register at the beginning of each burst. The BURST_COUNT decrements each word that is transferred and when the register reaches a zero value, the burst is complete, indicating that the next channel can be serviced. The behavior of the current channel is defined by the ONE_SHOT bit in the MODE register. The maximum size of the burst is dictated by the type of peripheral. For the ADC, the burst size can be all 16 registers (if all 16 registers are used). For RAM, the burst size can be up to the maximum allowed by the BURST_SIZE register, which is 32. See Table 11-2 to understand how BURST_SIZE register affects the number of 16-bit words transferred with respect to DATASIZE.

Table 11-2. BURSTSIZE versus DATASIZE Behavior

BURSTSIZE	Number of 16-bit words transferred in	
	DataSize = 16-bit data	DataSize = 32-bit data
0	1	2
1	2	2
2	3	4
3	4	4
4	5	6
5	6	6
6	7	8
7	8	8
8	9	10
9	10	10
10	11	12
11	12	12
*	*	*
*	*	*
*	*	*
30	31	32
31	32	32

Source and Destination Transfer Size (TRANSFER_SIZE)

This specifies the number of bursts to be transferred per CPU interrupt (if enabled).

Whether this interrupt is generated at the beginning or the end of the transfer is defined in the CHINTMODE bit in the MODE register. Whether the channel remains enabled or not after the transfer is completed is defined by the CONTINUOUS bit in the MODE register. The TRANSFER_SIZE register is loaded into the TRANSFER_COUNT register at the beginning of each transfer. The TRANSFER_COUNT register keeps track of how many bursts of data the channel has transferred and when the register reaches zero, the DMA transfer is complete.

Source/Destination Wrap Size (SRC/ DST_WRAP_SIZE)

This specifies the number of bursts to be transferred before the current address pointer wraps around to the beginning.

This feature is used to implement a circular addressing type function. This value is loaded into the appropriate SRC/DST_WRAP_COUNT register at the beginning of each transfer. The SRC/DST_WRAP_COUNT registers keep track of how many bursts of data the channel has transferred and when the registers reach zero, the wrap procedure is performed on the appropriate source or destination address pointer. A separate size and count register is allocated for source and destination pointers. To disable the wrap function, assign the value of these registers to be larger than the TRANSFER_SIZE.

Note

The value written to the SIZE registers is one less than the intended size. So, to transfer three 16-bit words, the value 2 can be placed in the SIZE register.

Regardless of the state of the DATASIZE bit, the value specified in the SIZE registers are for 16-bit addresses. So, to transfer three 32-bit words, the value 5 can be placed in the SIZE register.

For each source/destination pointer, the address changes can be controlled with the following step values:

Source/Destination Burst Step (SRC/ DST_BURST_STEP)

Within each burst transfer, the address source and destination step sizes are specified by these registers.

This value is a signed 2s compliment number so that the address pointer can be incremented or decremented as required. If no increment is desired, such as when accessing the data receive or transmit registers in a communication peripheral, the value of these registers can be set to zero.

Source/Destination Transfer Step (SRC/ DST_TRANSFER_STEP)

This specifies the address offset to start the next burst transfer after completing the current burst transfer.

This is used in cases where registers or data memory locations are spaced at constant intervals. This value is a signed 2s compliment number so that the address pointer can be incremented or decremented as required.

Source/Destination Wrap Step (SRC/ DST_WRAP_STEP)

When the wrap counter reaches zero, this value specifies the number of words to add/subtract from the SRC/DST_BEG_ADDR pointer and hence sets the new start address.

This implements a circular type of addressing mode, useful in many applications. This value is a signed 2s compliment number so that the address pointer can be incremented or decremented as required.

Note

Regardless of the state of the DATASIZE bit, the value specified in the STEP registers are for 16-bit addresses. So, to increment one 32-bit address, a value of 2 can be placed in these registers.

Channel Interrupt Mode (CHINTMODE)

This mode bit selects whether the DMA interrupt from the respective channel is generated at the beginning of a new transfer or at the end of the transfer.

If implementing a ping-pong buffer scheme with continuous mode of operation, then the interrupt can be generated at the beginning, just after the working registers are copied to the shadow set. If the DMA does not operate in continuous mode, then the interrupt is typically generated at the end when the transfer is complete.

All of the previous features and modes are shown in Figure 11-4. The following items are in reference to Figure 11-4.

- The *HALT* points represent where the channel halts operation when interrupted by a high priority channel 1 trigger, or when the HALT command is set, or when an emulation halt is issued and the FREE bit is cleared to 0.
- The SRC/DST_ADDR_ACTIVE registers are not affected by SRC/DST_BEG_ADDR_ACTIVE at the start of a transfer. SRC/DST_BEG_ADDR_ACTIVE only affects the SRC/DST_ADDR_ACTIVE registers on a wrap. Following is what happens when a transfer first starts:
  - SRC/DST_BEG_ADDR_SHADOW remains unchanged.
  - SRC/DST_ADDR_SHADOW remains unchanged.
  - SRC/DST_BEG_ADDR_ACTIVE = SRC/DST_BEG_ADDR_SHADOW
  - SRC/DST_ADDR_ACTIVE = SRC/DST_ADDR_SHADOW
- The active registers get updated when a wrap occurs. The shadow registers remain unchanged. Specifically:
  - SRC/DST_BEG_ADDR_SHADOW remains unchanged.
  - SRC/DST_ADDR_SHADOW remains unchanged.
  - SRC/DST_BEG_ADDR_ACTIVE += SRC/DST_WRAP_STEP
  - SRC/DST_ADDR_ACTIVE = SRC/DST_BEG_ADDR_ACTIVE
- The best way to remember this is:
  - The shadow registers never change except by software.
  - The active registers never change except by hardware, and a shadow register is only copied into their own active register, never an active register by another name.

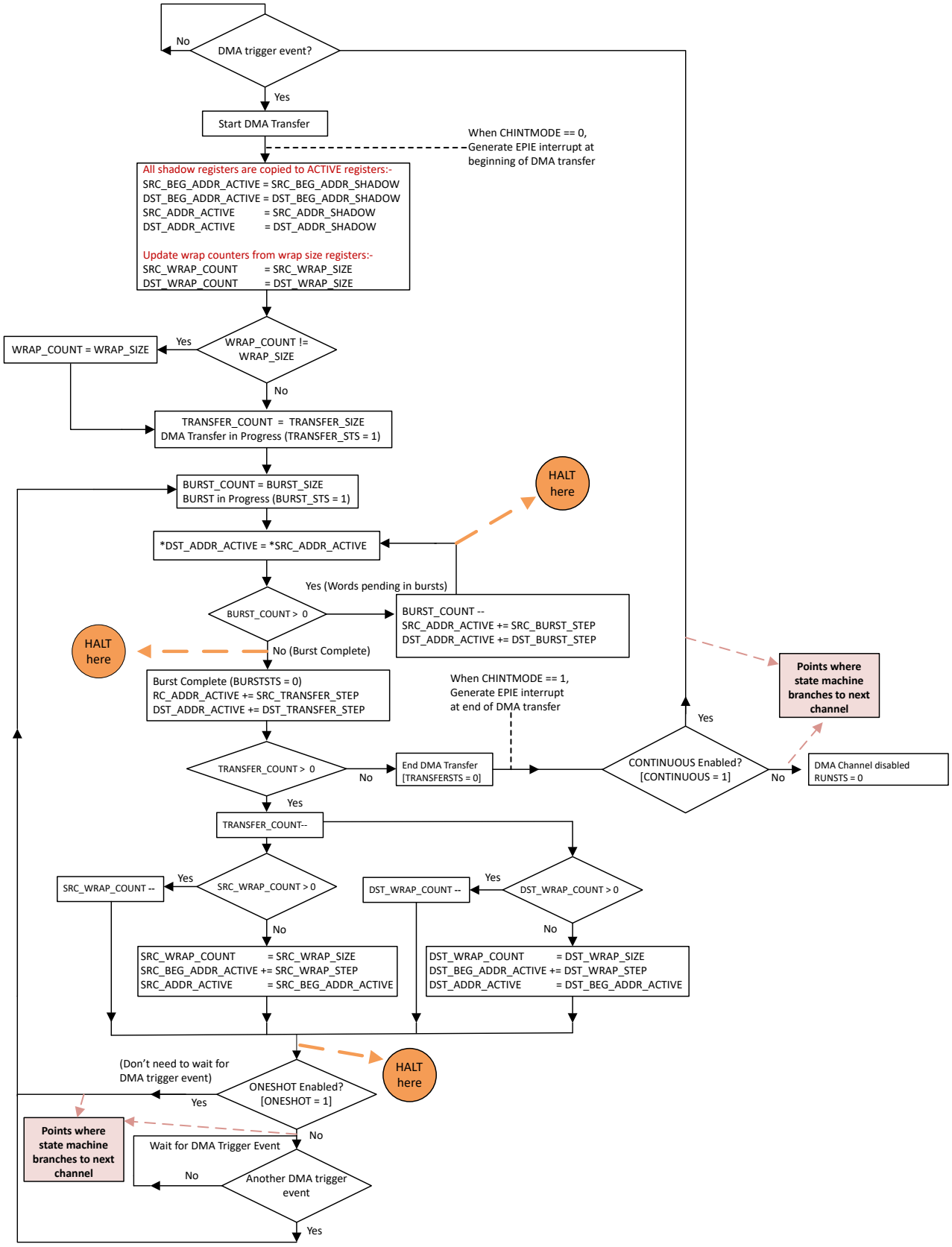


Figure 11-4. DMA State Diagram

11.4 Pipeline Timing and Throughput

In addition to the pipeline there are a few other behaviors of the DMA that affect the total throughput:

- A 1-cycle delay is added at the beginning of each burst
- A 1-cycle delay is added when returning from a CH1 high-priority interrupt
- Collisions with the CPU can add delay slots (see Section 11.5)
- 32-bit transfers run at double the speed of a 16-bit transfer (takes the same amount of time to transfer a 32-bit word as to transfer a 16-bit word)

For example, to transfer 128 16-bit words from GS0 RAM to GS3 RAM, a channel can be configured to transfer 8 bursts of 16 words/burst. This gives:

8 bursts * [(3 cycles/word * 16 words/burst) + 1] = 392 cycles

If instead the channel were configured to transfer the same amount of data 32 bits at a time (the word size is configured to 32 bits), the transfer can take:

8 bursts * [(3 cycles/word * 8 words/burst) + 1] = 200 cycles

The DMA module consists of a 3-stage pipeline as shown in Figure 11-5 and Figure 11-6.

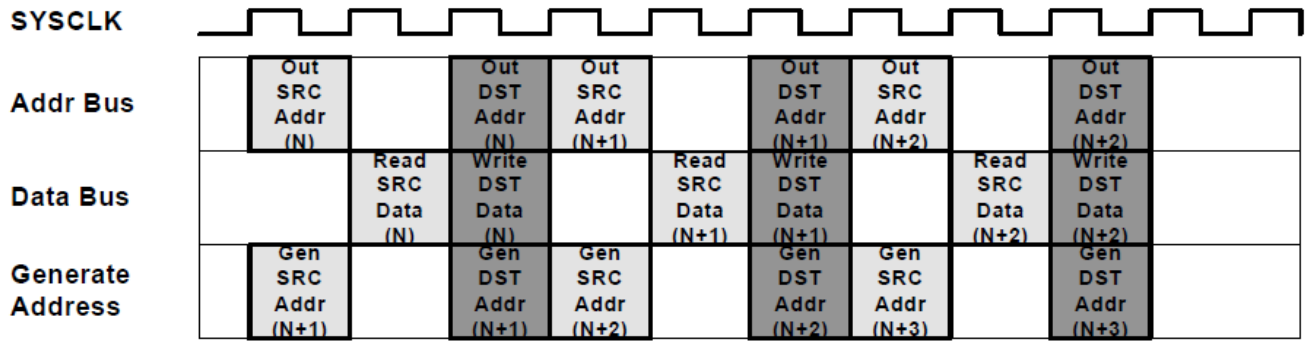


Figure 11-5. 3-Stage Pipeline DMA Transfer

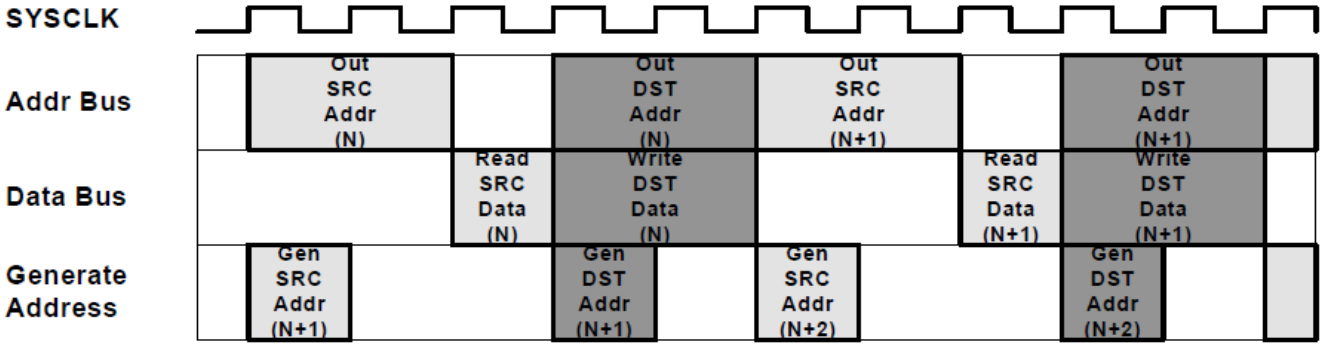


Figure 11-6. 3-stage Pipeline with One Read Stall

11.5 CPU and CLA Arbitration

Typically, DMA activity is independent of CPU and CLA activity. However, when the DMA and CPU (or CLA) try to access the same peripheral at the same time, an arbitration procedure is required to resolve the conflict. All instances of the same peripheral type conflict with each other. For instance, CAN-A and CAN-B conflict. Accesses to global shared RAM, across different instances, do not have this conflict. Different peripheral types can share a bus interface, which creates further opportunities for conflicts. These bus interfaces are:

- Peripheral frame 1: ePWM, eCAP, eQEP, CMPSS, DAC , SDFM
- Peripheral frame 2: PMBus, SPI , FSI

Conflict Example: The CLA is accessing DAC-A while the DMA is simultaneously accessing DAC-B.

Conflict Example: The CPU is accessing an SPI FIFO while the DMA is simultaneously accessing a PMBus register.

Non-conflict Example: The CPU is accessing a shared ePWM while the DMA is accessing an SPI.

Non-conflict Example: The CPU is accessing GS0 while the DMA is accessing GS1

The exception to all this is the ADC result registers, which are duplicated for each bus master. The CPU, DMA, and CLA can all simultaneously read these result registers with no stalls or arbitration needed for any master.

A DMA transfer consists of four phases: send source address, read source data, send destination address, and write destination data (see Section 11.4). Suppose CPU accesses a peripheral/memory causing conflict in middle of a DMA transfer, CPU is stalled till the current DMA access is complete and not until the completion of whole DMA transfer.

The following priority schemes are implemented for the various interfaces on the device:

- The arbitration follows a fixed arbitration scheme with highest priority first:
  - DMA WRITE
  - DMA READ
  - CLA WRITE
  - CLA READ
  - CPU WRITE
  - CPU READ
- The priority scheme for GSx RAM accesses is round-robin.
- The ADC results are duplicated for CPU, CLA, and DMA so that no arbitration is needed when reading the result registers. This allows all masters to access the ADC result registers simultaneously without delay.

Note

If the CPU is performing a read-modify-write operation and the DMA performs a write to the same location, the DMA write may be lost if the operation occurs in between the CPU read and the CPU write. Avoid mixing CPU writes with DMA writes to the same locations.

Arbitration within DMA channels is based on a round-robin priority (or) Channel 1 high priority scheme described in Section 11.6.

11.6 Channel Priority

Two priority schemes exist when determining channel priority: Round-robin mode and Channel 1 high-priority mode.

11.6.1 Round-Robin Mode

In this mode, all channels have *equal* priority and each enabled channel is serviced in round-robin fashion as:

CH1 → CH2 → CH3 → CH4 → CH5 → CH6 → CH1 → CH2 → …

In the case above, after each channel has transferred a burst of words, the next channel is serviced. The user can specify the size of the burst for each channel. Once CH6 (or the last enabled channel) has been serviced, and no other channels are pending, the round-robin state machine enters an idle state.

From the idle state, channel 1 (if enabled) is always serviced first. However, if the DMA is currently processing another channel x, all other pending channels between x and the end of the round are serviced before CH1. All the channels are of *equal* priority. For instance, take an example where CH1, CH4, and CH5 are enabled in round-robin mode and CH4 is currently being processed. Then CH1 and CH5 both receive an interrupt trigger from their respective peripherals before CH4 completes. CH1 and CH5 are now both pending. When CH4 completes the burst, CH5 is serviced next. Only after CH5 completes is CH1 serviced. Upon completion of CH1, if there are no more channels pending, the round-robin state machine enters an idle state.

A more complicated example is:

- Assume all channels are enabled, and the DMA is in an idle state,
- Initially a trigger occurs on CH1, CH3, and CH5 on the same cycle,
- When the CH1 burst transfer starts, requests from CH3 and CH5 are pending,
- Before completion of the CH1 burst, the DMA receives a request from CH2. Now the pending requests are from CH2, CH3, and CH5,
- After completing the CH1 burst, CH2 is serviced since this channel is next in the round-robin scheme after CH1.
- After the burst from CH2 is finished, the CH3 burst is serviced, followed by CH5 burst.
- Now while the CH5 burst is being serviced, the DMA receives a request from CH1, CH3, and CH6.
- The burst from CH6 starts after the completion of the CH5 burst, since this channel is the next channel after CH5 in the round-robin scheme.
- This is followed by the CH1 burst and then the CH3 burst
- After the CH3 burst finishes, assuming no more triggers have occurred, the round-robin state machine enters an idle state.

The round-robin state machine can be reset to the idle state using the DMACTRL[PRIORITYRESET] bit.

11.6.2 Channel 1 High-Priority Mode

In this mode, Channel 1 has high priority over all the other channels. Channels 2 to 6 have equal priority and each enabled channel is serviced in a round-robin fashion.

Higher priority:	CH1
Lower priority:	CH2 → CH3 → CH4 → CH5 → CH6 → CH2 → …

Given an example where CH1, CH4, and CH5 are enabled in Channel 1 high-priority mode and CH4 is currently being processed. Then CH1 and CH5 both receive an interrupt trigger from their respective peripherals before CH4 completes. CH1 and CH5 are now both pending. When the current CH4 word transfer is completed, regardless of whether the DMA has completed the entire CH4 burst, CH4 execution is suspended and CH1 is serviced. After the CH1 burst completes, CH4 resumes execution.

Upon completion of CH4, CH5 is serviced. After CH5 completes, if there are no more channels pending, the round-robin state machine enters an idle state.

Typically Channel 1 is used in this mode for the ADC, since the data rate is so high. However, Channel 1 high-priority mode can be used in conjunction with any peripheral.

Note

High-priority mode and ONESHOT mode cannot be used at the same time on Channel 1. Other channels can use ONESHOT mode when Channel 1 is in high-priority mode.

11.7 Overrun Detection Feature

The DMA contains overrun detection logic. When a peripheral event trigger is received by the DMA, the PERINTFLG bit in the CONTROL register is set, pending the channel to the DMA state machine. When the burst for that channel is started, the PERINTFLG is cleared. If however, between the time that the PERINTFLG bit is set by an event trigger and cleared by the start of the burst, an additional event trigger arrives, the second trigger is lost. This condition sets the OVRFLG bit in the CONTROL register as in Figure 11-7. If the overrun interrupt is enabled, the channel interrupt is generated to the PIE module.

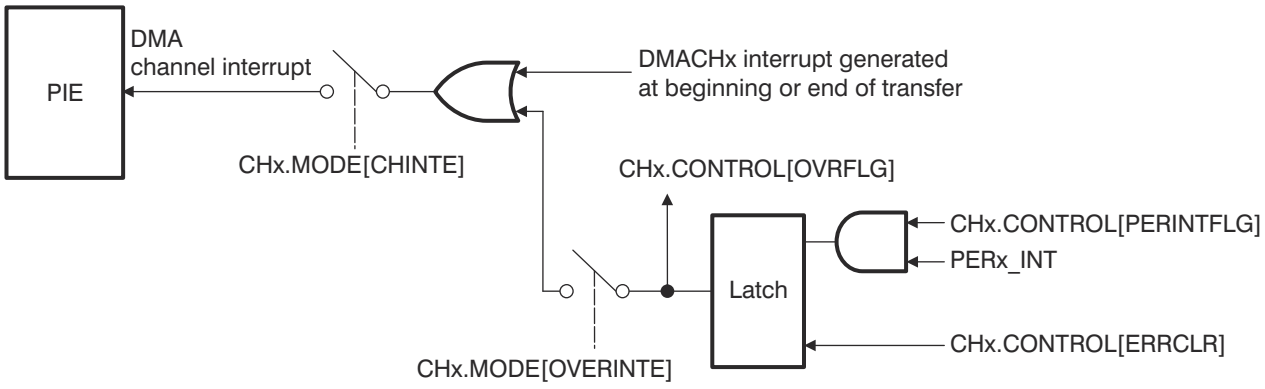


Figure 11-7. Overrun Detection Logic

11.8 Software

11.8.1 DMA Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/dma

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

11.8.1.1 DMA GSRAM Transfer (dma_ex1_gsram_transfer)

FILE: dma_ex1_gsram_transfer.c

This example uses one DMA channel to transfer data from a buffer in RAMGS0 to a buffer in RAMGS1. The example sets the DMA channel PERINTFRC bit repeatedly until the transfer of 16 bursts (where each burst is 8 16-bit words) has been completed. When the whole transfer is complete, it will trigger the DMA interrupt.

Watch Variables
- *sData* - Data to send
- *rData* - Received data

11.8.1.2 DMA Transfer Shared Peripheral - C28X_DUAL

FILE: dma_ex1_shared_periph_cpu1.c

This example shows how to initiate a DMA transfer on CPU1 from a shared peripheral which is owned by CPU2. In this specific example, a timer ISR is used on CPU2 to initiate a SPI transfer which will trigger the CPU1 DMA. CPU1's DMA will then in turn update the ePWM1 CMPA value for the PWM which it owns. The PWM output can be observed on the GPIO pins. It is recommended to run the c28x1 core first, followed by the C28x2 core.

Watch Pins
- GPIO0 and GPIO1 - ePWM output can be viewed with oscilloscope

11.8.1.3 DMA Transfer for Shared Peripheral Example (CPU2) - C28X_DUAL

FILE: dma_ex1_shared_periph_cpu2.c

This example shows how to initiate a DMA transfer on CPU1 from a shared peripheral that is owned by CPU2. In this specific example, a timer ISR is used on CPU2 to initiate a SPI transfer that triggers the CPU1 DMA. CPU1 DMA then updates the ePWM1 CMPA value for the PWM that the DMA owns. The PWM output can be observed on the GPIO pins. It is recommended to run the c28x1 core first, followed by the C28x2 core.

11.8.1.4 DMA GSRAM Transfer (dma_ex2_gsram_transfer)

FILE: dma_ex2_gsram_transfer.c

This example uses one DMA channel to transfer data from a buffer in RAMGS0 to a buffer in RAMGS1. The example sets the DMA channel PERINTFRC bit repeatedly until the transfer of 16 bursts (where each burst is 8 16-bit words) has been completed. When the whole transfer is complete, it will trigger the DMA interrupt.

Watch Variables
- *sData* - Data to send
- *rData* - Received data

11.8.1.5 DMA Transfer Shared Peripheral - C28X_DUAL

FILE: dma_ex2_sysconfig_cpu1.c

This example shows how to initiate a DMA transfer on CPU1 from a shared peripheral which is owned by CPU2. In this specific example, a timer ISR is used on CPU2 to initiate a SPI transfer which will trigger the CPU1 DMA. CPU1's DMA will then in turn update the ePWM1 CMPA value for the PWM which it owns. The PWM output can be observed on the GPIO pins. It is recommended to run the c28x1 core first, followed by the C28x2 core.

Watch Pins

- GPIO0 and GPIO1 - ePWM output can be viewed with oscilloscope

11.9 DMA Registers

This section describes the C28x Direct Memory Access Registers.

11.9.1 DMA Base Address Table (C28)

Table 11-3. DMA Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
DmaRegs	DMA_REGS	DMA_BASE	0x0000_1000	YES	YES	-	-	-
Dmach1Regs	DMA_CH_REGS	DMA_CH1_BASE	0x0000_1020	YES	YES	-	-	-
Dmach2Regs	DMA_CH_REGS	DMA_CH2_BASE	0x0000_1040	YES	YES	-	-	-
Dmach3Regs	DMA_CH_REGS	DMA_CH3_BASE	0x0000_1060	YES	YES	-	-	-
Dmach4Regs	DMA_CH_REGS	DMA_CH4_BASE	0x0000_1080	YES	YES	-	-	-
Dmach5Regs	DMA_CH_REGS	DMA_CH5_BASE	0x0000_10A0	YES	YES	-	-	-
Dmach6Regs	DMA_CH_REGS	DMA_CH6_BASE	0x0000_10C0	YES	YES	-	-	-

11.9.2 DMA_REGS Registers

Table 11-4 lists the memory-mapped registers for the DMA_REGS registers. All register offset addresses not listed in Table 11-4 should be considered as reserved locations and the register contents should not be modified.

Table 11-4. DMA_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	DMACTRL	DMA Control Register	EALLOW	Go
1h	DEBUGCTRL	Debug Control Register	EALLOW	Go
4h	PRIORITYCTRL1	Priority Control 1 Register	EALLOW	Go
6h	PRIORITYSTAT	Priority Status Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 11-5 shows the codes that are used for access types in this section.

Table 11-5. DMA_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

11.9.2.1 DMACTRL Register (Offset = 0h) [Reset = 0h]

DMACTRL is shown in Figure 11-8 and described in Table 11-6.

Return to the Summary Table.

DMA Control Register

Figure 11-8. DMACTRL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						PRIORITYRESET	HARDRESET
R-0h						R-0/W1S-0h	R-0/W1S-0h

Table 11-6. DMACTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R	0h	Reserved
1	PRIORITYRESET	R-0/W1S	0h	The priority reset bit resets the round-robin state machine when a 1 is written. Service starts from the first enabled channel. Writes of 0 are ignored and this bit always reads back a 0. When a 1 is written to this bit, any pending burst transfer completes before resetting the channel priority machine. If CH1 is configured as a high-priority channel, and this bit is written to while CH1 is servicing a burst, both the CH1 burst and the next pending low-priority burst are completed before the state machine is reset. If CH1 is high-priority, the state machine restarts from CH2 (or the next highest enabled channel). Reset type: SYSRSn
0	HARDRESET	R-0/W1S	0h	Writing a 1 to the hard reset bit resets the whole DMA and aborts any current access (similar to applying a device reset). Writes of 0 are ignored and this bit always reads back a 0. For a soft reset, a bit is provided for each channel to perform a gentler reset. Refer to the channel control registers. When writing to this bit, there is a one cycle delay before it takes effect. Hence, a one-cycle delay (such as a NOP instruction) is required in software before attempting to access any other DMA register. Reset type: SYSRSn

11.9.2.2 DEBUGCTRL Register (Offset = 1h) [Reset = 0h]

DEBUGCTRL is shown in Figure 11-9 and described in Table 11-7.

Return to the Summary Table.

Debug Control Register

Figure 11-9. DEBUGCTRL Register

15	14	13	12	11	10	9	8
FREE	RESERVED						
R/W-0h	R-0h						
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 11-7. DEBUGCTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FREE	R/W	0h	Emulation Control This bit specifies the action when an emulation halt event occurs. Reset type: SYSRSn 0h (R/W) = The DMA completes the current read-write operation, then halts. 1h (R/W) = The DMA continues running during an emulation halt.
14-0	RESERVED	R	0h	Reserved

11.9.2.3 PRIORITYCTRL1 Register (Offset = 4h) [Reset = 0h]

PRIORITYCTRL1 is shown in Figure 11-10 and described in Table 11-8.

Return to the Summary Table.

Priority Control 1 Register

Figure 11-10. PRIORITYCTRL1 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CH1PRIORITY
R-0h							R/W-0h

Table 11-8. PRIORITYCTRL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	CH1PRIORITY	R/W	0h	DMA Channel 1 Priority This bit selects whether CH1 has high priority or not. The priority can only be changed when all channels are disabled. A priority reset should be performed before restarting channels after changing priority Reset type: SYSRSn 0h (R/W) = CH1 has the same priority as the other channels 1h (R/W) = CH1 has a higher priority than the other channels

11.9.2.4 PRIORITYSTAT Register (Offset = 6h) [Reset = 0h]

PRIORITYSTAT is shown in Figure 11-11 and described in Table 11-9.

Return to the Summary Table.

Priority Status Register

Figure 11-11. PRIORITYSTAT Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	ACTIVESTS_SHADOW			RESERVED	ACTIVESTS		
R-0h	R-0h			R-0h	R-0h		

Table 11-9. PRIORITYSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6-4	ACTIVESTS_SHADOW	R	0h	Active Channel Status Shadow These bits are only meaningful when CH1 is in high-priority mode. When CH1 is serviced, the ACTIVESTS bits are copied to the shadow bits and indicate which channel was interrupted by CH1. When CH1 service is completed, the shadow bits are copied back to the ACTIVESTS bits. If this bit field is zero or the same as the ACTIVESTS bit field, then no channel is pending due to a CH1 interrupt. When CH1 is not a higher priority channel, these bits should be ignored. Reset type: SYSRSn 0h (R/W) = No channel is active 1h (R/W) = CH 1 2h (R/W) = CH 2 3h (R/W) = CH 3 4h (R/W) = CH 4 5h (R/W) = CH 5 6h (R/W) = CH 6 7h (R/W) = Reserved
3	RESERVED	R	0h	Reserved
2-0	ACTIVESTS	R	0h	Active Channel Status These bits indicate which channel (if any) is currently active or performing a transfer. Reset type: SYSRSn 0h (R/W) = No channel is active 1h (R/W) = CH 1 2h (R/W) = CH 2 3h (R/W) = CH 3 4h (R/W) = CH 4 5h (R/W) = CH 5 6h (R/W) = CH 6 7h (R/W) = Reserved

11.9.3 DMA_CH_REGS Registers

Table 11-10 lists the memory-mapped registers for the DMA_CH_REGS registers. All register offset addresses not listed in Table 11-10 should be considered as reserved locations and the register contents should not be modified.

Table 11-10. DMA_CH_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	MODE	Mode Register	EALLOW	Go
1h	CONTROL	Control Register	EALLOW	Go
2h	BURST_SIZE	Burst Size Register	EALLOW	Go
3h	BURST_COUNT	Burst Count Register	EALLOW	Go
4h	SRC_BURST_STEP	Source Burst Step Register	EALLOW	Go
5h	DST_BURST_STEP	Destination Burst Step Register	EALLOW	Go
6h	TRANSFER_SIZE	Transfer Size Register	EALLOW	Go
7h	TRANSFER_COUNT	Transfer Count Register	EALLOW	Go
8h	SRC_TRANSFER_STEP	Source Transfer Step Register	EALLOW	Go
9h	DST_TRANSFER_STEP	Destination Transfer Step Register	EALLOW	Go
Ah	SRC_WRAP_SIZE	Source Wrap Size Register	EALLOW	Go
Bh	SRC_WRAP_COUNT	Source Wrap Count Register	EALLOW	Go
Ch	SRC_WRAP_STEP	Source Wrap Step Register	EALLOW	Go
Dh	DST_WRAP_SIZE	Destination Wrap Size Register	EALLOW	Go
Eh	DST_WRAP_COUNT	Destination Wrap Count Register	EALLOW	Go
Fh	DST_WRAP_STEP	Destination Wrap Step Register	EALLOW	Go
10h	SRC_BEG_ADDR_SHADOW	Source Begin Address Shadow Register	EALLOW	Go
12h	SRC_ADDR_SHADOW	Source Address Shadow Register	EALLOW	Go
14h	SRC_BEG_ADDR_ACTIVE	Source Begin Address Active Register	EALLOW	Go
16h	SRC_ADDR_ACTIVE	Source Address Active Register	EALLOW	Go
18h	DST_BEG_ADDR_SHADOW	Destination Begin Address Shadow Register	EALLOW	Go
1Ah	DST_ADDR_SHADOW	Destination Address Shadow Register	EALLOW	Go
1Ch	DST_BEG_ADDR_ACTIVE	Destination Begin Address Active Register	EALLOW	Go
1Eh	DST_ADDR_ACTIVE	Destination Address Active Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 11-11 shows the codes that are used for access types in this section.

Table 11-11. DMA_CH_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 11-11. DMA_CH_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

11.9.3.1 MODE Register (Offset = 0h) [Reset = 0h]

MODE is shown in Figure 11-12 and described in Table 11-12.

Return to the Summary Table.

Mode Register

Figure 11-12. MODE Register

15	14	13	12	11	10	9	8
CHINTE	DATASIZE	RESERVED	RESERVED	CONTINUOUS	ONESHOT	CHINTMODE	PERINTE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
OVRINTE	RESERVED		PERINTSEL				
R/W-0h	R-0h		R/W-0h				

Table 11-12. MODE Register Field Descriptions

Bit	Field	Type	Reset	Description
15	CHINTE	R/W	0h	Channel Interrupt Enable Bit This bit enables the DMA channel's CPU interrupt. Reset type: SYSRSn 0h (R/W) = Interrupt disabled 1h (R/W) = Interrupt enabled
14	DATASIZE	R/W	0h	Data Size Mode Bit This bit determines whether the DMA channel transfers 16 bits or 32 bits of data per read/write operation. Regardless of this setting, all data lengths and offsets in other DMA registers refer to 16- bit words. The pointer step increments must be configured to accomodate 32-bit words. Reset type: SYSRSn 0h (R/W) = 16-bit data transfer size 1h (R/W) = 32-bit data transfer size
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	CONTINUOUS	R/W	0h	Continuous Mode Bit If this bit is set to 1, then the channel re-initializes when TRANSFER_COUNT is zero and waits for the next event trigger. Otherwise, the DMA stops and clears the RUNSTS bit. Reset type: SYSRSn
10	ONESHOT	R/W	0h	One Shot Mode If this bit is set to 1, each peripheral event trigger causes the channel to perform an entire transfer. Otherwise, the channel only performs one burst per trigger. Reset type: SYSRSn
9	CHINTMODE	R/W	0h	Channel Interrupt Generation Mode This bit specifies when the DMA channel generates a CPU interrupt for a transfer. Reset type: SYSRSn 0h (R/W) = Generate interrupt at beginning of new transfer 1h (R/W) = Generate interrupt at end of transfer.
8	PERINTE	R/W	0h	Peripheral Event Trigger Enable This bit enables peripheral event triggers on the DMA channel. Reset type: SYSRSn 0h (R/W) = Peripheral event trigger disabled. Neither the selected peripheral nor software can start a DMA burst. 1h (R/W) = Peripheral event trigger enabled.

Table 11-12. MODE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	OVRINTE	R/W	0h	Overflow Interrupt Enable The bit determines whether the DMA module generates a CPU interrupt when it detects an overflow event. Reset type: SYSRSn 0h (R/W) = Overflow interrupt disabled 1h (R/W) = Overflow interrupt enabled
6-5	RESERVED	R	0h	Reserved
4-0	PERINTSEL	R/W	0h	Peripheral Event Trigger Source Select These are legacy bits and should be set to the channel number. The actual source selection is done via the DMACHSRCSELn registers, which are part of the DMA_CLA_SRC_SEL_REGS group. Reset type: SYSRSn

11.9.3.2 CONTROL Register (Offset = 1h) [Reset = 0h]

CONTROL is shown in Figure 11-13 and described in Table 11-13.

Return to the Summary Table.

Control Register

Figure 11-13. CONTROL Register

15	14	13	12	11	10	9	8
RESERVED	OVRFLG	RUNSTS	BURSTSTS	TRANSFERSTS	RESERVED	RESERVED	PERINTFLG
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
ERRCLR	RESERVED	RESERVED	PERINTCLR	PERINTFRC	SOFTRESET	HALT	RUN
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 11-13. CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	OVRFLG	R	0h	Overflow Flag This bit indicates that a peripheral event trigger was received while PERINTFLG was already set. It can be cleared by writing to the ERRCLR bit. Reset type: SYSRSn 0h (R/W) = No overflow detected 1h (R/W) = Overflow detected
13	RUNSTS	R	0h	Run Status Flag This bit indicates that the DMA channel is ready to respond to peripheral event triggers. This bit is set when a 1 is written to the RUN bit. It is cleared when a transfer completes (TRANSFER_COUNT = 0) and continuous mode is disabled, or when the HARDRESET, SOFTRESET, or HALT bit is set. Reset type: SYSRSn 0h (R/W) = The channel is disabled 1h (R/W) = The channel is enabled
12	BURSTSTS	R	0h	Burst Status Flag This bit is set when a DMA burst begins. The BURST_COUNT is set to the BURST_SIZE. This bit is cleared when BURST_COUNT reaches zero, or when the HARDRESET or SOFTRESET bit is set. Reset type: SYSRSn 0h (R/W) = No burst activity 1h (R/W) = The DMA is currently servicing or suspending a burst transfer from this channel
11	TRANSFERSTS	R	0h	Transfer Status Flag This bit is set when a DMA transfer begins. The address registers are copied to the shadow set and the TRANSFER_COUNT is set to the TRANSFER_SIZE. This bit is cleared when TRANSFER_COUNT reaches zero, or when the HARDRESET or SOFTRESET bit is set. Reset type: SYSRSn 0h (R/W) = No transfer activity 1h (R/W) = The channel is currently in the middle of a transfer regardless of whether a burst of data is actively being transferred or not
10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Reserved

Table 11-13. CONTROL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	PERINTFLG	R	0h	Peripheral Event Trigger Flag This bit indicates whether a peripheral event trigger has arrived. This bit is automatically cleared when the first burst transfer begins. Reset type: SYSRSn 0h (R/W) = Waiting for event trigger 1h (R/W) = Event trigger pending
7	ERRCLR	R-0/W1S	0h	Clear Error Writing a 1 to this bit will clear the OVRFLG bit. This is normally done when initializing the DMA module or if an overflow condition is detected. If an overflow event occurs at the same time this bit is set, the overrun has priority and the OVRFLG bit is set. Reset type: SYSRSn
6	RESERVED	R-0/W1S	0h	Reserved
5	RESERVED	R-0/W1S	0h	Reserved
4	PERINTCLR	R-0/W1S	0h	Clear Peripheral Event Trigger Writing a 1 to this bit clears PERINTFLG, which cancels a pending event trigger. This is normally done when initializing the DMA module. If an event trigger arrives at the same time this bit is set, the trigger has priority and PERINTFLG is set. Reset type: SYSRSn
3	PERINTFRC	R-0/W1S	0h	Force Peripheral Event Trigger If the PERINTE bit of the MODE register is set, writing a 1 to this bit sets PERINTFLG, which triggers a DMA burst. This bit can be used to start a DMA transfer in software. Reset type: SYSRSn
2	SOFTRESET	R-0/W1S	0h	Channel Soft Reset Writing a 1 to this bit places the channel into its default state after the current read/write access has completed: RUNSTS = 0 TRANSFERSTS = 0 BURSTSTS = 0 BURST_COUNT = 0 TRANSFER_COUNT = 0 SRC_WRAP_COUNT = 0 DST_WRAP_COUNT = 0 When writing to this bit, there is a one cycle delay before it takes effect. Hence, a one-cycle delay (such as a NOP instruction) is required in software before attempting to access any other DMA register. Reset type: SYSRSn
1	HALT	R-0/W1S	0h	Halt Channel Writing a 1 to this bit halts the DMA channel in its current state after any ongoing read/write access has completed. Reset type: SYSRSn
0	RUN	R-0/W1S	0h	Run Channel Writing a 1 to this bit enables the DMA channel and sets the RUNSTS bit to 1. This bit is also used to resume after a channel halt. The RUN bit is typically used to start the DMA channel after configuration. The channel will then wait for the first peripheral event trigger (PERINTFLG == 1) to start a burst. Reset type: SYSRSn

11.9.3.3 BURST_SIZE Register (Offset = 2h) [Reset = 0h]

BURST_SIZE is shown in Figure 11-14 and described in Table 11-14.

Return to the Summary Table.

Burst Size Register

Figure 11-14. BURST_SIZE Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			BURSTSIZE				
R-0h			R/W-0h				

Table 11-14. BURST_SIZE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4-0	BURSTSIZE	R/W	0h	These bits specify the burst size in 16-bit words. The actual size is equal to BURSTSIZE + 1. Reset type: SYSRSn

11.9.3.4 BURST_COUNT Register (Offset = 3h) [Reset = 0h]

BURST_COUNT is shown in Figure 11-15 and described in Table 11-15.

Return to the Summary Table.

Burst Count Register

Figure 11-15. BURST_COUNT Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			BURSTCOUNT				
R-0h			R-0h				

Table 11-15. BURST_COUNT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4-0	BURSTCOUNT	R	0h	These bits indicate the number of words left in the current burst. Reset type: SYSRSn 0h (R/W) = 0 word left in a burst 1h (R/W) = 1 word left in a burst 2h (R/W) = 2 word left in a burst 3h (R/W) = 3 word left in a burst 4h (R/W) = 4 word left in a burst 5h (R/W) = 5 word left in a burst 6h (R/W) = 6 word left in a burst 7h (R/W) = 7 word left in a burst 8h (R/W) = 8 word left in a burst 9h (R/W) = 9 word left in a burst Ah (R/W) = 10 word left in a burst Bh (R/W) = 11 word left in a burst Ch (R/W) = 12 word left in a burst Dh (R/W) = 13 word left in a burst Eh (R/W) = 14 word left in a burst Fh (R/W) = 15 word left in a burst 10h (R/W) = 16 word left in a burst 11h (R/W) = 17 word left in a burst 12h (R/W) = 18 word left in a burst 13h (R/W) = 19 word left in a burst 14h (R/W) = 20 word left in a burst 15h (R/W) = 21 word left in a burst 16h (R/W) = 22 word left in a burst 17h (R/W) = 23 word left in a burst 18h (R/W) = 24 word left in a burst 19h (R/W) = 25 word left in a burst 1Ah (R/W) = 26 word left in a burst 1Bh (R/W) = 27 word left in a burst 1Ch (R/W) = 28 word left in a burst 1Dh (R/W) = 29 word left in a burst 1Eh (R/W) = 30 word left in a burst 1Fh (R/W) = 31 word left in a burst

11.9.3.5 SRC_BURST_STEP Register (Offset = 4h) [Reset = 0h]

SRC_BURST_STEP is shown in Figure 11-16 and described in Table 11-16.

Return to the Summary Table.

Source Burst Step Register

Figure 11-16. SRC_BURST_STEP Register

15	14	13	12	11	10	9	8
SRCBURSTSTEP							
R/W-0h							
7	6	5	4	3	2	1	0
SRCBURSTSTEP							
R/W-0h							

Table 11-16. SRC_BURST_STEP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SRCBURSTSTEP	R/W	0h	These bits specify the change in the source address after each word in a burst. The size must be a 16-bit two's complement value between -4096 and 4095 (inclusive). This value is added to the source address after each read/write operation in a burst. Reset type: SYSRSn 0h (R/W) = No address change 1h (R/W) = Add 1 to the address 2h (R/W) = Add 2 to the address FFEh (R/W) = Add 4094 to the address FFFh (R/W) = Add 4095 to the address F000h (R/W) = Subtract 4096 from the address F001h (R/W) = Subtract 4095 from the address FFFEh (R/W) = Subtract 2 from the address FFFFh (R/W) = Subtract 1 from the address

11.9.3.6 DST_BURST_STEP Register (Offset = 5h) [Reset = 0h]

DST_BURST_STEP is shown in Figure 11-17 and described in Table 11-17.

Return to the Summary Table.

Destination Burst Step Register

Figure 11-17. DST_BURST_STEP Register

15	14	13	12	11	10	9	8
DSTBURSTSTEP							
R/W-0h							
7	6	5	4	3	2	1	0
DSTBURSTSTEP							
R/W-0h							

Table 11-17. DST_BURST_STEP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DSTBURSTSTEP	R/W	0h	These bits specify the change in the destination address after each word in a burst. The size must be a 16-bit two's complement value between -4096 and 4095 (inclusive). This value is added to the destination address after each read/write operation in a burst. Reset type: SYSRSn 0h (R/W) = No address change 1h (R/W) = Add 1 to the address 2h (R/W) = Add 2 to the address FFEh (R/W) = Add 4094 to the address FFFh (R/W) = Add 4095 to the address F000h (R/W) = Subtract 4096 from the address F001h (R/W) = Subtract 4095 from the address FFFEh (R/W) = Subtract 2 from the address FFFFh (R/W) = Subtract 1 from the address

11.9.3.7 TRANSFER_SIZE Register (Offset = 6h) [Reset = 0h]

TRANSFER_SIZE is shown in Figure 11-18 and described in Table 11-18.

Return to the Summary Table.

Transfer Size Register

Figure 11-18. TRANSFER_SIZE Register

15	14	13	12	11	10	9	8
TRANSFERSIZE							
R/W-0h							
7	6	5	4	3	2	1	0
TRANSFERSIZE							
R/W-0h							

Table 11-18. TRANSFER_SIZE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TRANSFERSIZE	R/W	0h	These bits specify the transfer size in bursts. The actual size is equal to TRANSFERSIZE + 1. Reset type: SYSRSn

11.9.3.8 TRANSFER_COUNT Register (Offset = 7h) [Reset = 0h]

TRANSFER_COUNT is shown in Figure 11-19 and described in Table 11-19.

Return to the Summary Table.

Transfer Count Register

Figure 11-19. TRANSFER_COUNT Register

15	14	13	12	11	10	9	8
TRANSFERCOUNT							
R-0h							
7	6	5	4	3	2	1	0
TRANSFERCOUNT							
R-0h							

Table 11-19. TRANSFER_COUNT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TRANSFERCOUNT	R	0h	These bits indicate the number of bursts left in the current transfer. Reset type: SYSRSn

11.9.3.9 SRC_TRANSFER_STEP Register (Offset = 8h) [Reset = 0h]

SRC_TRANSFER_STEP is shown in Figure 11-20 and described in Table 11-20.

Return to the Summary Table.

Source Transfer Step Register

Figure 11-20. SRC_TRANSFER_STEP Register

15	14	13	12	11	10	9	8
SRCTRANSFERSTEP							
R/W-0h							
7	6	5	4	3	2	1	0
SRCTRANSFERSTEP							
R/W-0h							

Table 11-20. SRC_TRANSFER_STEP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SRCTRANSFERSTEP	R/W	0h	These bits specify the change in the source address after a burst completes. The size must be a 16-bit two's complement value between -4096 and 4095 (inclusive). This value is added to the source address after each burst completes. Reset type: SYSRSn 0h (R/W) = No address change 1h (R/W) = Add 1 to the address 2h (R/W) = Add 2 to the address FFEh (R/W) = Add 4094 to the address FFFh (R/W) = Add 4095 to the address F000h (R/W) = Subtract 4096 from the address F001h (R/W) = Subtract 4095 from the address FFFEh (R/W) = Subtract 2 from the address FFFFh (R/W) = Subtract 1 from the address

11.9.3.10 DST_TRANSFER_STEP Register (Offset = 9h) [Reset = 0h]

DST_TRANSFER_STEP is shown in Figure 11-21 and described in Table 11-21.

Return to the Summary Table.

Destination Transfer Step Register

Figure 11-21. DST_TRANSFER_STEP Register

15	14	13	12	11	10	9	8
DSTTRANSFERSTEP							
R/W-0h							
7	6	5	4	3	2	1	0
DSTTRANSFERSTEP							
R/W-0h							

Table 11-21. DST_TRANSFER_STEP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DSTTRANSFERSTEP	R/W	0h	These bits specify the change in the destination address after a burst completes. The size must be a 16-bit two's complement value between -4096 and 4095 (inclusive). This value is added to the destination address after each burst completes. Reset type: SYSRSn 0h (R/W) = No address change 1h (R/W) = Add 1 to the address 2h (R/W) = Add 2 to the address FFEh (R/W) = Add 4094 to the address FFFh (R/W) = Add 4095 to the address F000h (R/W) = Subtract 4096 from the address F001h (R/W) = Subtract 4095 from the address FFFEh (R/W) = Subtract 2 from the address FFFFh (R/W) = Subtract 1 from the address

11.9.3.11 SRC_WRAP_SIZE Register (Offset = Ah) [Reset = FFFFh]

SRC_WRAP_SIZE is shown in Figure 11-22 and described in Table 11-22.

Return to the Summary Table.

Source Wrap Size Register

Figure 11-22. SRC_WRAP_SIZE Register

15	14	13	12	11	10	9	8
WRAPSIZE							
R/W-FFFFh							
7	6	5	4	3	2	1	0
WRAPSIZE							
R/W-FFFFh							

Table 11-22. SRC_WRAP_SIZE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	WRAPSIZE	R/W	FFFFh	These bits specify the number of bursts to transfer before the source address wraps around to the beginning address. The actual number is equal to WRAPSIZE + 1. To disable the wrapping function, set WRAPSIZE to a value larger than TRANSFERSIZE. Reset type: SYSRSn

11.9.3.12 SRC_WRAP_COUNT Register (Offset = Bh) [Reset = 0h]

SRC_WRAP_COUNT is shown in Figure 11-23 and described in Table 11-23.

Return to the Summary Table.

Source Wrap Count Register

Figure 11-23. SRC_WRAP_COUNT Register

15	14	13	12	11	10	9	8
WRAPSIZE							
R-0h							
7	6	5	4	3	2	1	0
WRAPSIZE							
R-0h							

Table 11-23. SRC_WRAP_COUNT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	WRAPSIZE	R	0h	These bits indicate the number of bursts left before wrapping the source address. Reset type: SYSRSn

11.9.3.13 SRC_WRAP_STEP Register (Offset = Ch) [Reset = 0h]

SRC_WRAP_STEP is shown in Figure 11-24 and described in Table 11-24.

Return to the Summary Table.

Source Wrap Step Register

Figure 11-24. SRC_WRAP_STEP Register

15	14	13	12	11	10	9	8
WRAPSTEP							
R/W-0h							
7	6	5	4	3	2	1	0
WRAPSTEP							
R/W-0h							

Table 11-24. SRC_WRAP_STEP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	WRAPSTEP	R/W	0h	These bits specify the change in the source beginning address when the wrap counter reaches zero. The size must be a 16-bit two's complement value between -4096 and 4095 (inclusive). This value is added to the source address when wrapping occurs. Reset type: SYSRSn 0h (R/W) = No address change 1h (R/W) = Add 1 to the address 2h (R/W) = Add 2 to the address FFEh (R/W) = Add 4094 to the address FFFh (R/W) = Add 4095 to the address F000h (R/W) = Subtract 4096 from the address F001h (R/W) = Subtract 4095 from the address FFFEh (R/W) = Subtract 2 from the address FFFFh (R/W) = Subtract 1 from the address

11.9.3.14 DST_WRAP_SIZE Register (Offset = Dh) [Reset = FFFFh]

DST_WRAP_SIZE is shown in Figure 11-25 and described in Table 11-25.

Return to the Summary Table.

Destination Wrap Size Register

Figure 11-25. DST_WRAP_SIZE Register

15	14	13	12	11	10	9	8
WRAPSIZE							
R/W-FFFFh							
7	6	5	4	3	2	1	0
WRAPSIZE							
R/W-FFFFh							

Table 11-25. DST_WRAP_SIZE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	WRAPSIZE	R/W	FFFFh	These bits specify the number of bursts to transfer before the destination address wraps around to the beginning address. The actual number is equal to WRAPSIZE + 1. To disable the wrapping function, set WRAPSIZE to a value larger than TRANSFERSIZE. Reset type: SYSRSn

11.9.3.15 DST_WRAP_COUNT Register (Offset = Eh) [Reset = 0h]

DST_WRAP_COUNT is shown in Figure 11-26 and described in Table 11-26.

Return to the Summary Table.

Destination Wrap Count Register

Figure 11-26. DST_WRAP_COUNT Register

15	14	13	12	11	10	9	8
WRAPSIZE							
R-0h							
7	6	5	4	3	2	1	0
WRAPSIZE							
R-0h							

Table 11-26. DST_WRAP_COUNT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	WRAPSIZE	R	0h	These bits indicate the number of bursts left before wrapping the destination address. Reset type: SYSRSn

11.9.3.16 DST_WRAP_STEP Register (Offset = Fh) [Reset = 0h]

DST_WRAP_STEP is shown in Figure 11-27 and described in Table 11-27.

Return to the Summary Table.

Destination Wrap Step Register

Figure 11-27. DST_WRAP_STEP Register

15	14	13	12	11	10	9	8
WRAPSTEP							
R/W-0h							
7	6	5	4	3	2	1	0
WRAPSTEP							
R/W-0h							

Table 11-27. DST_WRAP_STEP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	WRAPSTEP	R/W	0h	These bits specify the change in the destination beginning address when the wrap counter reaches zero. The size must be a 16-bit two's complement value between -4096 and 4095 (inclusive). This value is added to the destination address when wrapping occurs. Reset type: SYSRSn 0h (R/W) = No address change 1h (R/W) = Add 1 to the address 2h (R/W) = Add 2 to the address FFEh (R/W) = Add 4094 to the address FFFh (R/W) = Add 4095 to the address F000h (R/W) = Subtract 4096 from the address F001h (R/W) = Subtract 4095 from the address FFFEh (R/W) = Subtract 2 from the address FFFFh (R/W) = Subtract 1 from the address

11.9.3.17 SRC_BEG_ADDR_SHADOW Register (Offset = 10h) [Reset = 0h]

SRC_BEG_ADDR_SHADOW is shown in Figure 11-28 and described in Table 11-28.

Return to the Summary Table.

Source Begin Address Shadow Register

Figure 11-28. SRC_BEG_ADDR_SHADOW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BEGADDR																															
R/W-0h																															

Table 11-28. SRC_BEG_ADDR_SHADOW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BEGADDR	R/W	0h	Shadow Source Beginning Address At the start of a transfer, the value in this register is loaded into the SRC_BEG_ADDR_ACTIVE register and used as the beginning value for the source address. This register can be safely updated during a transfer. Reset type: SYSRSn

11.9.3.18 SRC_ADDR_SHADOW Register (Offset = 12h) [Reset = 0h]

SRC_ADDR_SHADOW is shown in Figure 11-29 and described in Table 11-29.

Return to the Summary Table.

Source Address Shadow Register

Figure 11-29. SRC_ADDR_SHADOW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R/W-0h																															

Table 11-29. SRC_ADDR_SHADOW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R/W	0h	Shadow Source Address At the start of a transfer, the value in this register is loaded into the SRC_ADDR_ACTIVE register and used as the value of the source address. This register can be safely updated during a transfer. Reset type: SYSRSn

11.9.3.19 SRC_BEG_ADDR_ACTIVE Register (Offset = 14h) [Reset = 0h]

SRC_BEG_ADDR_ACTIVE is shown in Figure 11-30 and described in Table 11-30.

Return to the Summary Table.

Source Begin Address Active Register

Figure 11-30. SRC_BEG_ADDR_ACTIVE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BEGADDR																															
R-0h																															

Table 11-30. SRC_BEG_ADDR_ACTIVE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BEGADDR	R	0h	Active Source Beginning Address If a transfer is ongoing, this register holds the current beginning value for the source address. This address may be updated after wrapping. When a transfer starts, this register is loaded with the shadow address from the SRC_BEG_ADDR_SHADOW register. Reset type: SYSRSn

11.9.3.20 SRC_ADDR_ACTIVE Register (Offset = 16h) [Reset = 0h]

SRC_ADDR_ACTIVE is shown in Figure 11-31 and described in Table 11-31.

Return to the Summary Table.

Source Address Active Register

Figure 11-31. SRC_ADDR_ACTIVE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R-0h																															

Table 11-31. SRC_ADDR_ACTIVE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R	0h	Active Source Address If a transfer is ongoing, this register holds the current value of the source address. This address may change after a write, a burst, or wrapping. Reset type: SYSRSn

11.9.3.21 DST_BEG_ADDR_SHADOW Register (Offset = 18h) [Reset = 0h]

DST_BEG_ADDR_SHADOW is shown in Figure 11-32 and described in Table 11-32.

Return to the Summary Table.

Destination Begin Address Shadow Register

Figure 11-32. DST_BEG_ADDR_SHADOW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BEGADDR																															
R/W-0h																															

Table 11-32. DST_BEG_ADDR_SHADOW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BEGADDR	R/W	0h	Shadow Destination Beginning Address At the start of a transfer, the value in this register is loaded into the DST_BEG_ADDR_ACTIVE register and used as the beginning value for the destination address. This register can be safely updated during a transfer. Reset type: SYSRSn

11.9.3.22 DST_ADDR_SHADOW Register (Offset = 1Ah) [Reset = 0h]

DST_ADDR_SHADOW is shown in Figure 11-33 and described in Table 11-33.

Return to the Summary Table.

Destination Address Shadow Register

Figure 11-33. DST_ADDR_SHADOW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R/W-0h																															

Table 11-33. DST_ADDR_SHADOW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R/W	0h	Shadow Destination Address At the start of a transfer, the value in this register is loaded into the DST_ADDR_ACTIVE register and used as the value of the destination address. This register can be safely updated during a transfer. Reset type: SYSRSn

11.9.3.23 DST_BEG_ADDR_ACTIVE Register (Offset = 1Ch) [Reset = 0h]

DST_BEG_ADDR_ACTIVE is shown in Figure 11-34 and described in Table 11-34.

Return to the Summary Table.

Destination Begin Address Active Register

Figure 11-34. DST_BEG_ADDR_ACTIVE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BEGADDR																															
R-0h																															

Table 11-34. DST_BEG_ADDR_ACTIVE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BEGADDR	R	0h	Active Destination Beginning Address If a transfer is ongoing, this register holds the current destination value for the source address. This address may be updated after wrapping. When a transfer starts, this register is loaded with the shadow address from the DST_BEG_ADDR_SHADOW register. Reset type: SYSRSn

11.9.3.24 DST_ADDR_ACTIVE Register (Offset = 1Eh) [Reset = 0h]

DST_ADDR_ACTIVE is shown in Figure 11-35 and described in Table 11-35.

Return to the Summary Table.

Destination Address Active Register

Figure 11-35. DST_ADDR_ACTIVE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R-0h																															

Table 11-35. DST_ADDR_ACTIVE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R	0h	Active Destination Address If a transfer is ongoing, this register holds the current value of the destination address. This address may change after a write, a burst, or wrapping. Reset type: SYSRSn

11.9.4 DMA Registers to Driverlib Functions

Table 11-36. DMA Registers to Driverlib Functions

File	Driverlib Function
CTRL	
dma.h	DMA_initController
DEBUGCTRL	
dma.h	DMA_setEmulationMode
PRIORITYCTRL1	
dma.h	DMA_setPriorityMode
PRIORITYSTAT	
-	
MODE	
dma.c	DMA_configMode
dma.h	DMA_enableTrigger
dma.h	DMA_disableTrigger
dma.h	DMA_enableInterrupt
dma.h	DMA_disableInterrupt
dma.h	DMA_enableOverrunInterrupt
dma.h	DMA_disableOverrunInterrupt
dma.h	DMA_setInterruptMode
CONTROL	
dma.h	DMA_triggerSoftReset
dma.h	DMA_forceTrigger
dma.h	DMA_clearTriggerFlag
dma.h	DMA_getTransferStatusFlag
dma.h	DMA_getBurstStatusFlag
dma.h	DMA_getRunStatusFlag
dma.h	DMA_getOverflowFlag
dma.h	DMA_getTriggerFlagStatus
dma.h	DMA_startChannel

Table 11-36. DMA Registers to Driverlib Functions (continued)

File	Driverlib Function
dma.h	DMA_stopChannel
dma.h	DMA_clearErrorFlag
BURST_SIZE	
dma.c	DMA_configBurst
BURST_COUNT	
-	
SRC_BURST_STEP	
dma.c	DMA_configBurst
DST_BURST_STEP	
dma.c	DMA_configBurst
TRANSFER_SIZE	
dma.c	DMA_configTransfer
TRANSFER_COUNT	
-	
SRC_TRANSFER_STEP	
dma.c	DMA_configTransfer
DST_TRANSFER_STEP	
dma.c	DMA_configTransfer
SRC_WRAP_SIZE	
dma.c	DMA_configWrap
SRC_WRAP_COUNT	
-	
SRC_WRAP_STEP	
dma.c	DMA_configWrap
DST_WRAP_SIZE	
dma.c	DMA_configWrap
DST_WRAP_COUNT	
-	
DST_WRAP_STEP	
dma.c	DMA_configWrap
SRC_BEG_ADDR_SHADOW	
dma.c	DMA_configAddresses
dma.h	DMA_configSourceAddress
SRC_ADDR_SHADOW	
dma.c	DMA_configAddresses
dma.h	DMA_configSourceAddress
SRC_BEG_ADDR_ACTIVE	
-	
SRC_ADDR_ACTIVE	
-	
DST_BEG_ADDR_SHADOW	
dma.c	DMA_configAddresses
dma.h	DMA_configDestAddress
DST_ADDR_SHADOW	
dma.c	DMA_configAddresses

Table 11-36. DMA Registers to Driverlib Functions (continued)

File	Driverlib Function
dma.h	DMA_configDestAddress
DST_BEG_ADDR_ACTIVE	
-	
DST_ADDR_ACTIVE	
-	

This page intentionally left blank.

Chapter 12

External Memory Interface (EMIF)

This chapter describes the external memory interface (EMIF).

Further information can be found in the following documents:

Accessing External SDRAM on the TMS320F2837x/2807x Microcontrollers Using C/C++ Application Report

Design and Usage Guidelines for the C2000™ External Memory Interface (EMIF) Application Report

12.1 Introduction

This device supports dual-core architecture. To have a dedicated EMIF for each CPU subsystem, the device supports two EMIF modules — EMIF1 and EMIF2, as shown in Figure 12-1. Both modules are exactly the same with the same feature set, but have different address/data sizes. EMIF1 is shared between the CPU1 and CPU2 subsystem, whereas EMIF2 is dedicated to the CPU1 subsystem.

Table 12-1 gives the configuration for two EMIF modules.

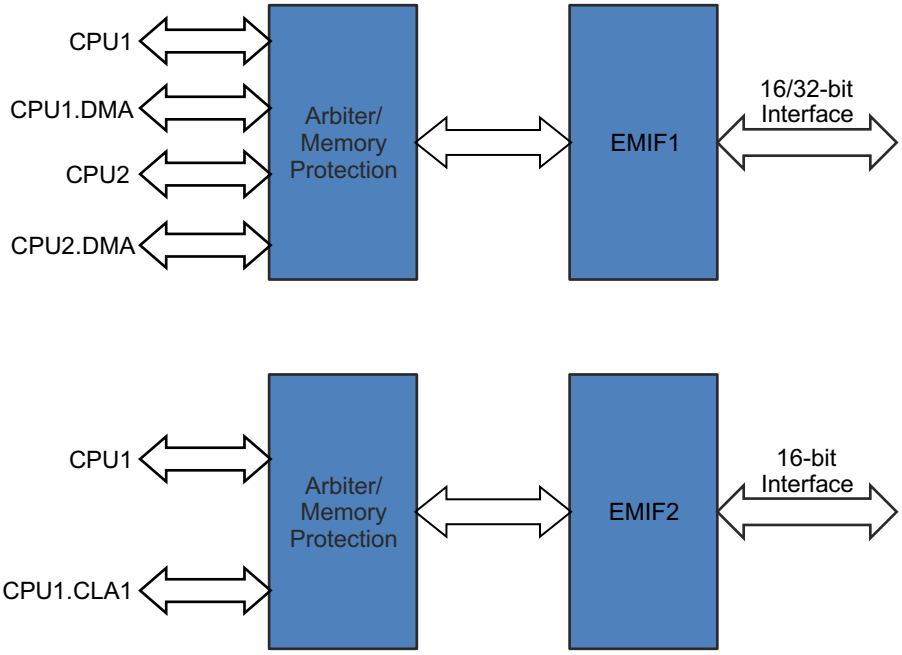


Figure 12-1. EMIF Module Overview

Table 12-1. Configuration for EMIF1 and EMIF2 Modules

	EMIF1	EMIF2
176-Pin Package	Yes	NA
337- Pin Package	Yes	Yes
Maximum Data Width	32	16
Maximum Address Width	22 (Some of the EMIF1 pins are muxed with each other. Refer to Section 12.2.11 for usage)	13
SDRAM CSx Support	1 (CS0)	1 (CS0)
ASRAM CSx Support	3 (CS2/CS3/CS4)	1 (CS2)

Note

Subsequent sections in this chapter provide the details on the generic EMIF module. Unless otherwise specified, pin names are used from EMIF1 to define the functionality.

12.1.1 EMIF Related Collateral

Foundational Materials

- C2000 Academy - EMIF

Getting Started Materials

- *Design and Usage Guidelines for the C2000 External Memory Interface (EMIF) Application Report*

Expert Materials

- *Accessing External SDRAM on the TMS320F2837x/2807x Microcontrollers Using C/C++ Application Report*
  - In addition to the devices in the title, this material also applies to other devices with EMIF.

12.1.2 Purpose of the Peripheral

This EMIF memory controller is compliant with the JESD21-C SDR SDRAM memories utilizing a 32-bit/16-bit data bus. The purpose of this EMIF is to provide a means for the CPU to connect to a variety of external devices including:

- Single data rate (SDR) SDRAM
- Asynchronous devices including NOR Flash and SRAM

A common use for the EMIF is to interface with both a Flash device and an SDRAM device simultaneously. Section 12.3 contains an example of operating the EMIF in this configuration.

12.1.3 Features

The EMIF controller includes many features to enhance the ease and flexibility of connecting to the external SDR SDRAM and asynchronous devices.

12.1.3.1 Asynchronous Memory Support

The EMIF controller supports asynchronous:

- SRAM memories
- NOR Flash memories

There is an external wait input that allows slower asynchronous memories to extend the memory access. The EMIF module supports more than one chip select (enable). Each chip select has the following individually programmable attributes:

- Data bus width
- Read cycle timings: setup, hold, strobe
- Write cycle timings: setup, hold, strobe
- Bus turnaround time
- Extended wait option with programmable timeout
- Select strobe option

12.1.3.2 Synchronous DRAM Memory Support

The EMIF module supports 16-bit/32-bit SDRAM in addition to the asynchronous memories listed in Section 12.1.3.1. The EMIF module has a single SDRAM chip select. SDRAM configurations that are supported are:

- One, two and four bank SDRAM devices
- Devices with eight, nine, ten, and eleven column address
- CAS latency of two or three clock cycles
- 16-bit/32-bit data bus width
- 3.3V LVCMOS interface

Additionally, the EMIF supports placing the SDRAM in self-refresh and power-down modes. The self-refresh mode allows the SDRAM to be put in a low-power state while still retaining memory contents, since the SDRAM continues to refresh itself even without clocks from the microcontroller. The power-down mode achieves even lower power, except the microcontroller must periodically wake up and issue refreshes if data retention is required.

Note that the EMIF module does not support mobile SDRAM devices.

12.1.4 Functional Block Diagram

Figure 12-2 illustrates the connections between the EMIF and its internal requesters, along with the external EMIF pins. Section 12.2.2 contains a description of the entities internal to the MCU that can send requests to the EMIF, along with their prioritization. Section 12.2.3 describes the EMIF external pins and summarizes their purpose when interfacing with the SDRAM and asynchronous devices.

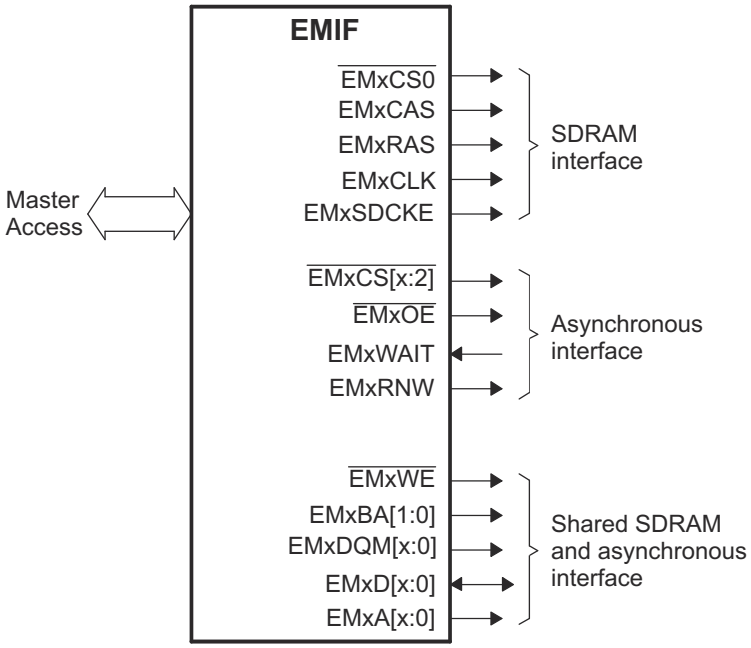


Figure 12-2. EMIF Functional Block Diagram

12.1.5 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

Some IO functionality is defined by GPIO register settings independent of this peripheral. For input signals, the GPIO input qualification must be set to asynchronous mode by setting the appropriate GPxQSELn register bits to 11b. The internal pullups can be configured in the GPyPUD register.

See the *General-Purpose Input/Output (GPIO)* chapter for more details on GPIO mux and settings.

12.2 EMIF Module Architecture

This section provides details about the architecture and operation of the EMIF. Both the SDRAM and asynchronous interface are covered, along with other system-related configurations such as clock control.

12.2.1 EMIF Clock Control

The EMIF clock is output on the EMxCLK pin and must be used when interfacing to external SDRAM devices. The EMIF module gets the PLLSYSCLK clock domain as the input. The user can choose to run the EMIF at PLLSYSCLK/1 or PLLSYSCLK/2 clock frequency by configuring the EMIFxCLKDIV field in the PERCLKDIVSEL register in the *Clock Control* module.

12.2.2 EMIF Requests

Different sources within the MCU can make requests to the EMIF. These requests consist of accesses to the SDRAM memory, the asynchronous memory, and the EMIF registers. The EMIF can process only one request at a time. Therefore, a high performance master arbitration block exists within the MCU to provide prioritized requests from the different sources to the EMIF. The sources are:

- CPU1
- CPU1.DMA
- CPU2
- CPU2.DMA

If a request is submitted from two or more sources simultaneously, the crossbar switch will forward the highest priority request to the EMIF first. Upon completion of a request, the master arbitration block again evaluates the pending requests and forwards the highest priority pending request to the EMIF.

The master arbitration block always allows RD access from any of the masters. But for WR access (or execute access), the master arbitration block only allows access of masters from a CPU subsystem that takes master ownership of the EMIF module based on the configuration in the EMIF1MSEL register in the *Memory Controller* module. Note that only the EMIF1 has access from both the CPU subsystems; hence, the concept of grab-semaphore is applicable for the EMIF1 only.

When the EMIF receives a request, it may or may not be immediately processed. In some cases, the EMIF will perform one or more auto refresh cycles before processing the request. For details on the EMIF internal arbitration between performing requests and performing auto refresh cycles, see Section 12.2.13.

12.2.3 EMIF Signal Descriptions

This section describes the function of each of the EMIF signals.

Table 12-2. EMIF Pins Used to Access Both SDRAM and Asynchronous Memories

Pins	I/O	Description
EM1D[x:0]	I/O	**EMIF data bus.**
EM1A[x:0]	O	**EMIF address bus.** When interfacing to an SDRAM device, these pins are primarily used to provide the row and column address to the SDRAM. The mapping from the internal program address to the external values placed on these pins is found in Table 12-14. EM1A[10] is also used during the PRE command to select which banks to deactivate. When interfacing to an asynchronous device, these pins are used in conjunction with the EM1BA pins to form the address that is sent to the device. The mapping from the internal program address to the external values placed on these pins is found in Section 12.2.6.1.
EM1BA[1:0]	O	**EMIF bank address.** When interfacing to an SDRAM device, these pins are used to provide the bank address inputs to the SDRAM. The mapping from the internal program address to the external values placed on these pins is found in Table 12-14. When interfacing to an asynchronous device, these pins are used in conjunction with the EM1A pins to form the address that is sent to the device. The mapping from the internal program address to the external values placed on these pins is found in Section 12.2.6.1.
EM1DQM[x:0]	O	**Active-low byte enables.** When interfacing to SDRAM, these pins are connected to the DQM pins of the SDRAM to individually enable/disable each of the bytes in a data access. When interfacing to an asynchronous device, these pins are connected to byte enables. See Section 12.2.6 for details.
$\overline{\text{EM1WE}}$	O	**Active-low write enable.** When interfacing to SDRAM, this pin is connected to the nWE pin of the SDRAM and is used to send commands to the device. When interfacing to an asynchronous device, this pin provides a signal which is active-low during the strobe period of an asynchronous write access cycle.

Table 12-3. EMIF Pins Specific to SDRAM

Pins	I/O	Description
$\overline{\text{EM1CS0}}$	O	**Active-low chip enable pin for SDRAM devices.** This pin is connected to the chip-select pin of the attached SDRAM device and is used for enabling/disabling commands. By default, EMIF keeps this SDRAM chip select active, even if EMIF is not interfaced with an SDRAM device. This pin is deactivated when accessing the asynchronous memory bank and is reactivated on completion of the asynchronous access.
EM1RAS	O	**Active-low row address strobe pin.** This pin is connected to the nRAS pin of the attached SDRAM device and is used for sending commands to the device.
EM1CAS	O	**Active-low column address strobe pin.** This pin is connected to the nCAS pin of the attached SDRAM device and is used for sending commands to the device.
EM1SDCKE	O	**Clock enable pin.** This pin is connected to the CKE pin of the attached SDRAM device and is used for issuing the SELF REFRESH command which places the device in self-refresh mode. See Section 12.2.5.7 for details.
EM1CLK	O	**SDRAM clock pin.** This pin is connected to the CLK pin of the attached SDRAM device. See Section 12.2.1 for details on the clock signal.

Table 12-4. EMIF Pins Specific to Asynchronous Memory

Pins	I/O	Description
$\overline{\text{EM1CS}}$[4:2]	O	**Active-low chip enable pins for asynchronous devices.** These pins are meant to be connected to the chip-select pins of the attached asynchronous device. These pins are active only during accesses to the asynchronous memory.
EM1WAIT	I	**Wait input with programmable polarity / NAND Flash ready input.** A connected asynchronous device can extend the strobe period of an access cycle by asserting the EM1WAIT input to EMIF as described in Section 12.2.6.6. To enable this functionality, the EW bit in the asynchronous 1 configuration register (ASYNC_CS2_CFG) must be set to 1. In addition, the WP0 bit in ASYNC_CS2_CFG must be configured to define the polarity of the EM1WAIT pin. When the CS2NAND/CS3NAND/CS4NAND/CS5NAND bit in the NAND Flash control register (NANDFCR) is set, this pin instead functions as a NAND Flash ready input.
$\overline{\text{EM1OE}}$	O	**Active-low pin enable for asynchronous devices.** This pin provides a signal which is active-low during the strobe period of an asynchronous read access cycle.
EM1RNW	O	**EMIF asynchronous read/write control.** This pin stays high during reads and stays low during writes (same duration as CS).

12.2.4 EMIF Signal Multiplexing Control

Several EMIF signals are multiplexed with other functions on this microcontroller. Refer to the multiplexing section of the *General-Purpose Input/Output (GPIO)* chapter for more information on how to enable the output of these EMIF signals.

12.2.5 SDRAM Controller and Interface

The EMIF controller provides a glueless interface to most standard SDR SDRAM devices and supports features like self-refresh mode and prioritized refresh. In addition, it provides flexibility through programmable parameters such as the refresh rate, CAS latency, and many SDRAM timing parameters. The following sections include details on how to Interface and properly configure the EMIF to perform read and write operations to externally connected SDR SDRAM devices. Also, Section 12.3 provides a detailed example of interfacing the EMIF to a common SDRAM device.

12.2.5.1 SDRAM Commands

The EMIF controller supports the SDRAM commands described in Table 12-5. Table 12-6 shows the truth table for the SDRAM commands, and an example timing waveform of the PRE command is shown in Figure 12-3. EM1A[10] is pulled low in this example to deactivate only the bank specified by the EM1BA pins.

Table 12-5. EMIF SDRAM Commands

Command	Function
PRE	**Precharge.** Depending on the value of EM1A[10], the PRE command either deactivates the open row in all banks (EM1A[10] = 1) or only the bank specified by the EM1BA[1:0] pins (EM1A[10] = 0).
ACTV	**Activate**. The ACTV command activates the selected row in a particular bank for the current access.
READ	**Read.** The READ command outputs the starting column address and signals the SDRAM to begin the burst read operation. Address EM1A[10] is always pulled low to avoid auto precharge. This allows for better bank interleaving performance.
WRT	**Write.** The WRT command outputs the starting column address and signals the SDRAM to begin the burst write operation. Address EM1A[10] is always pulled low to avoid auto precharge. This allows for better bank interleaving performance.
BT	**Burst terminate**. The BT command is used to truncate the current read or write burst request. On this device, all the SDRAM accesses are single access except when EMIF controller splits a single access into multiple access (for example, a 32-bit access from CPU is split into two 16-bit accesses if external SDRAM device is 16 bit (NM =1).
LMR	**Load mode register**. The LMR command sets the mode register of the attached SDRAM devices and is only issued during the SDRAM initialization sequence described in Section 12.2.5.4.
REFR	**Auto refresh**. The REFR command signals the SDRAM to perform an auto refresh according to the internal address.
SLFR	**Self refresh**. The self-refresh command places the SDRAM into self-refresh mode, during which the SDRAM provides its own clock signal and auto refresh cycles.
NOP	**No operation**. The NOP command is issued during all cycles in which one of the above commands is not issued.

Table 12-6. Truth Table for SDRAM Commands

SDRAM Pins:	CKE	nCS	nRAS	nCAS	nWE	BA[1:0]	A[12:11]	A[10]	A[9:0]
EMIF Pins:	EM1SDCKE	$\overline{\text{EM1CS}}$[0]	EM1RAS	EM1CAS	$\overline{\text{EM1WE}}$	EM1BA[1:0]	EM1A[12:11]	EM1A[10]	EM1A[9:0]
PRE	H	L	L	H	L	Bank/X	X	L/H	X
ACTV	H	L	L	H	H	Bank	Row	Row	Row
READ	H	L	H	L	H	Bank	Column	L	Column
WRT	H	L	H	L	L	Bank	Column	L	Column
BT	H	L	H	H	L	X	X	X	X
LMR	H	L	L	L	L	X	Mode	Mode	Mode
REFR	H	L	L	L	H	X	X	X	X
SLFR	L	L	L	L	H	X	X	X	X
NOP	H	L	H	H	H	X	X	X	X

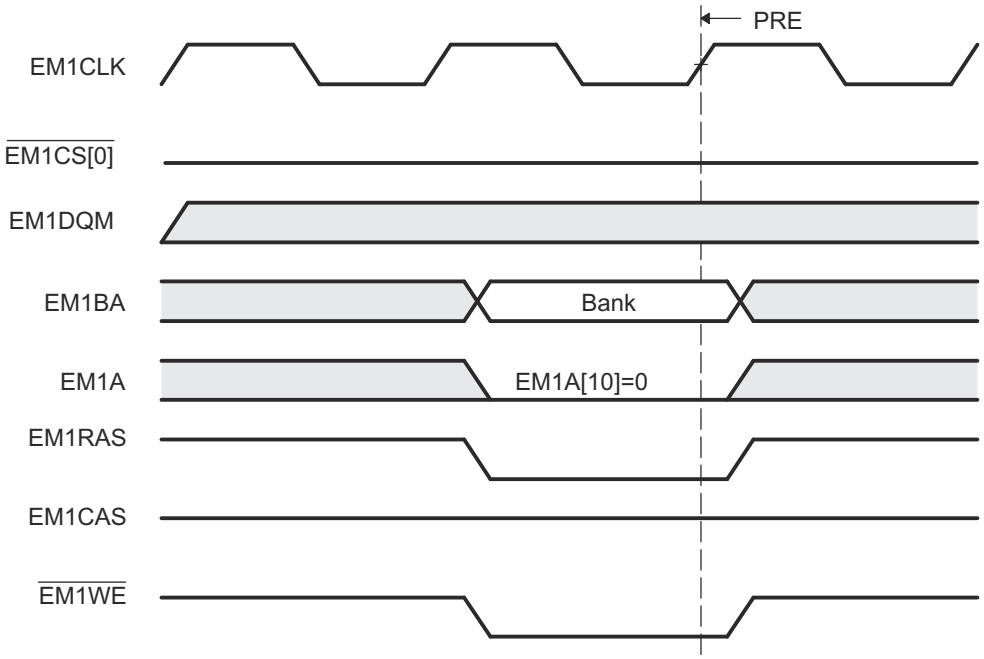


Figure 12-3. Timing Waveform of SDRAM PRE Command

12.2.5.2 Interfacing to SDRAM

The EMIF supports a glueless interface to SDRAM devices with the following characteristics:

- Pre-charge bit is A[10]
- The number of column address bits is 8, 9, 10, or 11.
- The number of row address bits is 13, 14, 15, or 16.
- The number of internal banks is 1, 2, or 4.

Figure 12-4 shows an interface between the EMIF and a 2M × 16 × 4 bank SDRAM device, and Figure 12-5 shows an interface between the EMIF and a 512K × 16 × 2 bank SDRAM device. For devices supporting 16-bit interface, refer to Table 12-7 for list of commonly-supported SDRAM devices and the required connections for the address pins.

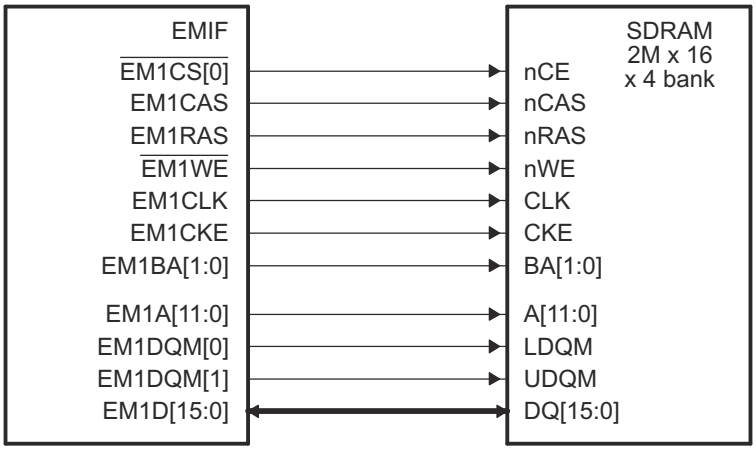


Figure 12-4. EMIF to 2M × 16 × 4 Bank SDRAM Interface

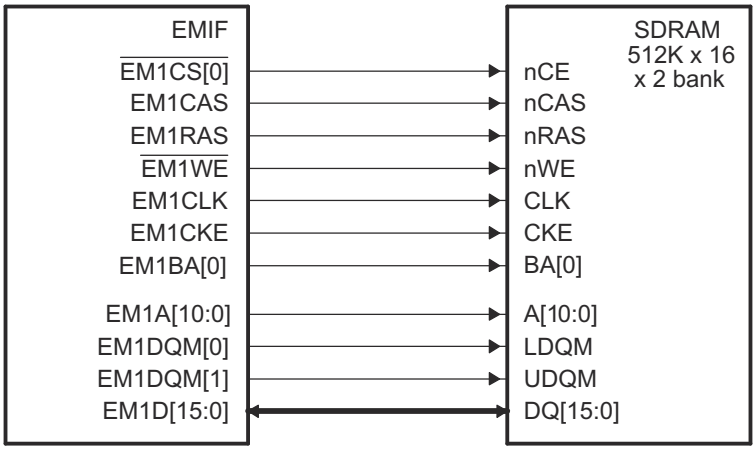


Figure 12-5. EMIF to 512K × 16 × 2 Bank SDRAM Interface

Table 12-7. 16-bit EMIF Address Pin Connections

SDRAM Size	Width	Banks	Device	Address Pins
16M bits	×16	2	SDRAM	A[10:0]
			EMIF	EM1A[10:0]
64M bits	×16	4	SDRAM	A[11:0]
			EMIF	EM1A[11:0]
128M bits	×16	4	SDRAM	A[11:0]
			EMIF	EM1A[11:0]
256M bits	x16	4	SDRAM	A[12:0]
			EMIF	EM1A[12:0]
512M bits	x16	4	SDRAM	A[12:0]
			EMIF	EM1A[12:0]

12.2.5.3 SDRAM Configuration Registers

The operation of the EMIF SDRAM interface is controlled by programming the appropriate configuration registers. This section describes the purpose and function of each configuration register, but Section 12.5 can be referred to for a more detailed description of each register, including the default registers values and bit-field positions. The following tables list the four such configuration registers, along with a description of each of their programmable fields.

Note

Writing to any of the fields: NM, CL, IBANK, and PAGESIZE in the SDRAM configuration register (SDRAM_CR) causes the EMIF to abandon whatever the EMIF is currently doing and trigger the SDRAM initialization procedure described in Section 12.2.5.4.

Table 12-8. Description of the SDRAM Configuration Register (SDRAM_CR)

Parameter	Description
SR	This bit controls entering and exiting of the self-refresh mode
PD	This bit controls entering and exiting of the power-down mode. If both SR and PD bits are set, the EMIF goes into self-refresh mode.
PDWR	Perform refreshes during power down. Writing a 1 to this bit causes the EMIF to exit the power-down state and issue an AUTO REFRESH command every time Refresh May level is set. This bit must be set along with PD when entering power-down mode.
NM	**Narrow Mode.** This bit defines the width of the data bus between the EMIF and the attached SDRAM device. When set to 1, the data bus is set to 16-bits. When set to 0, the data bus is set to 32-bits.
CL	**CAS latency.** This field defines the number of clock cycles between when an SDRAM issues a READ command and when the first piece of data appears on the bus. The value in this field is sent to the attached SDRAM device using the LOAD MODE REGISTER command during the SDRAM initialization procedure as described in Section 12.2.5.4. Only, values of 2h (CAS latency = 2) and 3h (CAS latency = 3) are supported and must be written to this field. While updating the CL field, BIT11_9LOCK bit field must be set to 1 simultaneously.
IBANK	**Number of Internal SDRAM Banks.** This field defines the number of banks inside the attached SDRAM devices in the following way: • When IBANK = 0, 1 internal bank is used • When IBANK = 1h, 2 internal banks are used • When IBANK = 2h, 4 internal banks are used This field value affects the mapping of logical addresses to the SDRAM row, column, and bank addresses. See Section 12.2.5.11 for details.
PAGESIZE	**Page Size.** This field defines the internal page size of the attached SDRAM devices in the following way: • When PAGESIZE = 0, 256-word pages are used • When PAGESIZE = 1h, 512-word pages are used • When PAGESIZE = 2h, 1024-word pages are used • When PAGESIZE = 3h, 2048-word pages are used This field value affects the mapping of logical addresses to the SDRAM row, column, and bank addresses. See Section 12.2.5.11 for details.

Table 12-9. Description of the SDRAM Refresh Control Register (SDRAM_RCR)

Parameter	Description
RR	**Refresh Rate**. This field controls the rate at which attached SDRAM devices are refreshed. The following equation can be used to determine the required value of RR for an SDRAM device: • RR = f_{EM1CLK} / (Required SDRAM Refresh Rate) More information about the operation of the SDRAM refresh controller is found in Section 12.2.5.6.

Table 12-10. Description of the SDRAM Timing Register (SDRAM_TR)

Parameter	Description
T_RFC T_RP T_RCD T_WR T_RAS T_RC T_RRD	**SDRAM Timing Parameters.** These fields configure the EMIF to comply with the AC timing requirements of the attached SDRAM devices. This allows the EMIF to avoid violating SDRAM timing constraints and to more efficiently schedule operations. More details about each of these parameters can be found in the SDRAM_TR register description. These parameters must be set to satisfy the corresponding timing requirements found in the SDRAM data sheet.

Table 12-11. Description of the SDRAM Self Refresh Exit Timing Register (SDR_EXT_TMNG)

Parameter	Description
T_XS	**Self Refresh Exit Parameter.** The T_XS field of this register informs the EMIF about the minimum number of EM1CLK cycles required between exiting self-refresh and issuing any command. This parameter must be set to satisfy the t_{XSR} value for the attached SDRAM device.

12.2.5.4 SDRAM Auto-Initialization Sequence

The EMIF automatically performs an SDRAM initialization sequence, regardless of whether the EMIF is interfaced to an SDRAM device, when either of the following two events occur:

- The EMIF comes out of reset. No memory accesses to the SDRAM and asynchronous interfaces are performed until this auto-initialization is complete.
- A write is performed to any of the three least significant bytes of the SDRAM configuration register (SDRAM_CR)

An SDRAM initialization sequence consists of the following steps:

1. If the initialization sequence is activated by a write to SDRAM_CR, and if any of the SDRAM banks are open, the EMIF issues a PRE command with EM1A[10] held high to indicate all banks. This is done so that the maximum ACTV to PRE timing for an SDRAM is not violated.
2. The EMIF drives EM1SDCKE high and begins continuously issuing NOP commands until eight SDRAM refresh intervals have elapsed. An SDRAM refresh interval is equal to the value of the RR field of the SDRAM refresh control register (SDRAM_RCR), divided by the frequency of EM1CLK (RR/f_{EM1CLK}). This step is used to avoid violating the power-up constraint of most SDRAM devices that requires 200 µs (sometimes 100 µs) between receiving stable Vdd and CLK and the issuing of a PRE command. Depending on the frequency of EM1CLK, this step may or may not be sufficient to avoid violating the SDRAM constraint. See Section 12.2.5.5 for more information.
3. After the refresh intervals have elapsed, the EMIF issues a PRE command with EM1A[10] held high to indicate all banks.
4. The EMIF issues eight AUTO REFRESH commands.
5. The EMIF issues the LMR command with the EM1A[9:0] pins set as described in Table 12-12.
6. Finally, the EMIF performs a refresh cycle, which consists of the following steps:
   a. Issuing a PRE command with EM1A[10] held high if any banks are open
   b. Issuing an REF command

Table 12-12. SDRAM LOAD MODE REGISTER Command

EM1A[9:7]	EM1A[6:4]	EM1A[3]	EM1A[2:0]
0 (Write bursts are of the programmed burst length in EM1A[2:0])	These bits control the CAS latency of the SDRAM and are set according to CL field in the SDRAM configuration register (SDRAM_CR) as follows: • If CL = 2, EM1A[6:4] = 2h (CAS latency = 2) • If CL = 3, EM1A[6:4] = 3h (CAS latency = 3)	0 (Sequential Burst Type. Interleaved Burst Type not supported)	These bits control the burst length of the SDRAM and are set according to the NM field in the SDRAM configuration register (SDRAM_CR) as follows: • If NM = 0, EM1A[2:0] = 2h (Burst Length = 4) • If NM = 1, EM1A[2:0] = 3h (Burst Length = 8)

12.2.5.5 SDRAM Configuration Procedure

There are two different SDRAM configuration procedures. Although the EMIF automatically performs the SDRAM initialization sequence described in Section 12.2.5.4 when coming out of reset, it is recommended to follow one of the procedures before performing any EMIF memory requests.

Procedure A must be followed if the SDRAM power-up constraint was not violated during the SDRAM auto-initialization sequence detailed in Section 12.2.5.4 on coming out of Reset. The SDRAM power-up constraint specifies that 200 µs (sometimes 100 µs) must exist between receiving stable Vdd and CLK and the issuing of a PRE command.

Procedure B must be followed if the SDRAM power-up constraint was violated. The 200 µs (100 µs) SDRAM power-up constraint is violated, if the frequency of EM1CLK is greater than 50 MHz (100 MHz for 100 µs SDRAM power-up constraint) during SDRAM Auto-Initialization Sequence. Procedure B must be followed if there is any doubt that the power-up constraint was not met.

Procedure A — Following is the procedure to be followed if the SDRAM power-up constraint was NOT violated:

1. Place the SDRAM into self-refresh mode by setting the SR bit of SDRAM_CR to 1. The SDRAM can be placed into self-refresh mode when changing the frequency of the EM1CLK to avoid incurring the 200 µs power-up constraint again.
2. Configure the desired EMIF1 clock (EM1CLK) frequency. The frequency of the memory clock must meet the timing requirements in the SDRAM manufacturer's documentation and the timing limitations shown in the electrical specifications of the device data manual.
3. Remove the SDRAM from self-refresh mode by clearing the SR bit of the SDRAM_CR to 0.
4. Program SDRAM_TR and SDR_EXT_TMNG to satisfy the timing requirements for the attached SDRAM device. The timing parameters must be taken from the SDRAM data sheet.
5. Program the RR field of SDRAM_RCR to match that of the attached device's refresh interval. See Section 12.2.5.6.1 details on determining the appropriate value.
6. Program the SDRAM_CR to match the characteristics of the attached SDRAM device. This causes the auto-initialization sequence in Section 12.2.5.4 to be re-run. This second initialization generally takes much less time due to the increased frequency of EM1CLK.

Procedure B — Following is the procedure to be followed if the SDRAM power-up constraint was violated:

1. Configure the desired EM1CLK clock frequency. The frequency of the memory clock must meet the timing requirements in the SDRAM manufacturer's documentation and the timing limitations shown in the electrical specifications of the device data manual.
2. Program SDRAM_TR and SDR_EXT_TMNG to satisfy the timing requirements for the attached SDRAM device. The timing parameters must be taken from the SDRAM data sheet.
3. Program the RR field of the SDRAM_RCR such that the following equation is satisfied: $(RR \times 8)/(f_{EM1CLK}) >$ 200 µs (sometimes 100 µs). For example, an EM1CLK frequency of 100 MHz requires setting RR to 2501 (9C5h) or higher to meet a 200 µs constraint.
4. Program the SDRAM_CR to match the characteristics of the attached SDRAM device. This causes the auto-initialization sequence in Section 12.2.5.4 to be re-run with the new value of RR.
5. Perform a read from the SDRAM to make sure that step 5 of this procedure occurs after the initialization process has completed. Alternatively, wait for 200 µs instead of performing a read.
6. Finally, program the RR field to match that of the attached device's refresh interval. See Section 12.2.5.6.1 details on determining the appropriate value.

After following the above procedure, the EMIF is ready to perform accesses to the attached SDRAM device.

12.2.5.6 EMIF Refresh Controller

An SDRAM device requires that each of the rows be refreshed at a minimum required rate. The EMIF can meet this constraint by performing auto refresh cycles at or above this required rate. An auto-refresh cycle consists of issuing a PRE command to all banks of the SDRAM device followed by issuing a REFR command. To inform the EMIF of the required rate for performing auto refresh cycles, the RR field of the SDRAM refresh control register (SDRAM_RCR) must be programmed. The EMIF uses this value along with two internal counters to automatically perform auto refresh cycles at the required rate. The auto-refresh cycles cannot be disabled, even if the EMIF is not interfaced with an SDRAM. The remainder of this section details the EMIF's refresh scheme and provides an example for determining the appropriate value to place in the RR field of the SDRAM_RCR.

The two counters used to perform auto-refresh cycles are a 13-bit refresh interval counter and a 4-bit refresh backlog counter. At reset and upon writing to the RR field, the refresh interval counter is loaded with the value from RR field and begins decrementing, by one, each EMIF clock cycle. When the refresh interval counter reaches zero, the following actions occur:

- The refresh interval counter is reloaded with the value from the RR field and restarts decrementing.
- The 4-bit refresh backlog counter increments unless the counter has already reached the maximum value.

The refresh backlog counter records the number of auto refresh cycles that the EMIF currently has outstanding. This counter is decremented by one each time an auto refresh cycle is performed and incremented by one each time the refresh interval counter expires. The refresh backlog counter saturates at the values of 0000b and 1111b. The EMIF uses the refresh backlog counter to determine the urgency with which an auto refresh cycle must be performed. The four levels of urgency are described in Table 12-13. This refresh scheme allows the required refreshes to be performed with minimal impact on access requests.

Table 12-13. Refresh Urgency Levels

Urgency Level	Refresh Backlog Counter Range	Action Taken
Refresh May	1-3	An auto-refresh cycle is performed only if the EMIF has no requests pending and none of the SDRAM banks are open.
Refresh Release	4-7	An auto-refresh cycle is performed if the EMIF has no requests pending, regardless of whether any SDRAM banks are open.
Refresh Need	8-11	An auto-refresh cycle is performed at the completion of the current access unless there are read requests pending.
Refresh Must	12-15	Multiple auto-refresh cycles are performed at the completion of the current access until the Refresh Release urgency level is reached. At that point, the EMIF can begin servicing any new read or write requests.

12.2.5.6.1 Determining the Appropriate Value for the RR Field

The value that must be programmed into the RR field of the SDRAM_RCR must can be calculated by using the frequency of the EM1CLK signal (f_{EM1CLK}) and the required refresh rate of the SDRAM ($f_{Refresh}$). The following formula can be used:

$RR = f_{EM1CLK} / f_{Refresh}$

The SDRAM data sheet often communicates the required SDRAM Refresh Rate in terms of the number of REFR commands required in a given time interval. The required SDRAM Refresh Rate in the formula above can therefore be calculated by dividing the number of required cycles per time interval (n_{cycles}) by the time interval given in the data manual ($t_{Refresh\ Period}$) :

$f_{Refresh} = n_{cycles} / t_{Refresh\ Period}$

Combining these formulas, the value that must be programmed into the RR field can be computed as:

$RR = f_{EM1CLK} \times t_{Refresh\ Period} / n_{cycles}$

The following example illustrates calculating the value of RR. Given that:

- f_{EM1CLK} = 100 MHz (frequency of EMIF clock)
- $t_{Refresh\ Period}$ = 64 ms (required refresh interval of the SDRAM)
- n_{cycles} = 8192 (number of cycles in a refresh interval for the SDRAM)

RR can be calculated as:

RR = 100 MHz × 64 ms/8192

RR = 781.25

RR = 782 cycles = 30Eh cycles

12.2.5.7 Self-Refresh Mode

The EMIF can be programmed to enter the self-refresh state by setting the SR bit of SDRAM_CR to 1. This causes the EMIF to issue the SLFR command after completing any outstanding SDRAM access requests and clearing the refresh backlog counter by performing one or more auto refresh cycles. This places the attached SDRAM device into self-refresh mode in which the EMIF consumes a minimal amount of power while performing the refresh cycles.

While in the self-refresh state, the EMIF continues to service asynchronous bank requests and register accesses as normal, with one caveat. The EMIF does not park the data bus following a read to asynchronous memory while in the self-refresh state. Instead, the EMIF tri-states the data bus. Therefore, it is not recommended to perform asynchronous read operations while the EMIF is in the self-refresh state to prevent floating inputs on the data bus. More information about data bus parking can be found in Section 12.2.7.

The EMIF exits from the self-refresh state, if either of the following events occur:

- The SR bit of SDRAM_CR is cleared to 0.
- An SDRAM accesses is requested.

The EMIF exits from the self-refresh state by driving EM1SDCKE high and performing an auto refresh cycle.

The attached SDRAM device must also be placed into self-refresh mode when changing the frequency of EM1CLK. If the frequency of EM1CLK changes while the SDRAM is not in self-refresh mode, Procedure B in Section 12.2.5.5 must be followed to reinitialize the device.

12.2.5.8 Power-Down Mode

To support low-power modes, the EMIF can be requested to issue a POWER DOWN command to the SDRAM by setting the PD bit in the SDRAM configuration register (SDRAM_CR). When this bit is set, the EMIF continues normal operation until all outstanding memory access requests have been serviced and the SDRAM refresh backlog (if there is one) has been cleared. At this point, the EMIF enters the power-down state. Upon entering this state, the EMIF issues a POWER DOWN command (same as a NOP command but driving the EM1SDCKE low on the same cycle). The EMIF then maintains the EM1SDCKE low until the EMIF exits the power-down state.

Since the EMIF services the refresh backlog before the EMIF enters the power-down state, all internal banks of the SDRAM are closed (precharged) prior to issuing the POWER DOWN command. Therefore, the EMIF only supports precharge power-down. The EMIF does not support active power-down, where internal banks of the SDRAM are open (active) before the POWER DOWN command is issued.

During the power-down state, the EMIF services the SDRAM, asynchronous memory, and register accesses as normal, returning to the power-down state upon completion.

The PDWR bit in the SDRAM_CR indicates whether the EMIF must perform refreshes in power-down state. If the PDWR bit is set, the EMIF exits the power-down state every time the Refresh Must level is set, performs AUTO REFRESH commands to the SDRAM, and returns back to the power-down state. This evenly distributes the refreshes to the SDRAM in power-down state. If the PDWR bit is not set, the EMIF does not perform any refreshes to the SDRAM. Therefore, the data integrity of the SDRAM is not maintained upon power-down exit, if the PDWR bit is not set.

If the PD bit is cleared while in the power-down state, the EMIF comes out of the power-down state. The EMIF:

- Drives EM1SDCKE high
- Enters the idle state

12.2.5.9 SDRAM Read Operation

When the EMIF receives a read request to the SDRAM from one of the requesters listed in Section 12.2.2, the EMIF performs one or more read access cycles. A read access cycle begins with the issuing of the ACTV command to select the desired bank and row of the SDRAM device. After the row has been opened, the EMIF proceeds to issue a READ command while specifying the desired bank and column address. EM1A[10] is held low during the READ command to avoid auto-precharging. The READ command signals the SDRAM device to output data from the specified address while EMIF issues NOP commands. Following a READ command, the CL field of the SDRAM configuration register (SDRAM_CR) defines how many delay cycles are present before the read data appears on the data bus. This is referred to as the CAS latency.

Figure 12-6 shows the signal waveforms for a basic SDRAM read operation in which multiple data is read from a single page. On this device, burst accesses are not supported; hence, the EMIF issues a READ command for each data access. Only when the EMIF SDRAM interface is configured to 16-bit by setting the NM bit of the SDRAM configuration register (SDRAM_CR) to 1 and CPU (or any other master) does a 32-bit READ access, a burst access is issued with a size of two.

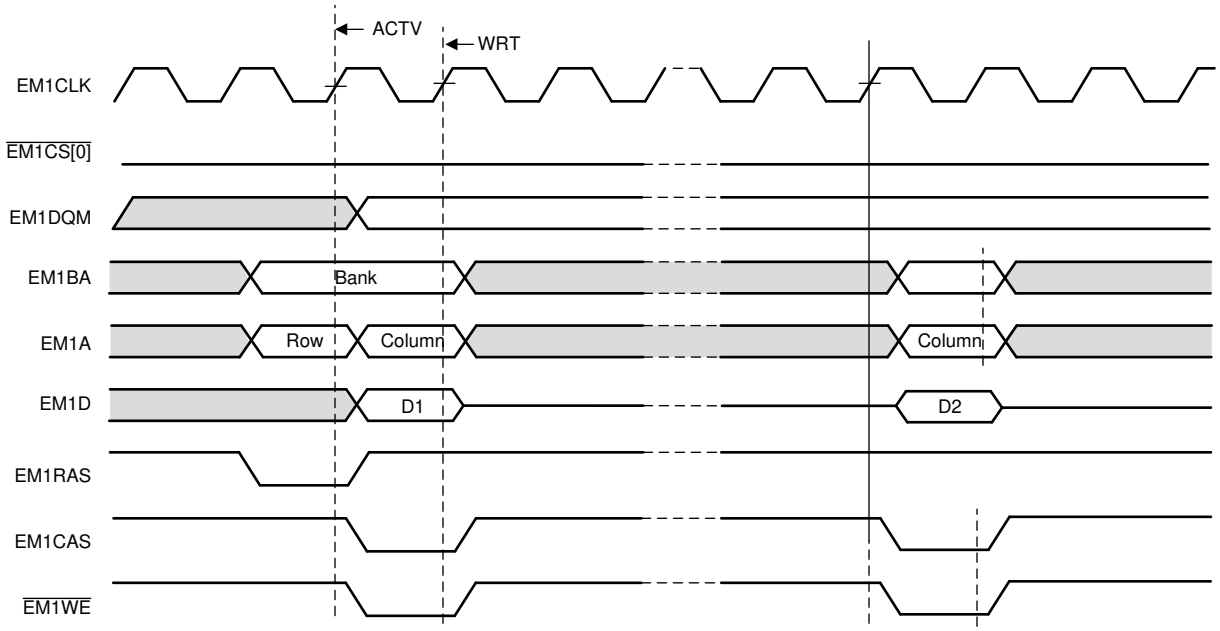


Figure 12-6. Timing Waveform for Basic SDRAM Read Operation

Several other pins are also active during a read access. The EM1DQM[x:0] pins are driven low during the READ commands and are kept low during the NOP commands that correspond to the burst request. The state of the other EMIF pins during each command can be found in Table 12-6.

The EMIF schedules the commands based on the timing information that is provided to the EMIF in the SDRAM timing register (SDRAM_TR). The values for the timing parameters in this register must be chosen to satisfy the timing requirements listed in the SDRAM data manual. The EMIF uses this timing information to avoid violating any timing constraints related to issuing commands. This is commonly accomplished by inserting NOP commands between various commands during an access. Refer to the register description of SDRAM_TR in the SDTIMER register for more details on the various timing parameters.

12.2.5.10 SDRAM Write Operations

When the EMIF receives a write request to SDRAM from one of the requesters listed in Section 12.2.2, the EMIF performs one or more write-access cycles. A write-access cycle begins with the issuing of the ACTV command to select the desired bank and row of the SDRAM device. After the row has been opened, the EMIF proceeds to issue a WRT command while specifying the desired bank and column address. EM1A[10] is held low during the WRT command to avoid auto-precharging. The WRT command signals the SDRAM device to start writing the given data to the specified address while the EMIF issues NOP commands. On this device, burst accesses are not supported; hence, the EMIF issues a WRITE command for each data access. Only when the EMIF SDRAM interface is configured to 16-bit by setting the NM bit of the SDRAM configuration register (SDRAM_CR) to 1 and CPU (or any other master) does a 32bit WRITE access, a burst access is issued with size of two.

Figure 12-7 shows the signal waveforms for a basic SDRAM write operation.

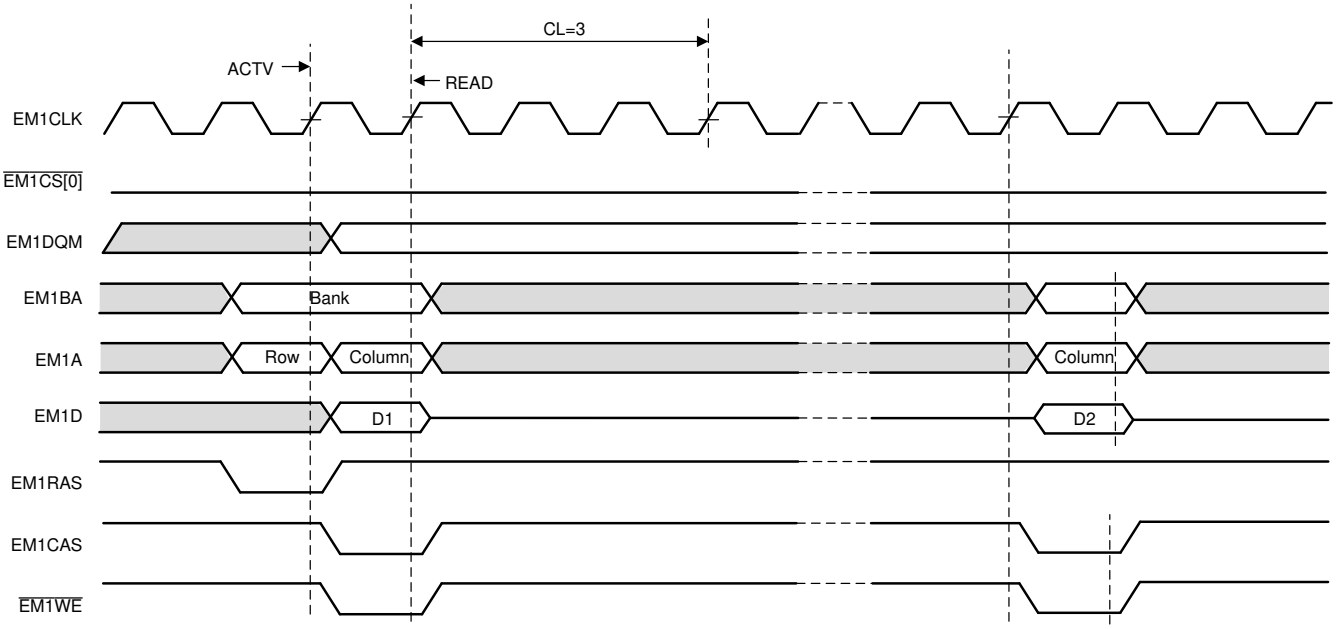


Figure 12-7. Timing Waveform for Basic SDRAM Write Operation

The EMIF truncates a series of bursting data if the remaining addresses of the burst are not part of the write request. The EMIF can truncate the burst in three ways:

- By issuing another WRT to the same page
- By issuing a PRE command to prepare for accessing a different page of the same bank
- By issuing a BT command to prepare for accessing a page in a different bank

Several other pins are also active during a write access. The EM1DQM[x:0] pins are driven to select which bytes of the data word are written to the SDRAM device. The pins are also used to mask out entire undesired data words during a burst access. The state of the other EMIF pins during each command can be found in Table 12-6.

The EMIF schedules the commands based on the timing information that is provided to the EMIF in the SDRAM timing register (SDRAM_TR). The values for the timing parameters in this register must be chosen to satisfy the timing requirements listed in the SDRAM data sheet. The EMIF uses this timing information to avoid violating any timing constraints related to issuing commands. This is commonly accomplished by inserting NOP commands during various cycles of an access. Refer to the register description of SDRAM_TR in the SDTIMR register for more details on the various timing parameters.

12.2.5.11 Mapping from Logical Address to EMIF Pins

When the EMIF receives an SDRAM access request, it must convert the address of the access into the appropriate signals to send to the SDRAM device. The details of this address mapping are shown in Table 12-14 for 32-bit operation and in Table 12-15 for 16-bit operation. Using the settings of the IBANK and PAGESIZE fields of the SDRAM configuration register (SDRAM_CR), the EMIF determines which bits of the logical address will be mapped to the SDRAM row, column, and bank addresses.

As the logical address is incremented by one halfword (16-bit operation), the column address is likewise incremented by one until a page boundary is reached. When the logical address increments across a page boundary, the EMIF moves into the same page in the next bank of the attached device by incrementing the bank address EM1BA and resetting the column address. The page in the previous bank is left open until it is necessary to close it. This method of traversal through the SDRAM banks helps maximize the number of open banks inside of the SDRAM and results in an efficient use of the device. There is no limitation on the number of banks that can be open at one time, but only one page within a bank can be open at a time.

The EMIF uses the EM1DQM[3:0] pins during a WRT command to mask out selected bytes or entire words. The EM1DQM[3:0] pins are always low during a READ command.

Table 12-14. Mapping from Logical Address to EMIF Pins for 32-bit SDRAM

IBANK	PAGESIZE	Logical Address													
		31:27	26	25	24	23	22	21:14	13	12	11	10	9	8:1	0
0	0	-						Row Address						Col Address	EM1DQM[2]/EM1DQM[3]
1	0	-						Row Address					EM1BA[0]	Col Address	EM1DQM[2]/EM1DQM[3]
2	0	-						Row Address				EM1BA[1:0]		Col Address	EM1DQM[2]/EM1DQM[3]
0	1	-						Row Address					Column Address		EM1DQM[2]/EM1DQM[3]
1	1	-						Row Address			EM1BA[0]		Column Address		EM1DQM[2]/EM1DQM[3]
2	1	-						Row Address			EM1BA[1:0]		Column Address		EM1DQM[2]/EM1DQM[3]
0	2	-						Row Address				Column Address			EM1DQM[2]/EM1DQM[3]
1	2	-						Row Address			EM1BA[0]	Column Address			EM1DQM[2]/EM1DQM[3]
2	2	-						Row Address		EM1BA[1:0]		Column Address			EM1DQM[2]/EM1DQM[3]
0	3	-						Row Address			Column Address				EM1DQM[2]/EM1DQM[3]
1	3	-						Row Address		EM1BA[0]	Column Address				EM1DQM[2]/EM1DQM[3]
2	3	-						Row Address		EM1BA[1:0]	Column Address				EM1DQM[2]/EM1DQM[3]

Table 12-15. Mapping from Logical Address to EMIF Pins for 16-bit SDRAM

IBANK	PAGESIZE	Logical Address												
		31:26	25	24	23	22	21	20:13	12	11	10	9	8	7:0
0	0	-						Row Address						Col Address
1	0	-						Row Address					EM1BA[0]	Col Address
2	0	-						Row Address				EM1BA[1:0]		Col Address
0	1	-						Row Address					Column Address	
1	1	-						Row Address				EM1BA[0]	Column Address	
2	1	-						Row Address			EM1BA[1:0]		Column Address	
0	2	-						Row Address				Column Address		
1	2	-						Row Address			EM1BA[0]	Column Address		
2	2	-						Row Address		EM1BA[1:0]		Column Address		
0	3	-						Row Address			Column Address			
1	3	-						Row Address		EM1BA[0]	Column Address			
2	3	-						Row Address		EM1BA[1:0]	Column Address			

Note

The upper bit of the row address is used only when addressing 256-Mbit and 512-Mbit SDRAM memories.

12.2.6 Asynchronous Controller and Interface

The EMIF easily interfaces to a variety of asynchronous devices including NOR Flash and SRAM. It can be operated in two major modes (see Table 12-16):

- Normal Mode
- Select Strobe Mode

Table 12-16. Normal Mode vs. Select Strobe Mode

Mode	Function of EM1DQM pins	Operation of $\overline{\text{EM1CS}}$[4:2]
Normal Mode	Byte enables	Active during the entire asynchronous access cycle
Select Strobe Mode	Byte enables	Active only during the strobe period of an access cycle

The first mode of operation is normal mode, in which the EM1DQM pins of the EMIF function as byte enables. In this mode, the $\overline{\text{EM1CS}}$[4:2] pins behave as typical chip select signals, remaining active for the duration of the asynchronous access. See Section 12.2.6.1 for an example interface with multiple 8-bit devices.

The second mode of operation is select strobe mode, in which the $\overline{\text{EM1CS}}$[4:2] pins act as a strobe, active only during the strobe period of an access. In this mode, the EM1DQM pins of the EMIF function as standard byte enables for reads and writes. A summary of the differences between the two modes of operation are shown in Table 12-16. Refer to Section 12.2.6.4 for the details of asynchronous operations in normal mode, and to Section 12.2.6.5 for the details of asynchronous operations in select strobe mode. The EMIF hardware defaults to normal mode, but can be manually switched to select strobe mode by setting the SS bit in the asynchronous *m* (m = 1, 2, 3, or 4) configuration register (CE*n*CFG) (*n* = 2, 3, or 4). Throughout the chapter, *m* can hold the values 1, 2, 3 or 4; and *n* can hold the values 2, 3, or 4.

In both normal mode and select strobe mode, EMIF can be configured to operate in a sub-mode called NAND Flash Mode. In NAND Flash Mode, the EMIF is able to calculate an error correction code (ECC) for transfers up to 512 bytes.

The EMIF also provides configurable cycle timing parameters and an extended wait mode that allows the connected device to extend the strobe period of an access cycle. The following sections describe the features related to interfacing with external asynchronous devices.

12.2.6.1 Interfacing to Asynchronous Memory

Figure 12-8 shows the EMIF's external pins used in interfacing with an asynchronous device. In $\overline{\text{EM1CS}}$[n], n = 2, 3, or 4.

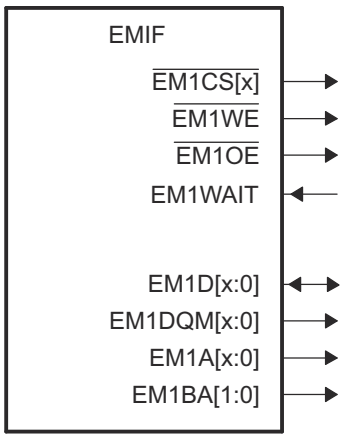
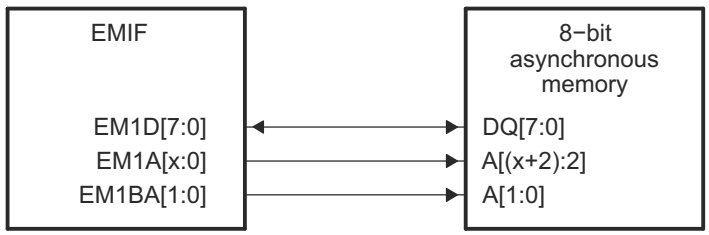


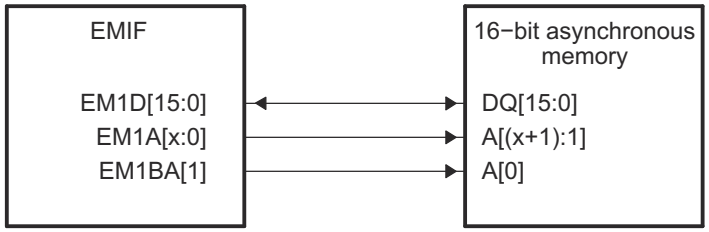
Figure 12-8. EMIF Asynchronous Interface

Of special note is the connection between the EMIF and the external device's address bus. The EMIF address pin EM1A[0] always provides the least-significant bit of a 32-bit word address. Therefore, when interfacing to a 16-bit or 8-bit asynchronous device, the EM1BA[1] and EM1BA[0] pins provide the least-significant bits of the halfword or byte address, respectively. Figure 12-9 and Figure 12-10 show the mapping between the EMIF and the connected device's data and address pins for various programmed data bus widths. The data bus width can be configured in the asynchronous *n* configuration register (ASYNC_CS*n*_CR).

Figure 12-10 shows an interface between the EMIF and an external memory with byte enables. The EMIF must be operated in either normal mode or select strobe mode when using this interface, so that the EM1DQM signals operate as byte enables.



a) EMIF to 8-bit memory interface



b) EMIF to 16-bit memory interface

Figure 12-9. EMIF to 8-bit/16-bit Memory Interface

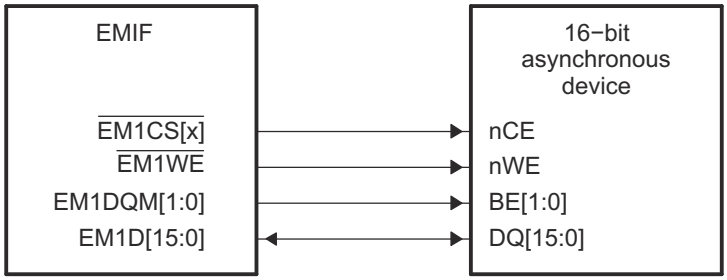


Figure 12-10. Common Asynchronous Interface

12.2.6.2 Accessing Larger Asynchronous Memories

If a device such as a large asynchronous Flash needs to be attached to the EMIF, then GPIO pins may be used to control the Flash device upper address lines.

12.2.6.3 Configuring EMIF for Asynchronous Accesses

The operation of the EMIF's asynchronous interface can be configured by programming the appropriate register fields. The reset value and bit position for each register field can be found in Section 12.5. The following tables list the register fields that can be programmed and describe the purpose of each field. These registers can be programmed prior to accessing the external memory, and the transfer following a write to these registers uses the new configuration.

Table 12-17. Description of the Asynchronous *m* Configuration Register (ASYNC_CSn_CR)

Parameter	Description
SS	**Select Strobe mode.** This bit selects the EMIF's mode of operation in the following way: • SS = 0 selects Normal Mode – EM1DQM pins function as byte enables – $\overline{\text{EM1CS}}$[4:2] active for duration of access • SS = 1 selects Select Strobe Mode – EM1DQM pins function as byte enables – $\overline{\text{EM1CS}}$[4:2] acts as a strobe.
EW	**Extended Wait Mode enable.** • EW = 0 disables Extended Wait Mode • EW = 1 enables Extended Wait Mode When set to 1, the EMIF enables the Extended Wait Mode in which the strobe width of an access cycle can be extended in response to the assertion of the EM1WAIT pin. The WP*n* bit in the asynchronous wait cycle configuration register (AWCC) controls the polarity of the EM1WAIT pin. Extended Wait Mode must not be used while in NAND Flash Mode. See Section 12.2.6.6 for more details on this mode of operation.
W_SETUP/R_SETUP	**Read/Write setup widths.** These fields define the number of EMIF clock cycles of setup time for the address pins (EM1A), byte enables (EM1DQM), and asynchronous chip enable ($\overline{\text{EM1CS}}$[4:2]) before the read strobe pin (EM103) or write strobe pin ($\overline{\text{EM1WE}}$) falls, minus one cycle. For writes, the W_SETUP field also defines the setup time for the data pins (EM1D). Refer to the asynchronous device 's data sheet to determine the appropriate setting for this field.
W_STROBE/R_STROBE	**Read/Write strobe widths.** These fields define the number of EMIF clock cycles between the falling and rising of the read strobe pin (EM103) or write strobe pin (EM1WEn), minus one cycle. If Extended Wait Mode is enabled by setting the EW field in the asynchronous *n* configuration register (ASYNC_CS*n*_CR), these fields must be set to a value greater than zero. Refer to the data manual of the external asynchronous device to determine the appropriate setting for this field.
W_HOLD/R_HOLD	**Read/Write hold widths.** These fields define the number of EMIF clock cycles of hold time for the address pins (EM1A and EM1BA), byte enables (EM1DQM), and asynchronous chip enable ($\overline{\text{EM1CS}}$[4:2]) after the read strobe pin (EM103) or write strobe pin ($\overline{\text{EM1WE}}$) rises, minus one cycle. For writes, the W_HOLD field also defines the hold time for the data pins (EM1D). Refer to the data manual of the external asynchronous device to determine the appropriate setting for this field.
TA	**Minimum turnaround time.** This field defines the minimum number of EMIF clock cycles between asynchronous reads and writes, minus one cycle. The purpose of this feature is to avoid contention on the bus. The value written to this field also determines the number of cycles that are inserted between asynchronous accesses and SDRAM accesses. Refer to the data manual of the external asynchronous device to determine the appropriate setting for this field.

Table 12-17. Description of the Asynchronous *m* Configuration Register (ASYNC_CSn_CR) (continued)

Parameter	Description
ASIZE	**Asynchronous Device Bus Width.** This field determines the data bus width of the asynchronous interface in the following way: • ASIZE = 0 selects an 8-bit bus • ASIZE = 1 selects a 16-bit bus • ASIZE = 2 selects a 32-bit bus The configuration of ASIZE determines the function of the EM1A and EM1BA pins as described in Section 12.2.6.1. This field also determines the number of external accesses required to fulfill a request generated by one of the sources mentioned in Section 12.2.2. For example, a request for a 32-bit word requires four external access when ASIZE = 0. Refer to the data manual of the external asynchronous device to determine the appropriate setting for this field.

Table 12-18. Description of the Asynchronous Wait Cycle Configuration Register (ASYNC_WCCR)

Parameter	Description
WP*n*	**EM_WAIT Polarity.** • WP*n* = 0 selects active-low polarity • WP*n* = 1 selects active-high polarity When set to 1, the EMIF waits if the EM1WAIT pin is high. When cleared to 0, the EMIF waits if the EM1WAIT pin is low. The EMIF must have the Extended Wait Mode enabled for the EM1WAIT pin to affect the width of the strobe period. The polarity of the EM1WAIT signal is not programmable in NAND Flash Mode.
MAX_EXT_WAIT	**Maximum Extended Wait Cycles.** This field configures the number of EMIF clock cycles the EMIF waits for the EM1WAIT pin to be deactivated during the strobe period of an access cycle. The maximum number of EMIF clock cycles the EMIF waits is determined by the following formula: Maximum Extended Wait Cycles = (MAX_EXT_WAIT + 1) × 16 If the EM1WAIT pin is not deactivated within the time specified by this field, the EMIF resumes the access cycle, registering whatever data is on the bus and proceeding to the hold period of the access cycle. This situation is referred to as an Asynchronous Timeout. An Asynchronous Timeout generates an interrupt, if the interrupt has been enabled in the EMIF interrupt mask set register (INT_MSK_SET). Refer to Section 12.2.9.1 for more information about EMIF interrupts. Extended Wait Mode can not be used while in NAND Flash Mode.

Table 12-19. Description of EMIF Interrupt Mask Set Register (INT_MSK_SET)

Parameter	Description
WR_MASK_SET	**Wait Rise Mask Set.** Writing a 1 enables an interrupt to be generated when a rising edge on EM1WAIT occurs.
AT_MASK_SET	**Asynchronous Timeout Mask Set.** Writing a 1 to this bit enables an interrupt to be generated when an Asynchronous Timeout occurs.

Table 12-20. Description of EMIF Interrupt Mast Clear Register (INT_MSK_CLR)

Parameter	Description
WR_MASK_CLR	**Wait Rise Mask Clear.** Writing a 1 to this bit disables the interrupt, clearing the WR_MASK_SET bit in EMIF interrupt mask set register (INT_MSK_SET).
AT_MASK_CLR	**Asynchronous Timeout Mask Clear.** Writing a 1 to this bit prevents an interrupt from being generated when an Asynchronous Timeout occurs.

Note

The EMIF performs SDRAM refreshes even if the SDRAM interface is not used. If using only the ASRAM interface, then SDRAM refreshes impact the ASRAM performance. To avoid this, set PD=1 in the SDRAM_CR register (Emif1Regs.SDRAM_CR.bit.PD=1). This bit can be updated only if there are no pending EMIF accesses.

12.2.6.4 Read and Write Operations in Normal Mode

Normal mode is the asynchronous interface's default mode of operation. It is selected when the SS bit in the asynchronous *n* configuration register (ASYNC_CS*n*_CR) is cleared to 0. In this mode, the EM1DQM pins operate as byte enables. Section 12.2.6.4.1 and Section 12.2.6.4.2 explain the details of read and write operations while in normal mode.

12.2.6.4.1 Asynchronous Read Operations (Normal Mode)

Note

During an entire asynchronous read operation, the $\overline{\text{EM1WE}}$ pin is driven high.

An asynchronous read is performed when any of the requesters mentioned in Section 12.2.2 request a read from the attached asynchronous memory. After the request is received, a read operation is initiated once the request becomes the EMIF's highest priority task, according to the priority scheme detailed in Section 12.2.13. In the event that the read request cannot be serviced by a single access cycle to the external device, multiple access cycles are performed by EMIF until the entire request is fulfilled. The details of an asynchronous read operation in normal mode are described in Table 12-21. Also, Figure 12-11 shows an example timing diagram of a basic read operation.

Table 12-21. Asynchronous Read Operation in Normal Mode

Time Interval	Pin Activity in Normal Mode
Turnaround period	Once the read operation becomes the highest priority task for EMIF, the EMIF waits for the programmed number of turn-around cycles before proceeding to the setup period of the operation. The number of wait cycles is taken directly from the TA field of the asynchronous *n* configuration register (ASYNC_CS*n*_CR). Between each access (write or read), the EMIF inserts two cycles of delay even though TA field is programmed as 0. After the EMIF has waited for the turnaround cycles to complete, the EMIF again checks to make sure that the read operation is still the highest priority task. If so, the EMIF proceeds to the setup period of the operation. If the read operation is no longer the highest priority task, the EMIF terminates the operation.
Start of the setup period	The following actions occur at the start of the setup period: • The setup, strobe, and hold values are set according to the R_SETUP, R_STROBE, and R_HOLD values in ASYNC_CS*n*_CR. • The address pins EM1A and EM1BA become valid and carry the values described in Section 12.2.6.1. • $\overline{\text{EM1CS}}$[4:2] falls to enable the external device (if not already low from a previous operation)

Table 12-21. Asynchronous Read Operation in Normal Mode (continued)

Time Interval	Pin Activity in Normal Mode
Strobe period	The following actions occur during the strobe period of a read operation: 1. $\overline{\text{EM1OE}}$ falls at the start of the strobe period 2. On the rising edge of the clock that is concurrent with the end of the strobe period: • $\overline{\text{EM1OE}}$ rises • The data on the EM1Dx bus is sampled by EMIF. In Figure 12-11, EM1WAIT is inactive. If EM1WAIT is instead activated, the strobe period can be extended by the external device to give it more time to provide the data. Section 12.2.6.6 contains more details on using the EM1WAIT pin.
End of the hold period	At the end of the hold period: • The address pins EM1A and EM1BA become invalid • $\overline{\text{EM1CS}}$[4:2] rises (if no more operations are required to complete the current request) The EMIF can be required to issue additional read operations to a device with a small data bus width to complete an entire word access. In this case, the EMIF immediately re-enters the setup period to begin another operation without incurring the turn-round cycle delay. The setup, strobe, and hold values are not updated in this case. If the entire word access has been completed, the EMIF returns to the previous state unless another asynchronous request has been submitted and is currently the highest priority task. If this is the case, EMIF instead enters directly into the turnaround period for the pending read or write operation.

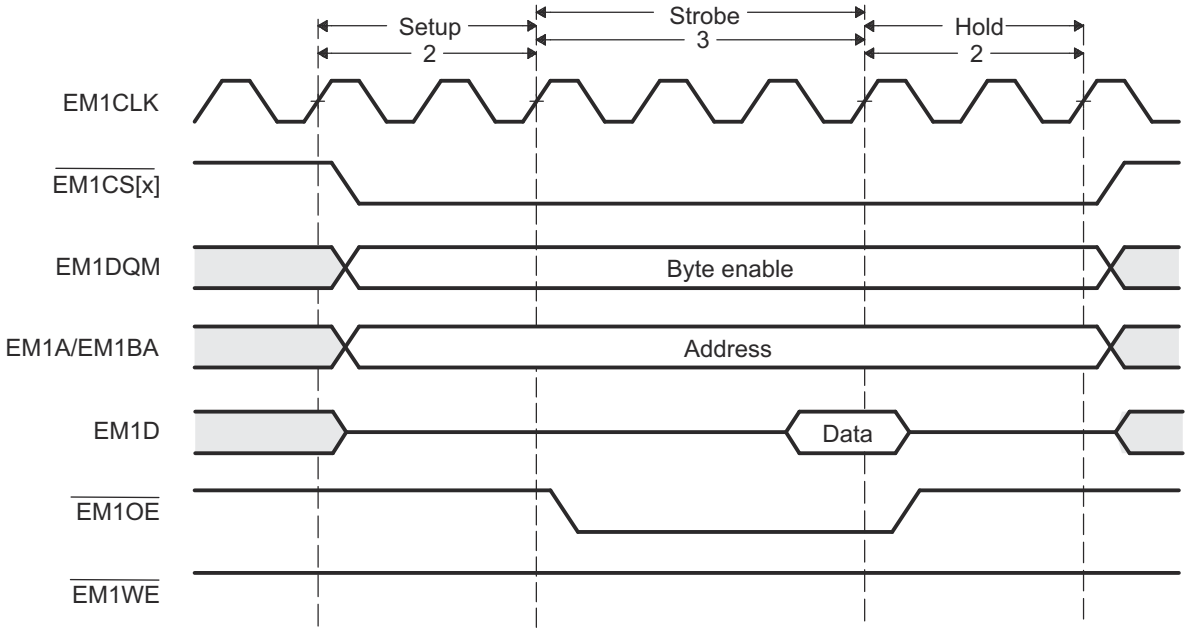


Figure 12-11. Timing Waveform of an Asynchronous Read Cycle in Normal Mode

12.2.6.4.2 Asynchronous Write Operations (Normal Mode)

Note

During an entire asynchronous write operation, the $\overline{\text{EM1OE}}$ pin is driven high.

An asynchronous write is performed when any of the requesters mentioned in Section 12.2.2 request a write to memory in the asynchronous bank of EMIF. After the request is received, a write operation is initiated once the request becomes the EMIF's highest priority task, according to the priority scheme detailed in Section 12.2.13. In the event that the write request cannot be serviced by a single access cycle to the external device, multiple access cycles are performed by the EMIF until the entire request is fulfilled. The details of an asynchronous write operation in normal mode are described in Table 12-22. Also, Figure 12-12 shows an example timing diagram of a basic write operation.

Table 12-22. Asynchronous Write Operation in Normal Mode

Time Interval	Pin Activity in Normal Mode
Turnaround period	Once the write operation becomes the highest priority task for the EMIF, the EMIF waits for the programmed number of turn-around cycles before proceeding to the setup period of the operation. The number of wait cycles is taken directly from the TA field of the asynchronous *n* configuration register (ASYNC_CS*n*_CR). Between each access (write or read) EMIF inserts two cycles of delay even though TA field is programmed as 0. After the EMIF has waited for the turn-around cycles to complete, the EMIF again checks to make sure that the write operation is still the highest priority task. If so, the EMIF proceeds to the setup period of the operation. If the write operation is no longer the highest priority task, EMIF terminates the operation.
Start of the setup period	The following actions occur at the start of the setup period: • The setup, strobe, and hold values are set according to the W_SETUP, W_STROBE, and W_HOLD values in ASYNC_CS*n*_CR. • The address pins EM1A and EM1BA and the data pins EM1Dx become valid. The EM1A and EM1BA pins carry the values described in Section 12.2.6.1. • $\overline{\text{EM1CS}}$[4:2] falls to enable the external device (if not already low from a previous operation).
Strobe period	The following actions occur at the start of the strobe period of a write operation: 1. $\overline{\text{EM1WE}}$ falls 2. The EM1DQM pins become valid as byte enables. The following actions occur on the rising edge of the clock that is concurrent with the end of the strobe period: 1. $\overline{\text{EM1WE}}$ rises 2. The EM1DQM pins deactivate In Figure 12-12, EM1WAIT is inactive. If EM1WAIT is instead activated, the strobe period can be extended by the external device to give it more time to accept the data. Section 12.2.6.6 contains more details on using the EM1WAIT pin.
End of the hold period	At the end of the hold period: • The address pins EM1Ax and EM1BAx become invalid • The data pins become invalid • $\overline{\text{EM1CS}}$[n] (n = 2, 3, or 4) rises (if no more operations are required to complete the current request) The EMIF can be required to issue additional write operations to a device with a small data bus width to complete an entire word access. In this case, the EMIF immediately re-enters the setup period to begin another operation without incurring the turnaround cycle delay. The setup, strobe, and hold values are not updated in this case. If the entire word access has been completed, the EMIF returns to the previous state unless another asynchronous request has been submitted and is currently the highest priority task. If this is the case, the EMIF instead enters directly into the turnaround period for the pending read or write operation.

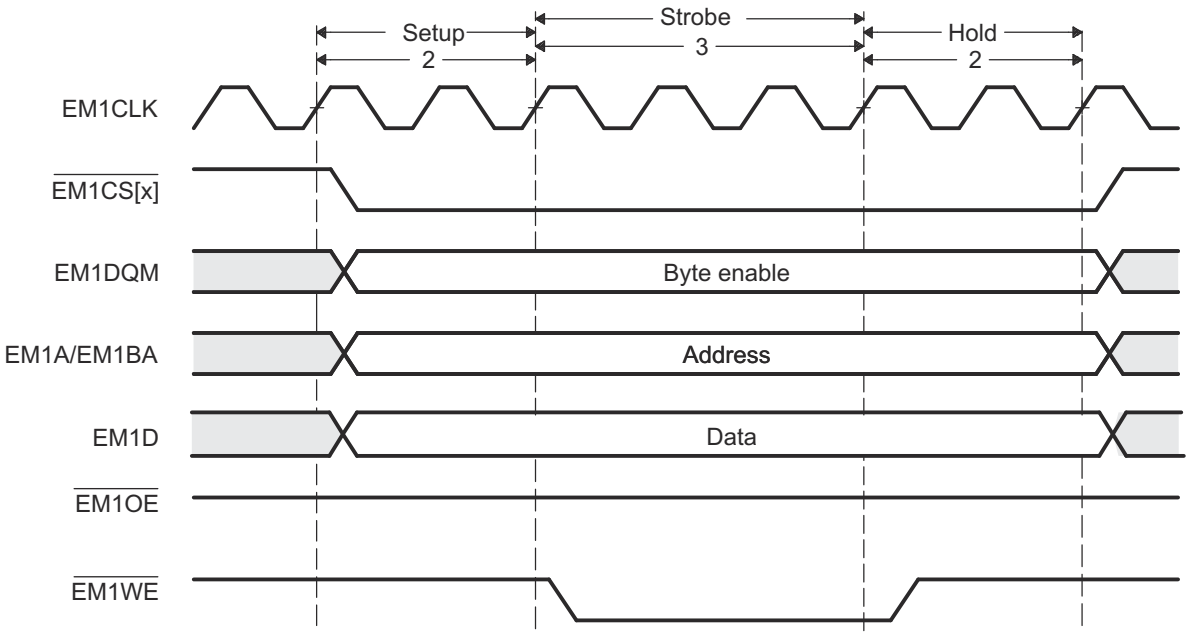


Figure 12-12. Timing Waveform of an Asynchronous Write Cycle in Normal Mode

12.2.6.5 Read and Write Operation in Select Strobe Mode

Select Strobe Mode is the EMIF's second mode of operation. It is selected when the SS bit of the asynchronous *n* configuration register (ASYNC_CSn_CR) is set to 1. In this mode, the EM1DQM pins operate as byte enables and the $\overline{\text{EM1CS}}$[n] (n = 2, 3, or 4) pin is only active during the strobe period of an access cycle. Section 12.2.6.4.1 and Section 12.2.6.4.2 explain the details of read and write operations while in select strobe mode.

12.2.6.5.1 Asynchronous Read Operations (Select Strobe Mode)

Note

During the entirety of an asynchronous read operation, the EM1WEn pin is driven high.

An asynchronous read is performed when any of the requesters mentioned in Section 12.2.2 request a read from the attached asynchronous memory. After the request is received, a read operation is initiated once the request becomes the EMIF's highest priority task, according to the priority scheme detailed in Section 12.2.13. In the event that the read request cannot be serviced by a single access cycle to the external device, multiple access cycles are performed by the EMIF until the entire request is fulfilled. The details of an asynchronous read operation in select strobe mode are described in Table 12-23. Also, Figure 12-13 shows an example timing diagram of a basic read operation.

Table 12-23. Asynchronous Read Operation in Select Strobe Mode

Time Interval	Pin Activity in Select Strobe Mode
Turnaround period	Once the read operation becomes the highest priority task for the EMIF, the EMIF waits for the programmed number of turnaround cycles before proceeding to the setup period of the operation. The number of wait cycles is taken directly from the TA field of the asynchronous *n* configuration register (ASYNC_CSn_CR). Between each access (Write or Read) EMIF inserts two cycles of delay even though TA field is programmed as 0. After the EMIF has waited for the turn-around cycles to complete, the EMIF again checks to make sure that the read operation is still the highest priority task. If so, the EMIF proceeds to the setup period of the operation. If the read operation is no longer the highest priority task, the EMIF terminates the operation.
Start of the setup period	The following actions occur at the start of the setup period: • The setup, strobe, and hold values are set according to the R_SETUP, R_STROBE, and R_HOLD values in ASYNC_CS*n*_CR. • The address pins EM1A and EM1BA become valid and carry the values described in Section 12.2.6.1. • The EM1DQM pins become valid as byte enables.

Strobe period	The following actions occur during the strobe period of a read operation:
	1. $\overline{\text{EM1CS}}$[n] (n = 2, 3, or 4) and $\overline{\text{EM1OE}}$ fall at the start of the strobe period
	2. On the rising edge of the clock that is concurrent with the end of the strobe period:
	  • $\overline{\text{EM1CS}}$[n] (n = 2, 3, or 4) and $\overline{\text{EM10E}}$ rise
	  • The data on the EM1D bus is sampled by EMIF.
	In Figure 12-13, EM1WAIT is inactive. If EM1WAIT is instead activated, the strobe period can be extended by the external device to give it more time to provide the data. Section 12.2.6.6 contains more details on using the EM1WAIT pin.

End of the hold period	At the end of the hold period: • The address pins EM1A and EM1BA become invalid • The EM1DQM pins become invalid The EMIF can be required to issue additional read operations to a device with a small data bus width to complete an entire word access. In this case, the EMIF immediately re-enters the setup period to begin another operation without incurring the turnaround cycle delay. The setup, strobe, and hold values are not updated in this case. If the entire word access has been completed, the EMIF returns to the previous state unless another asynchronous request has been submitted and is currently the highest priority task. If this is the case, the EMIF instead enters directly into the turnaround period for the pending read or write operation.

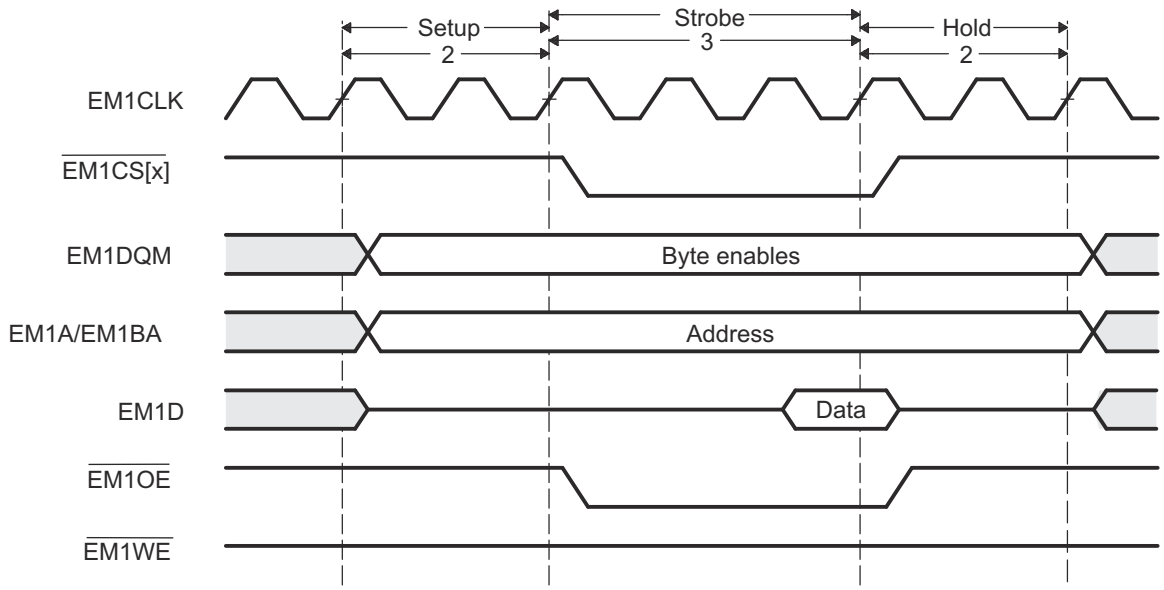


Figure 12-13. Timing Waveform of an Asynchronous Read Cycle in Select Strobe Mode

12.2.6.5.2 Asynchronous Write Operations (Select Strobe Mode)

Note

During the entirety of an asynchronous write operation, the $\overline{\text{EM1OE}}$ pin is driven high.

An asynchronous write is performed when any of the requesters mentioned in Section 12.2.2 request a write to memory in the asynchronous bank of EMIF. After the request is received, a write operation is initiated once the request becomes the EMIF's highest priority task, according to the priority scheme detailed in Section 12.2.13. In the event that the write request cannot be serviced by a single access cycle to the external device, multiple access cycles are performed by the EMIF until the entire request is fulfilled. The details of an asynchronous write operation in select strobe mode are described in Table 12-24. Also, Figure 12-14 shows an example timing diagram of a basic write operation.

Table 12-24. Asynchronous Write Operation in Select Strobe Mode

Time Interval	Pin Activity in Select Strobe Mode
Turnaround period	Once the write operation becomes the highest priority task for the EMIF, the EMIF waits for the programmed number of turnaround cycles before proceeding to the setup period of the operation. The number of wait cycles is taken directly from the TA field of the asynchronous *n* configuration register (ASYNC_CSn_CR). Between each access (Write or Read) EMIF inserts two cycles of delay even though TA field is programmed as 0. After the EMIF has waited for the turnaround cycles to complete, the EMIF again checks to make sure that the write operation is still the highest priority task. If so, the EMIF proceeds to the setup period of the operation. If the write operation is no longer the highest priority task, the EMIF terminates the operation.
Start of the setup period	The following actions occur at the start of the setup period: • The setup, strobe, and hold values are set according to the W_SETUP, W_STROBE, and W_HOLD values in ASYNC_CSn_CR. • The address pins EM1A and EM1BA and the data pins EM1D become valid. The EM1A and EM1BA pins carry the values described in Section 12.2.6.1. • The EM1DQM pins become active as byte enables.
Strobe period	The following actions occur at the start of the strobe period of a write operation: • $\overline{\text{EM1CS}}$[n] (n = 2, 3, or 4) and $\overline{\text{EM1WE}}$ fall The following actions occur on the rising edge of the clock which is concurrent with the end of the strobe period: • $\overline{\text{EM1CS}}$[n] (n = 2, 3, or 4) and $\overline{\text{EM1WE}}$ rise In Figure 12-14, EM1WAIT is inactive. If EM1WAIT is instead activated, the strobe period can be extended by the external device to give it more time to accept the data. Section 12.2.6.6 contains more details on using the EM1WAIT pin.
End of the hold period	At the end of the hold period: • The address pins EM1A and EM1BA become invalid • The data pins become invalid • The EM1DQM pins become invalid The EMIF can be required to issue additional write operations to a device with a small data bus width to complete an entire word access. In this case, the EMIF immediately re-enters the setup period to begin another operation without incurring the turnaround cycle delay. The setup, strobe, and hold values are not updated in this case. If the entire word access has been completed, the EMIF returns to the previous state unless another asynchronous request has been submitted and is currently the highest priority task. If this is the case, the EMIF instead enters directly into the turnaround period for the pending read or write operation.

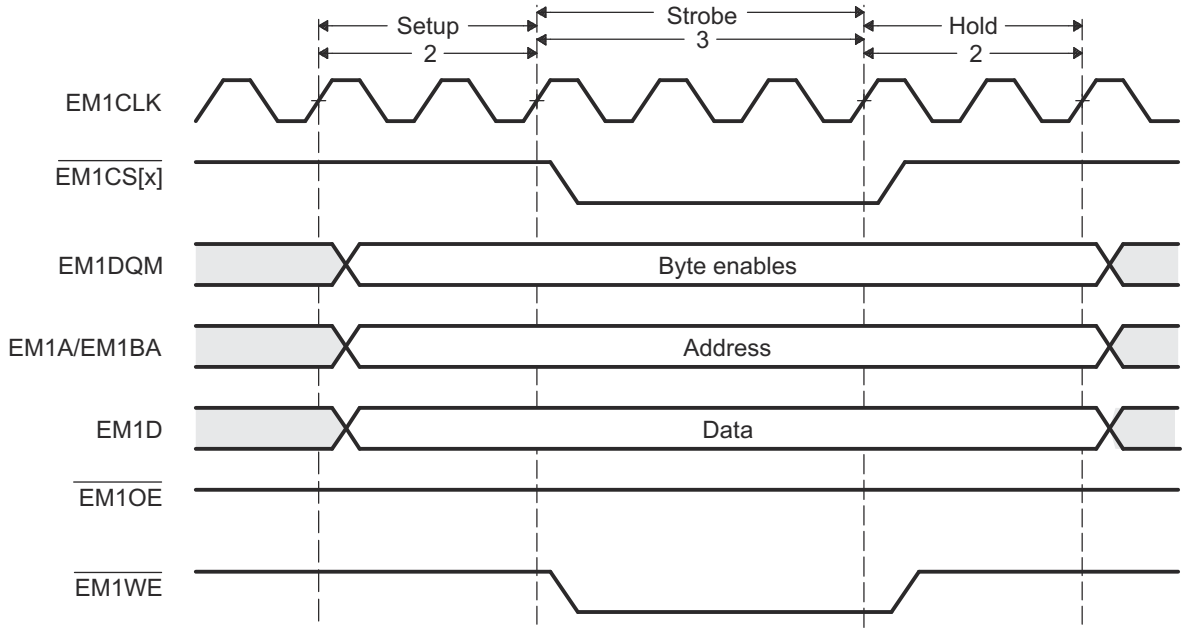


Figure 12-14. Timing Waveform of an Asynchronous Write Cycle in Select Strobe Mode

12.2.6.6 Extended Wait Mode and the EM1WAIT Pin

The EMIF supports the extend wait mode. This is a mode in which the external asynchronous device may assert control over the length of the strobe period. The extended wait mode can be entered by setting the EW bit in the asynchronous *n* configuration register (ASYNC_CS*n*_CR (*n* = 2, 3, or 4). When this bit is set, the EMIF monitors the EM1WAIT pin to determine if the attached device wishes to extend the strobe period of the current access cycle beyond the programmed number of clock cycles.

When the EMIF detects that the EM1WAIT pin has been asserted, it will begin inserting extra strobe cycles into the operation until the EM1WAIT pin is deactivated by the external device. The EMIF will then return to the last cycle of the programmed strobe period and the operation will proceed as usual from this point. Please refer to the device data manual for details on the timing requirements of the EM1WAIT signal.

The EM1WAIT pin cannot be used to extend the strobe period indefinitely. The programmable MAX_EXT_WAIT field in the asynchronous wait cycle configuration register (AWCC) determines the maximum number of EM1CLK cycles the strobe period may be extended beyond the programmed length. When the counter expires, the EMIF proceeds to the hold period of the operation regardless of the state of the EM1WAIT pin. The EMIF can also generate an interrupt upon expiration of this counter. See Section 12.2.9.1 for details on enabling this interrupt.

For the EMIF to function properly in the extended wait mode, the WP*n* bit of AWCC must be programmed to match the polarity of the EM1WAIT pin. In its reset state of 1, the EMIF will insert wait cycles when the EM1WAIT pin is sampled high. When set to 0, the EMIF will insert wait cycles only when EM1WAIT is sampled low. This programmability allows for a glueless connection to larger variety of asynchronous devices.

Finally, a restriction is placed on the strobe period timing parameters when operating in extended wait mode. Specifically, the sum of the W_SETUP and W_STROBE fields must be greater than four, and the sum of the R_SETUP and R_STROBE fields must be greater than four for the EMIF to recognize the EM1WAIT pin has been asserted. The W_SETUP, W_STROBE, R_SETUP, and R_STROBE fields are in ASYNC_CS*n*_CR.

12.2.7 Data Bus Parking

The EMIF always drives the data bus to the previous write data value when the EMIF is idle. This feature is called data bus parking. Only when the EMIF issues a read command to the external memory does the EMIF stop driving the data bus. After the EMIF latches the last read data, the EMIF immediately parks the data bus again.

The one exception to this behavior occurs after performing an asynchronous read operation while the EMIF is in the self-refresh state. In this situation, the read operation is not followed by the EMIF parking the data bus. Instead, the EMIF tri-states the data bus. Therefore, it is not recommended to perform asynchronous read operations while the EMIF is in the self-refresh state, to prevent floating inputs on the data bus. External pull-ups, such as 10-kohm resistors, must be placed on the 16 EMIF data bus pins (that do not have internal pull-ups) if required to perform reads in this situation. The precise resistor value must be chosen so that the worst case combined off-state leakage currents do not cause the voltage levels on the associated pins to drop below the high-level input voltage requirement.

For information about the self-refresh state, see Section 12.2.5.7.

12.2.8 Reset and Initialization Considerations

The EMIF memory controller has two active-low reset signals, CHIP_RST_n and MOD_G_RST_n. Both these reset signals are driven by the device system reset signal. This device does not offer the flexibility to reset just the EMIF state machine without also resetting the EMIF controller's memory-mapped registers. As soon as the device system reset is released (driven high), the EMIF memory controller immediately begins its initialization sequence. Command and data stored in the EMIF memory controller FIFOs are lost. Refer to Section 12.2 for more information on conditions that can cause a device system reset to be asserted.

When system reset is released, the EMIF automatically begins running the SDRAM initialization sequence described in Section 12.2.5.4. Even though the initialization procedure is automatic, a special procedure, found in Section 12.2.5.5 must still be followed.

12.2.9 Interrupt Support

The EMIF supports a single interrupt to the CPU. Section 12.2.9.1 details the generation and internal masking of EMIF interrupts.

12.2.9.1 Interrupt Events

There are three conditions that can cause the EMIF to generate an interrupt to the CPU. These conditions are:

- A rising edge on the EM1WAIT signal (wait rise interrupt)
- An asynchronous time out
- Usage of unsupported addressing mode (line trap interrupt)

The wait rise interrupt occurs when a rising edge is detected on EM1WAIT signal. This interrupt generation is not affected by the WP*n* bit in the asynchronous wait cycle configuration register (ASYNC_WCCR). The asynchronous time out interrupt condition occurs when the attached asynchronous device fails to deassert the EM1WAIT pin within the number of cycles defined by the MAX_EXT_WAIT bit in AWCC (this happens only in extended wait mode). The EMIF supports only linear incrementing and cache line wrap addressing modes . If an access request for an unsupported addressing mode is received, the EMIF sets the LT bit in the EMIF interrupt raw register (INT_RAW) and treats the request as a linear incrementing request.

Only when the interrupt is enabled by setting the appropriate bit (WR_MASK_SET/AT_MASK_SET/ LT_MASK_SET) in the EMIF interrupt mask set register (INT_MSK_SET) to 1, is the interrupt sent to the CPU. Once enabled, the interrupt can be disabled by writing a 1 to the corresponding bit in the EMIF interrupt mask clear register (INT_MSK_CLR). The bit fields in both the INT_MSK_SET and INT_MSK_CLR can be used to indicate whether the interrupt is enabled. When the interrupt is enabled, the corresponding bit field in both the INT_MSK_SET and INT_MSK_CLR have a value of 1; when the interrupt is disabled, the corresponding bit field has a value of 0.

The EMIF interrupt raw register (INT_RAW) and the IF interrupt mask register (INT_MSK) indicate the status of each interrupt. The appropriate bit (WR/AT/LT) in INT_RAW is set when the interrupt condition occurs, whether or not the interrupt has been enabled. However, the appropriate bit (WR_MASKED/AT_MASKED/LT_MASKED) in INT_MSK is set only when the interrupt condition occurs and the interrupt is enabled. Writing a 1 to the bit in INT_RAW clears the INT_RAW bit as well as the corresponding bit in INT_MSK. Table 12-25 contains a brief summary of the interrupt status and control bit fields. See Section 12.5 for complete details on the register fields.

Table 12-25. Interrupt Monitor and Control Bit Fields

Register Name	Bit Name	Description
EMIF interrupt raw register (INT_RAW)	WR	This bit is set when an rising edge on the EM1WAIT signal occurs. Writing a 1 clears the WR bit as well as the WR_MASKED bit in INT_MSK.
	AT	This bit is set when an asynchronous timeout occurs. Writing a 1 clears the AT bit as well as the AT_MASKED bit in INT_MSK.
	LT	This bit is set when an unsupported addressing mode is used. Writing a 1 clears LT bit as well as the LT_MASKED bit in INT_MSK.
EMIF interrupt mask register (INT_MSK)	WR_MASKED	This bit is set only when a rising edge on the EM1WAIT signal occurs and the interrupt has been enabled by writing a 1 to the WR_MASK_SET bit in INT_MSK_SET.
	AT_MASKED	This bit is set only when an asynchronous timeout occurs and the interrupt has been enabled by writing a 1 to the AT_MASK_SET bit in INT_MSK_SET.
	LT_MASKED	This bit is set only when line trap interrupt occurs and the interrupt has been enabled by writing a 1 to the LT_MASK_SET bit in INT_MSK_SET.
EMIF interrupt mask set register (INT_MSK_SET)	WR_MASK_SET	Writing a 1 to this bit enables the wait rise interrupt.
	AT_MASK_SET	Writing a 1 to this bit enables the asynchronous timeout interrupt.
	LT_MASK_SET	Writing a 1 to this bit enables the line trap interrupt.
EMIF interrupt mask clear register (INT_MSK_CLR)	WR_MASK_CLR	Writing a 1 to this bit disables the wait rise interrupt.
	AT_MASK_CLR	Writing a 1 to this bit disables the asynchronous timeout interrupt.
	LT_MASK_CLR	Writing a 1 to this bit disables the line trap interrupt.

12.2.10 DMA Event Support

The EMIF memory controller is a DMA slave peripheral and therefore does not generate DMA events. Data read and write requests may be made directly, by masters and the DMA.

12.2.11 EMIF Signal Multiplexing

For details on the EMIF signal multiplexing, see the *GPIO* chapter, I/O Multiplexing Module section, of this technical reference manual.

12.2.12 Memory Map

For information describing the device memory-map, see your device-specific data manual.

12.2.13 Priority and Arbitration

Section 12.2.2 describes the external prioritization and arbitration among requests from different sources within the microcontroller. The result of this external arbitration is that only one request is presented to the EMIF at a time. Once the EMIF completes a request, the external arbiter then provides the EMIF with the next pending request.

Internally, the EMIF undertakes memory device transactions according to a strict priority scheme. The highest priority events are:

- A device reset.
- A write to any of the three least significant bytes of the SDRAM configuration register (SDRAM_CR).

Either of these events will cause the EMIF to immediately commence its initialization sequence as described in Section 12.2.5.4.

Once the EMIF has completed its initialization sequence, it performs memory transactions according to the following priority scheme (highest priority listed first):

1. If the EMIF's backlog refresh counter is at the Refresh Must urgency level, the EMIF performs multiple SDRAM auto-refresh cycles until the Refresh Release urgency level is reached.
2. If an SDRAM or asynchronous read has been requested, the EMIF performs a read operation.
3. If the EMIF's backlog refresh counter is at the Refresh Need urgency level, the EMIF performs an SDRAM auto refresh cycle.
4. If an SDRAM or asynchronous write has been requested, the EMIF performs a write operation.
5. If the EMIF's backlog refresh counter is at the Refresh May or Refresh Release urgency level, the EMIF performs an SDRAM auto refresh cycle.
6. If the value of the SR bit in SDRAM_CR has been set to 1, the EMIF will enter the self-refresh state as described in Section 12.2.5.7.

After taking one of the actions listed above, the EMIF then returns to the top of the priority list to determine its next action.

Because the EMIF does not issue auto-refresh cycles when in the self-refresh state, the above priority scheme does not apply when in this state. See Section 12.2.5.7 for details on the operation of the EMIF when in the self-refresh state.

12.2.14 System Considerations

This section describes various system considerations to keep in mind when operating the EMIF.

12.2.14.1 Asynchronous Request Times

In a system that interfaces to both SDRAM and asynchronous memory, the asynchronous requests must not take longer than the smaller of the following two values:

- t_{RAS} (typically 120 µs) - to avoid violating the maximum time allowed between issuing an ACTV and PRE command to the SDRAM.
- $t_{Refresh\ Rate}$ × 11 (typically 15.7 µs × 11 = 172.7 µs) - to avoid refresh violations on the SDRAM.

The length of an asynchronous request is controlled by multiple factors, the primary factor being the number of access cycles required to complete the request. For example, an asynchronous request for 4 bytes will require four access cycles using an 8-bit data bus and only two access cycle using a 16-bit data bus. The maximum request size that the EMIF can be sent is 16 words, therefore the maximum number of access cycles per memory request is 64 when the EMIF is configured with an 8-bit data bus. The length of the individual access cycles that make up the asynchronous request is determined by the programmed setup, strobe, hold, and turnaround values, but can also be extended with the assertion of the EM1WAIT input signal up to a programmed maximum limit. It is up to the user to make sure that an entire asynchronous request does not exceed the timing values listed above when also interfacing to an SDRAM device. This can be done by limiting the asynchronous timing parameters.

12.2.15 Power Management

Power dissipation from the EMIF memory controller may be managed by following methods:

- Self-refresh mode
- Power-down mode
- Gating input clocks to the module off

Gating input clocks off to the EMIF memory controller achieves higher power savings when compared to the power savings of self-refresh or power down mode. The input clock to EMIF can be turned off through the use of the Global Clock Module (GCM). Before gating clocks off, the EMIF memory controller must place the SDR SDRAM memory in self-refresh mode. If the external memory requires a continuous clock, the VCLK3 clock domain must not be turned off because this may result in data corruption. See the following subsections for the proper procedures to follow when stopping EMIF memory controller clocks.

12.2.15.1 Power Management Using Self-Refresh Mode

The EMIF memory controller can be placed into a self-refresh state in order to place the attached SDRAM devices into self-refresh mode, which consumes less power for most SDRAM devices. In this state, the attached SDRAM device uses an internal clock to perform its own auto refresh cycles. This maintains the validity of the data in the SDRAM without the need for any external commands. Refer to Section 12.2.5.7 for more details on placing the EMIF into the self-refresh state.

12.2.15.2 Power Management Using Power Down Mode

In the power down mode, the EMIF drives EM1SDCKE low to lower the power consumption. EM1SDCKE goes high when there is a need to send refresh (REFR) commands, after which EM1SDCKE is again driven low. EM1SDCKE remains low until any request arrives. Refer to Section 12.2.5.8 for more details on placing the EMIF in power-down mode.

12.2.16 Emulation Considerations

The EMIF remains fully functional during emulation halts in order to allow emulation access to external memory.

12.3 Example Configuration

This section presents an example of interfacing the EMIF1 to both an SDR SDRAM device and an asynchronous Flash device.

12.3.1 Hardware Interface

Figure 12-15 shows the hardware interface between the EMIF, a Samsung K4S641632H-TC(L)70 64Mb SDRAM device, and a SHARP LH28F800BJE-PTTL90 8Mb Flash memory. The connection between EMIF and the SDRAM is straightforward, but the connection between the EMIF and the Flash deserves a detailed look.

The address inputs for the Flash are provided by three sources. The A[18:0] address inputs are provided by a combination of the EM1A and EM1BA pins according to Section 12.2.6.1, and a set of GPIO pins. The RD/nBY signal from Flash is connected to EM1WAIT pin of the EMIF.

Finally, this example configuration connects the $\overline{\text{EM1WE}}$ pin to the nWE input of the Flash and operates the EMIF in select strobe mode.

12.3.2 Software Configuration

The following sections describe how to configure the EMIF registers and bit fields to interface the EMIF with the Samsung K4S641632H-TC(L)70 SDRAM and the SHARP LH28F800BJE-PTTL90 8Mb Flash memory.

12.3.2.1 Configuring the SDRAM Interface

This section describes how to configure the EMIF to interface with the Samsung K4S641632H-TC(L)70 SDRAM with a clock frequency of f_{EM1CLK} = 100 MHz. Procedure A described in Section 12.2.5.5 is followed that assumes that the SDRAM power-up timing constraints were met during the SDRAM auto-initialization sequence after reset.

12.3.2.1.1 PLL Programming for EMIF to K4S641632H-TC(L)70 Interface

If the system PLL is programmed to provide a SYSCLK frequency > 100 MHz, then configure the EMIF1CLKDIV field in the PERSYSCLKDIVSEL register to make EM1CLK = SYSCLK/2 (default configuration). Before doing this, the SDRAM must be placed in self-refresh mode by setting the SR bit in the SDRAM configuration register. Once the EM1CLK frequency has been configured, remove the SDRAM from self-refresh by clearing the SR bit in SDRAM_CR.

Table 12-26. SR Field Value For EMIF to K4S641632H-TC(L)70 Interface

Field	Value	Purpose
SR	1 then 0	To place the EMIF into the self-refresh state

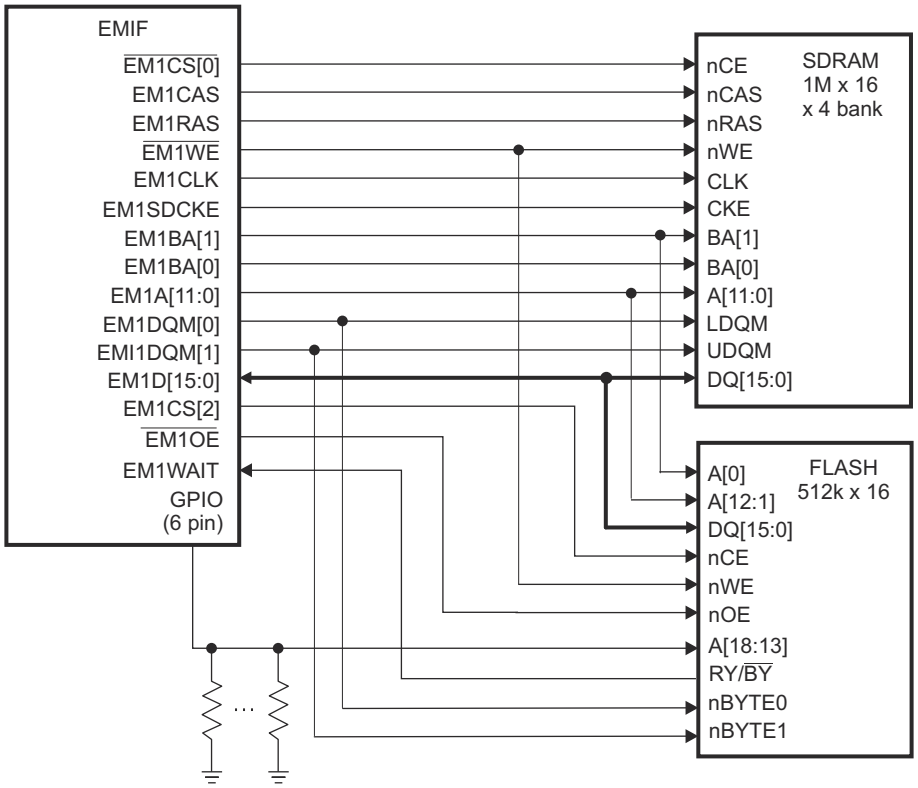


Figure 12-15. Example Configuration Interface

12.3.2.1.2 SDRAM Timing Register (SDRAM_TR) Settings for EMIF to K4S641632H-TC(L)70 Interface

The fields of the SDRAM timing register (SDRAM_TR) must be programmed first as described in Table 12-27 to satisfy the required timing parameters for the K4S641632H-TC(L)70. Based on these calculations, a value of 6111 4610h must be written to the SDRAM_TR. Figure 12-16 shows a graphical description of how the SDRAM_TR must be programmed.

Table 12-27. SDRAM_TR Field Calculations for EMIF to K4S641632H-TC(L)70 Interface

Field Name	Formula	Value from K4S641632H-TC(L)70 Data Sheet	Value Calculated for Field
T_RFC	T_RFC >= (t_{RFC} × f_{EM1CLK}) - 1	t_{RC} = 68 ns (min)(1)	6
T_RP	T_RP >= (t_{RP} × f_{EM1CLK}) - 1	t_{RP} = 20 ns (min)	1
T_RCD	T_RCD >= (t_{RCD} × f_{EM1CLK}) - 1	t_{RCD} = 20 ns (min)	1
T_WR	T_WR >= (t_{WR} × f_{EM1CLK}) - 1	t_{RDL} = 2 CLK = 20 ns (min)(2)	1
T_RAS	T_RAS >= (t_{RAS} × f_{EM1CLK}) - 1	t_{RAS} = 49 ns (min)	4
T_RC	T_RC >= (t_{RC} × f_{EM1CLK}) - 1	t_{RC} = 68 ns (min)	6
T_RRD	T_RRD >= (t_{RRD} × f_{EM1CLK}) - 1	t_{RRD} = 14 ns (min)	1

(1) The Samsung data sheet does not specify a t_{RFC} value. Instead, Samsung specifies t_{RC} as the minimum auto refresh period.
(2) The Samsung data sheet does not specify a t_{WR} value. Instead, Samsung specifies t_{RDL} as last data in to row precharge minimum delay.

Figure 12-16. SDRAM Timing Register (SDRAM_TR)

31	30	29	28	27	26	24	23	22	21	20	19	18	17	16
0 0110					001		0	001			0	001		
T_RFC					T_RP		Rsvd	T_RCD			Rsvd	T_WR		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0100				0110				0	001			0000			
T_RAS				T_RC				Rsvd	T_RRD			Reserved			

12.3.2.1.3 SDRAM Self Refresh Exit Timing Register (SDR_EXT_TMNG) Settings for EMIF to K4S641632H-TC(L)70 Interface

The SDRAM self-refresh exit timing register (SDSRETR) must be programmed second to satisfy the t_{XSR} timing requirement from the K4S641632H-TC(L)70 data sheet. Table 12-28 shows the calculation of the proper value to program into the T_XS field of this register. Based on this calculation, a value of 6h must be written to the SDSRETR. Figure 12-17 shows how the SDSRETR must be programmed.

Table 12-28. RR Calculation for EMIF to K4S641632H-TC(L)70 Interface

Field Name	Formula	Value from K4S641632H-TC(L)70 Data Sheet	Value Calculated for Field
T_XS	T_XS >= (t_{XSR} × f_{EM1CLK}) - 1	t_{RC} = 68 ns (min)(1)	6

(1) The Samsung data sheet does not specify a t_{XSR} value. Instead, Samsung specifies t_{RC} as the minimum required time after CKE going high to complete self-refresh exit.

Figure 12-17. SDRAM Self Refresh Exit Timing Register (SDR_EXT_TMNG)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0000 0000 0000 0000															
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
000 0000 0000													0 0110		
Reserved													T_XS		

12.3.2.1.4 SDRAM Refresh Control Register (SDRAM_RCR) Settings for EMIF to K4S641632H-TC(L)70 Interface

The SDRAM refresh control register (SDRAM_RCR) can next be programmed to satisfy the required refresh rate of the K4S641632H-TC(L)70. Table 12-29 shows the calculation of the proper value to program into the RR field of this register. Based on this calculation, a value of 61Ah must be written to the SDRAM_RCR. Figure 12-18 shows how the SDRAM_RCR must be programmed.

Table 12-29. RR Calculation for EMIF to K4S641632H-TC(L)70 Interface

Field Name	Formula	Values	Value Calculated for Field
RR	RR ≤ f_{EM1CLK} × $t_{Refresh\ Period}$ / n_{cycles}	From SDRAM data sheet: $t_{Refresh\ Period}$ = 64 ms; n_{cycles} = 4096 EMIF clock rate: f_{EM1CLK} = 100 MHz	RR = 1562 cycles = 61Ah cycles

Figure 12-18. SDRAM Refresh Control Register (SDRAM_RCR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0 0000 0000 0000														000	
Reserved														Reserved	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
000			0 0110 0001 1010 (61Ah)												
Reserved			RR												

12.3.2.1.5 SDRAM Configuration Register (SDRAM_CR) Settings for EMIF to K4S641632H-TC(L)70 Interface

Finally, the fields of the SDRAM configuration register (SDRAM_CR) must be programmed as described in Table 12-30 to properly interface with the K4S641632H-TC(L)70 device. Based on these settings, a value of 4720h must be written to the SDRAM_CR. Figure 12-19 shows how the SDRAM_CR must be programmed. The EMIF is now ready to perform read and write accesses to the SDRAM.

Table 12-30. SDRAM_CR Field Values For EMIF to K4S641632H-TC(L)70 Interface

Field	Value	Purpose
SR	0	To avoid placing the EMIF into the self-refresh state
NM	1	To configure the EMIF for a 16-bit data bus
CL	011b	To select a CAS latency of 3
BIT11_9LOCK	1	To allow the CL field to be written
IBANK	010b	To select 4 internal SDRAM banks
PAGESIZE	0	To select a page size of 256 words

Figure 12-19. SDRAM Configuration Register (SDRAM_CR)

31	30	29	28	27	26	25	24
0	0	0	0 0000				
SR	Reserved	Reserved	Reserved				

23	22	21	20	19	18	17	16
00 0000						0	0
Reserved						Reserved	Reserved

15	14	13	12	11	10	9	8
0	1	0	0	011			1
Reserved	NM	Reserved	Reserved	CL			BIT11_9LOCK

7	6	5	4	3	2	1	0
0	010			0	000		
Reserved	IBANK			Reserved	PAGESIZE		

12.3.2.2 Configuring the Flash Interface

This section describes how to configure the EMIF to interface with the SHARP LH28F800BJE-PTTL90 8Mb Flash memory with a clock frequency of f_{EM1CLK} = 100 MHz.

12.3.2.2.1 Asynchronous 1 Configuration Register (ASYNC_CS2_CFG) Settings for EMIF to LH28F800BJE-PTTL90 Interface

The asynchronous 1 configuration register (ASYNC_CS2_CFG) is the only register that is necessary to program for this asynchronous interface. The SS bit must be set to 1 to enable select strobe command and the ASIZE field can be set to 1 to select a 16-bit interface. The other fields in this register control the shaping of the EMIF signals, and the proper values can be determined by referring to the AC Characteristics in the Flash data sheet and the device data sheet. Table 12-31 and Table 12-32 show the pertinent AC Characteristics for reads and writes to the Flash device, and Figure 12-20 and Figure 12-21 show the associated timing waveforms. Finally, Figure 12-22 shows programming the ASYNC_CS2_CFG with the calculated values.

Table 12-31. AC Characteristics for a Read Access

AC Characteristic	Device	Definition	Minimum	Maximum	Unit
t_{SU}	EMIF	Setup time, read EM1D before EM1OE high	15		ns
t_{H}	EMIF	Data hold time, read EM1D after EM1OE high	0		ns
t_{ELQV}	Flash	nCE to Output Delay		90	ns
t_{EHQZ}	Flash	nCE High to Output in High Impedance		55	ns

Table 12-32. AC Characteristics for a Write Access

AC Characteristic	Device	Definition	Minimum	Maximum	Unit
t_{AVAV}	Flash	Write Cycle Time	90		ns
t_{ELEH}	Flash	nCE Pulse Width Low	50		ns
t_{EHEL}	Flash	nCE Pulse Width High (not shown in Figure 12-21)	30		ns

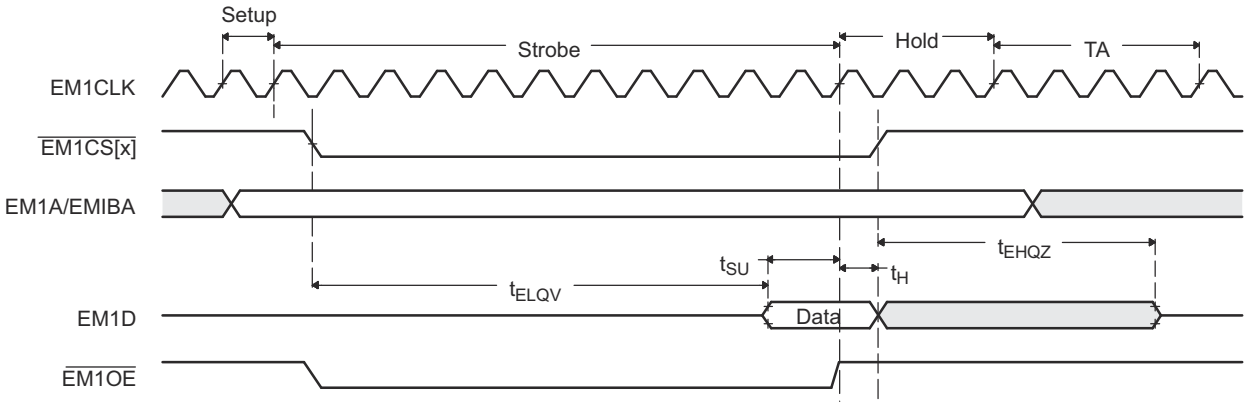


Figure 12-20. LH28F800BJE-PTTL90 to EMIF Read Timing Waveforms

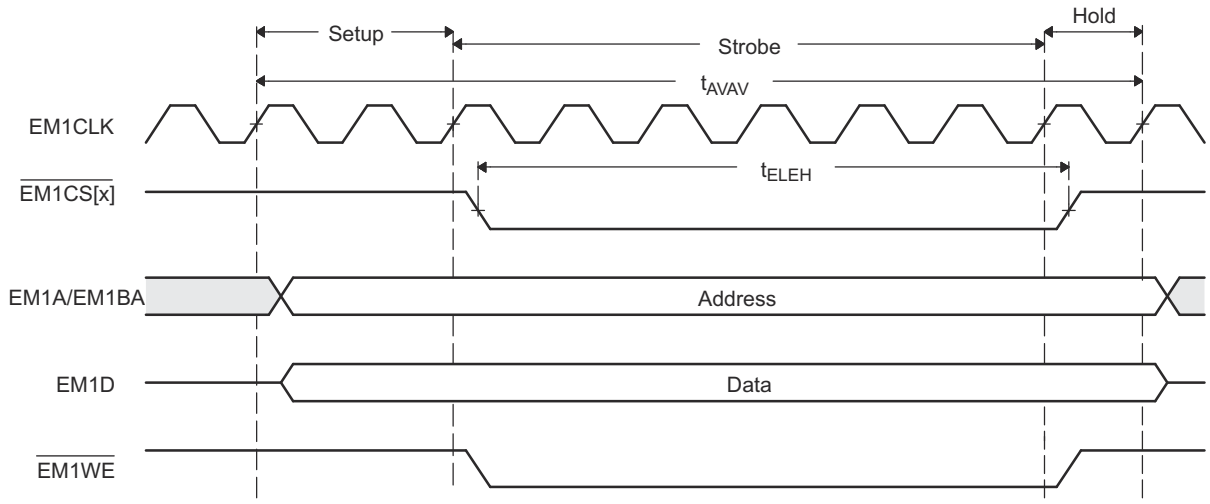


Figure 12-21. LH28F800BJE-PTTL90 to EMIF Write Timing Waveforms

The R_STROBE field must be set to meet the following equation:

$R\_STROBE >= (t_{ELQV} + t_{SU}) \times f_{EM1CLK} - 1$

R_STROBE >= (90 ns + 15 ns) × 200 MHz - 1

R_STROBE >= 20

R_STROBE = 20

The R_HOLD field must be large enough to satisfy EMIF Data hold time, t_H:

$R\_HOLD > = t_H \times f_{EM1CLK} - 1$

R_HOLD >= 0 ns × 200 MHz - 1

R_HOLD >= -1

The R_HOLD field must also combine with the TA field to satisfy the Flash's nCE High to Output in High Impedance time, t_{EHQZ}:

$R\_HOLD + TA >= t_{EHQZ} \times f_{EM1CLK} - 2$

R_HOLD + TA >= 55 ns × 200 MHz - 2

R_HOLD + TA >= 9

The largest value that can be programmed into the TA field is 3h, therefore the following values must be used:

R_HOLD = 6

TA = 3

For Writes, the W_STROBE field must be set to satisfy the Flash's nCE Pulse Width constraint, t_{ELEH}:

$W\_STROBE >= t_{ELEH} \times f_{EM1CLK} - 1$

W_STROBE >= 50 ns × 200 MHz - 1

W_STROBE >= 9

The W_SETUP and W_HOLD fields must combine to satisfy the Flash's nCE Pulse Width High constraint, t_{EHEL}:

W_SETUP + W_HOLD > = t_{EHEL} × f_{EM1CLK} - 2

W_SETUP + W_HOLD > = 30 ns × 200 MHz - 2

W_SETUP + W_HOLD > = 4

In addition, the entire Write access length must satisfy the Flash minimum Write Cycle Time, t_{AVAV}:

W_SETUP + W_STROBE + W_HOLD >= t_{AVAV} × f_{EM1CLK} - 3

W_SETUP + W_STROBE + W_HOLD >= 90 ns × 200 MHz - 3

W_SETUP + W_STROBE + W_HOLD >= 15

Solving the above equations for the Write fields results in the following:

W_SETUP = 4

W_STROBE = 10

W_HOLD = 1

Adding a 5 ns (1 cycle) margin to each of the periods (excluding TA that is already at the maximum) in this example produces the following recommended values:

W_SETUP = 5h

W_STROBE = Bh

W_HOLD = 2h

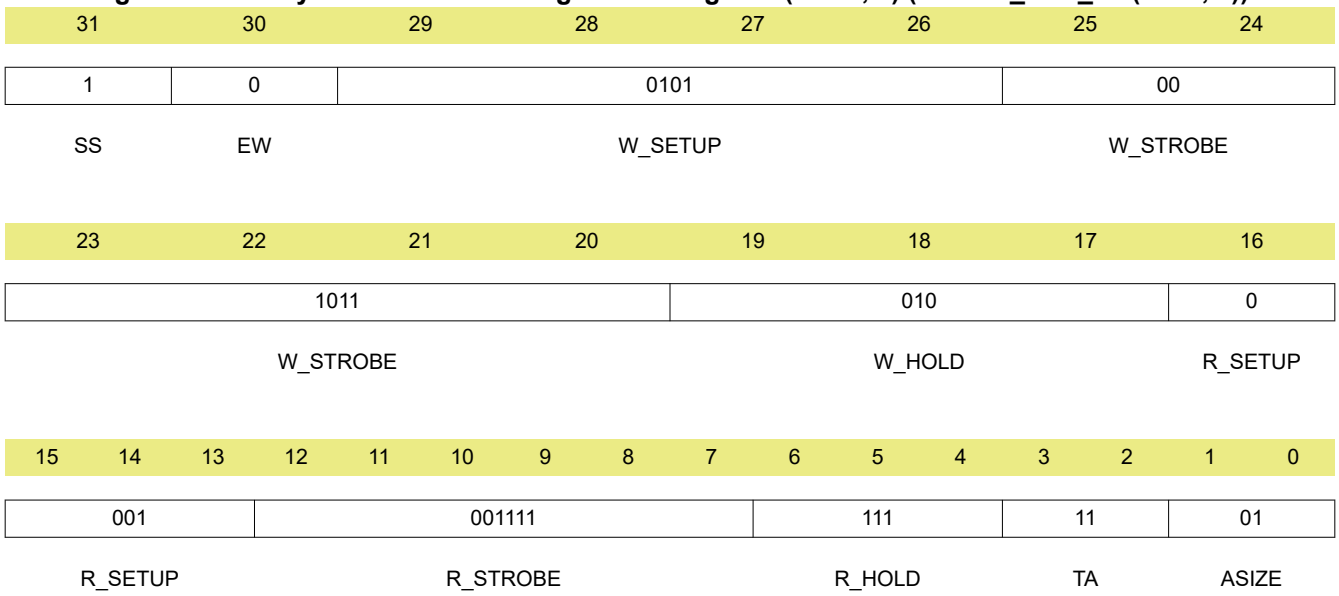
R_SETUP = 1h

R_STROBE = 15h

R_HOLD = 7h

TA = 3h

Figure 12-22. Asynchronous *m* Configuration Register (*m* = 1, 2) (ASYNC_CS*n*_CR(*n* = 2, 3))



12.4 Software

12.4.1 EMIF Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/emif

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

12.4.1.1 Pin setup for EMIF module accessing ASRAM.

FILE: emif_asram_pin_setup.c

This example configures pins for EMIF in ASYNC mode.

12.4.1.2 EMIF1 ASYNC module accessing 16bit ASRAM.

FILE: emif_ex1_16bit_asram.c

This example configures EMIF1 in 16bit ASYNC mode and uses CS2 as chip enable.

External Connections

- External ASRAM memory (CY7C1041CV33 -10ZSXA) daughter card

Watch Variables

- *testStatusGlobal* - Equivalent to *TEST_PASS* if test finished correctly, else the value is set to *TEST_FAIL*
- *errCountGlobal* - Error counter

12.4.1.3 EMIF1 ASYNC module accessing 16bit ASRAM through CPU1 and CPU2. - C28X_DUAL

FILE: emif_ex1_16bit_asram_dual_access_cpu1.c

This example configures EMIF1 in 16bit ASYNC mode and uses CS2 as chip enable. The EMIF1 ownership is passed between CPU1 and CPU2 to access different memory regions. Initially CPU2 grabs and configures the EMIF1, thereafter both CPU1 and CPU2 grabs EMIF1 to access different memory regions in external memory.

External Connections

- External ASRAM memory (CY7C1041CV33 -10ZSXA) daughter card

Watch Variables

- *testStatusGlobalCPU1* - Equivalent to *TEST_PASS* if test finished correctly, else the value is set to *TEST_FAIL*
- *errCountGlobalCPU1* - Error counter

12.4.1.4 EMIF1 ASYNC module accessing 16bit ASRAM trhough CPU1 and CPU2. - C28X_DUAL

FILE: emif_ex1_16bit_asram_dual_access_cpu2.c

This example configures EMIF1 in 16bit ASYNC mode and uses CS2 as chip enable. The EMIF1 ownership is passed between CPU1 and CPU2 to access different memory regions. Initially CPU2 grabs and configures the EMIF1, thereafter both CPU1 and CPU2 grabs EMIF1 to access different memory regions in external memory.

External Connections

- External ASRAM memory (CY7C1041CV33 -10ZSXA) daughter card

Watch Variables

- *testStatusGlobalCPU2* - Equivalent to *TEST_PASS* if test finished correctly, else the value is set to *TEST_FAIL*
- *errCountGlobalCPU2* - Error counter

12.4.1.5 EMIF1 module accessing 16bit ASRAM as code memory.

FILE: emif_ex2_16bit_asram_codemem.c

This example configures EMIF1 in 16bit ASYNC mode and uses CS2 as chip enable. This example enables use of ASRAM as code memory.

External Connections

- External ASRAM memory (CY7C1041CV33 -10ZSXA) daughter card

Watch Variables

- *testStatusGlobal* - Equivalent to *TEST_PASS* if test finished correctly, else the value is set to *TEST_FAIL*
- *errCountGlobal* - Error counter

12.4.1.6 EMIF1 module accessing 16bit SDRAM using memcpy_fast_far().

FILE: emif_ex3_16bit_sdram_far.c

This example configures EMIF1 in 16bit SYNC mode and uses CS0 as chip enable.It will first write to an array in the SDRAM and then read it back using the FPU function, memcpy_fast_far(), for both operations.

The buffer in SDRAM will be placed in the .farbss memory on account of the fact that its assigned the attribute "far" indicating it lies beyond the 22-bit program address space. The compiler will take care to avoid using instructions such as PREAD, which uses the Program Read Bus, or addressing modes restricted to the lower 22-bit space when accessing data with the attribute "far".

The memory space beyond 22-bits must be treated as data space for load/store operations only. The user is cautioned against using this space for either instructions or working memory. *External Connections*

- External SDR-SDRAM memory (MT48LC32M16A2 -75) daughter card

Watch Variables

- *testStatusGlobal* - Equivalent to *TEST_PASS* if test finished correctly, else the value is set to *TEST_FAIL*
- *errCountGlobal* - Error counter

12.4.1.7 EMIF1 module accessing 16bit SDRAM then puts into Self Refresh mode before entering Low Power Mode.

FILE: emif_ex4_16bit_sdram_far_lpm.c

This example configures EMIF1 in 16bit SYNC mode and uses CS0 as chip enable. This example puts SDRAM into self refresh before entering standby mode. Watchdog timer is configured to trigger WAKEINT interrupt.

As soon as the watchdog timer expires, the device should wake up, SDRAM should come out of self refresh mode and GPIO11 can be observed to toggle.

External Connections

- External SDR-SDRAM memory (MT48LC32M16A2 -75) daughter card

Watch Variables

- *testStatusGlobal* - Equivalent to *TEST_PASS* if test finished correctly, else the value is set to *TEST_FAIL*
- *errCountGlobal* - Error counter

12.4.1.8 EMIF1 module accessing 32bit SDRAM using DMA.

FILE: emif_ex5_16bit_sdram_dma.c

This example configures EMIF1 in 16bit SYNC(SDRAM) mode and uses CS0 as chip enable. It will first write to an array in the SDRAM and then read it back, using the DMA for both operations.

The buffer in SDRAM will be placed in the .farbss memory on account of the fact that its assigned the attribute "far" indicating it lies beyond the 22-bit program address space. The compiler will take care to avoid using instructions such as PREAD, which uses the Program Read Bus, or addressing modes restricted to the lower 22-bit space when accessing data with the attribute "far".

The memory space beyond 22-bits must be treated as data space for load/store operations only. The user is cautioned against using this space for either instructions or working memory. *External Connections*

- External SDR-SDRAM (Micron MT48LC32M16A2 "P -75 C") daughter card.

Watch Variables

- *testStatusGlobal* - Equivalent to *TEST_PASS* if test finished correctly, else the value is set to *TEST_FAIL*
- *errCountGlobal* - Error counter

12.4.1.9 EMIF1 module accessing 16bit SDRAM using alternate address mapping.

FILE: emif_ex6_16bit_sdram_nonfar.c

This example configures EMIF1 in 16bit SYNC mode and uses CS0 as chip enable.It will first write to an array in the SDRAM and then read it back.

The buffer in SDRAM will be placed in the emif_cs0_nonfar memory section which is dual mapped with CS2 memory range. This has been done to keep the SDRAM memory range within 22-bit address range in order to generate optimal code. EMIF1 Async RAM accesses will not be issued at the same time and program space reads & fetches will be allowed to SDRAM in non-far range.

External Connections

- External SDR-SDRAM memory (MT48LC32M16A2 -75) daughter card

Watch Variables

- *testStatusGlobal* - Equivalent to *TEST_PASS* if test finished correctly, else the value is set to *TEST_FAIL*
- *errCountGlobal* - Error counter

12.4.1.10 EMIF1 ASYNC module accessing 16bit ASRAM HIC FSI

FILE: emif_ex7_16bit_asram_hic_fsi.c

This example configures EMIF1 in 16bit ASYNC mode and uses CS2 as chip enable. This can be run with hic_ex3_config_16bit_fsi example on F28002x device. This is run first followed by the run of F28002x device example. This example configures EMIF1 in 16-bit ASYNC mode and uses CS2 as chip enable to access Host Interface Controller(HIC) on F28002X device

This follows example configuration for performance critical applications described in the Application note titled "Application guide for peripheral expansion using HIC"(SPRACR2).

-This example sets up the EMIF CS2 for ASRAM interface to access the Host interface Controller. This uses the Direct Access Mode of the HIC. -The example on F28002x side sets up the FSI module for internal loopback Sets up the Host interface Controller for direct access mode

- Sends a HIC_INIT_DONE_TOKEN after which this example accesses the FSI of the device side over HIC direct access
- Fills the Transmit frame and triggers transmit
- Receives an interrupt when the frame is received on the device side FSI(uses GPIO04 connected to HIC_INT, configured for XINT1)
- reads the FSI received frame and checks for correctness This demonstrates the usage of Direct access mode and 16 bit mode of HIC module. *External Connections*
  - This example will not work on F2838x Control Card and has been tested in TI Internal Validation platform.

Watch Variables

- *errCountGlobal* - Error counter
- *xint1Count* - Number of times HIC Interrupt is received for FSI Receive event

12.4.1.11 EMIF1 ASYNC module accessing 8bit HIC controller.

FILE: emif_ex8_8bit_asram_hic_adc.c

This can be run with hic_ex2_config_8bit example on F28002x device. This is run first followed by the run of F28002x device. This example configures EMIF1 in 8 bit ASYNC mode and uses CS2 as chip enable to access Host Interface controller on F28002X device It uses Select Strobe mode of EMIF Controller.

This follows example configuration for pin constrained applications described in the Application note titled Application guide for peripheral expansion using HIC(SPRACR2).

- This uses 8 bit ASRAM with control signals as explained in the note. uses the Mailbox access mode of Host interface Controller
- This sets up the EMIF waits for HIC_INIT_DONE_TOKEN from device
- sends HIC_START_TOKEN to the device to signal the device to start sampling the Analog channels
- The device sends periodic HIC_DATA_TOKEN token
- The samples are available in the HIC D2H Buffer
- Uses the HIC_INT interrupt from device(which is mapped to GPIO4, XINT1) Refer F28002X device TRM for further details on HIC.

External Connections

- This example will not work on F2838x Control Card and has been tested in TI Internal Validation platform.

Watch Variables

- *sampleCount* - Number of samples of data received from device.
- *xint1Count* - Number interrupts received from device.

12.5 EMIF Registers

This section describes the External Memory Interface Registers.

12.5.1 EMIF Base Address Table (C28)

Table 12-33. EMIF Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
Emif1ConfigRegs	EMIF1_CONFIG_REGS	EMIF1CONFIG_BASE	0x0005_F4C0	YES	YES	-	-	YES
Emif2ConfigRegs	EMIF2_CONFIG_REGS	EMIF2CONFIG_BASE	0x0005_F4E0	YES	-	-	-	YES
Emif1Regs	EMIF_REGS	EMIF1_BASE	0x0004_7000	YES	YES	-	-	YES
Emif2Regs	EMIF_REGS	EMIF2_BASE	0x0004_7800	YES	-	-	-	YES

12.5.2 EMIF_REGS Registers

Table 12-34 lists the memory-mapped registers for the EMIF_REGS registers. All register offset addresses not listed in Table 12-34 should be considered as reserved locations and the register contents should not be modified.

Table 12-34. EMIF_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	RCSR	Revision Code and Status Register		Go
2h	ASYNC_WCCR	Async Wait Cycle Config Register		Go
4h	SDRAM_CR	SDRAM (EMxCS0n) Config Register		Go
6h	SDRAM_RCR	SDRAM Refresh Control Register		Go
8h	ASYNC_CS2_CR	Async 1 (EMxCS2n) Config Register		Go
Ah	ASYNC_CS3_CR	Async 2 (EMxCS3n) Config Register		Go
Ch	ASYNC_CS4_CR	Async 3 (EMxCS4n) Config Register		Go
10h	SDRAM_TR	SDRAM Timing Register		Go
18h	TOTAL_SDRAM_AR	Total SDRAM Accesses Register		Go
1Ah	TOTAL_SDRAM_ACTR	Total SDRAM Activate Register		Go
1Eh	SDR_EXT_TMNG	SDRAM SR/PD Exit Timing Register		Go
20h	INT_RAW	Interrupt Raw Register		Go
22h	INT_MSK	Interrupt Masked Register		Go
24h	INT_MSK_SET	Interrupt Mask Set Register		Go
26h	INT_MSK_CLR	Interrupt Mask Clear Register		Go

Complex bit access types are encoded to fit into small table cells. Table 12-35 shows the codes that are used for access types in this section.

Table 12-35. EMIF_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 12-35. EMIF_REGS Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

12.5.2.1 RCSR Register (Offset = 0h) [Reset = 40000205h]

RCSR is shown in Figure 12-23 and described in Table 12-36.

Return to the Summary Table.

Revision Code and Status Register

Figure 12-23. RCSR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BE	FR	MODULE_ID													
R-0h	R-1h	R-0h													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAJOR_REVISION								MINOR_REVISION							
R-2h								R-5h							

Table 12-36. RCSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	BE	R	0h	EMIF endian mode. 0: Little Endian. 1: Big Endian. Reset type: SYSRSn
30	FR	R	1h	EMIF operating rate. 0: Half Rate. 1: Full Rate. Reset type: SYSRSn
29-16	MODULE_ID	R	0h	EMIF module ID. 0x0000: EMIF_24. 0x000E: EMIF_24 SDRAM. 0x000F: EMIF_24 ASYNC. Reset type: SYSRSn
15-8	MAJOR_REVISION	R	2h	Major Revision. EMIF code revisions are indicated by a revision code taking the format major_revision.minor_revision. Reset type: SYSRSn
7-0	MINOR_REVISION	R	5h	Minor Revision. See major_revision field description. Reset type: SYSRSn

12.5.2.2 ASYNC_WCCR Register (Offset = 2h) [Reset = F0000080h]

ASYNC_WCCR is shown in Figure 12-24 and described in Table 12-37.

Return to the Summary Table.

Async Wait Cycle Config Register

Figure 12-24. ASYNC_WCCR Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	WP0	RESERVED			
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R-0h			
23	22	21	20	19	18	17	16
RESERVED		RESERVED		RESERVED		RESERVED	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
MAX_EXT_WAIT							
R/W-80h							

Table 12-37. ASYNC_WCCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	1h	Reserved
30	RESERVED	R/W	1h	Reserved
29	RESERVED	R/W	1h	Reserved
28	WP0	R/W	1h	Defines the polarity of the EMxWAIT port.: 0: Wait if EMxWAIT port is low. 1: Wait if EMxWAIT port is high. Reset type: SYSRSn
27-24	RESERVED	R	0h	Reserved
23-22	RESERVED	R/W	0h	Reserved
21-20	RESERVED	R/W	0h	Reserved
19-18	RESERVED	R/W	0h	Reserved
17-16	RESERVED	R/W	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7-0	MAX_EXT_WAIT	R/W	80h	The EMIF will wait for (max_ext_wait + 1) * 16 clock cycles before an extended asynchronous cycle is terminated. Reset type: SYSRSn

12.5.2.3 SDRAM_CR Register (Offset = 4h) [Reset = 620h]

SDRAM_CR is shown in Figure 12-25 and described in Table 12-38.

Return to the Summary Table.

SDRAM (EMxCS0n) Config Register

Figure 12-25. SDRAM_CR Register

31	30	29	28	27	26	25	24
SR	PD	PDWR	RESERVED			RESERVED	
R/W-0h	R/W-0h	R/W-0h	R-0h			R/W-0h	
23	22	21	20	19	18	17	16
RESERVED	RESERVED			RESERVED	RESERVED		RESERVED
R/W-0h	R/W-0h			R/W-0h	R/W-0h		R/W-0h
15	14	13	12	11	10	9	8
RESERVED	NM	RESERVED	RESERVED	CL			BIT_11_9_LOCK
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-3h			R-0/W1S-0h
7	6	5	4	3	2	1	0
RESERVED	IBANK			RESERVED	PAGESIGE		
R-0h	R/W-2h			R/W-0h	R/W-0h		

Table 12-38. SDRAM_CR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SR	R/W	0h	Self Refresh. Writing a 1 to this bit will cause connected SDRAM devices to be placed into Self Refresh mode and the EMIF to enter the self refresh state. In this state the EMIF will service all asynchronous memory accesses immediately but any SDRAM access will take at least t_ras + 1 cycles due to the time required for the SDRAM devices to out of Self Refresh mode. If an SDRAM access immediately follows the setting of the sr bit, the access will take t_ras + t_xs + 2 cycles. If both sr and pd bits are set, the EMIF will go into Self Refresh. Reset type: SYSRSn
30	PD	R/W	0h	Power Down. Writing a 1 to this bit will cause connected SDRAM devices to be placed into Power Down mode. If both sr and pd bits are set, the EMIF will go into Self Refresh. Reset type: SYSRSn
29	PDWR	R/W	0h	Perform refreshes during Power Down. Writing a 1 to this bit will cause the EMIF to exit the power down state and issue an AUTO REFRESH command every time Refresh May level is set. Reset type: SYSRSn
28-26	RESERVED	R	0h	Reserved
25-23	RESERVED	R/W	0h	Reserved
22-20	RESERVED	R/W	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18-17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15	RESERVED	R	0h	Reserved
14	NM	R/W	0h	Narrow mode. Set to 1 when system bus width to memory bus width is 2:1 for SDR SDRAM. Set to 0 when system bus width to memory bus width is 1:1 for SDR SDRAM. A write to this field will cause the EMIF to start the SDRAM initialization sequence. Reset type: SYSRSn

Table 12-38. SDRAM_CR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11-9	CL	R/W	3h	The value of this field defines the CAS latency to be used when accessing connected SDRAM devices. Only CAS latencies of 2 (cl = 2) and 3 (cl = 3) are supported. A write to this field will cause the EMIF to start the SDRAM initialization sequence. Reset type: SYSRSn
8	BIT_11_9_LOCK	R-0/W1S	0h	Bits 11 to 9 can only be written if this bit is set to 1. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6-4	IBANK	R/W	2h	Defines number of banks inside connected SDRAM devices: 000: 1 bank SDRAM devices. 001: 2 bank SDRAM devices. 010: 4 bank SDRAM devices. 011: Reserved. 1xx: Reserved. A write to this field will cause the EMIF to start the SDRAM initialization sequence. Reset type: SYSRSn
3	RESERVED	R/W	0h	Reserved
2-0	PAGESIGE	R/W	0h	Defines the internal page size of connected SDRAM devices: 000: 256-word pages requiring 8 column address bits. 001: 512-word pages requiring 9 column address bits. 010: 1024-word pages requiring 10 column address bits. 011: 2048-word pages requiring 11 column address bits. 1xx: Reserved. A write to this field will cause the EMIF to start the SDRAM initialization sequence. Reset type: SYSRSn

12.5.2.4 SDRAM_RCR Register (Offset = 6h) [Reset = 80h]

SDRAM_RCR is shown in Figure 12-26 and described in Table 12-39.

Return to the Summary Table.

SDRAM Refresh Control Register

Figure 12-26. SDRAM_RCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						RESERVED	
R-0h						R/W-0h	
15	14	13	12	11	10	9	8
RESERVED			REFRESH_RATE				
R-0h			R/W-80h				
7	6	5	4	3	2	1	0
REFRESH_RATE							
R/W-80h							

Table 12-39. SDRAM_RCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved
18-16	RESERVED	R/W	0h	Reserved
15-13	RESERVED	R	0h	Reserved
12-0	REFRESH_RATE	R/W	80h	Value in this field is used to define the rate at which connected SDRAM devices will be refreshed, as follows: SDRAM refresh rate = EMIF rate/refresh_rate where EMIF rate=clk rate when full_rate=1, or EMIF rate=1/2 clk rate when full_rate=0. Writing a value < 0x0020 to this field will cause it to be loaded with (2 * t_rfc) + 1 value from SDRAM Timing register. Reset type: SYSRSn

12.5.2.5 ASYNC_CS2_CR Register (Offset = 8h) [Reset = 3FFFFFFDh]

ASYNC_CS2_CR is shown in Figure 12-27 and described in Table 12-40.

Return to the Summary Table.

Async 1 (EMxCS2n) Config Register

Figure 12-27. ASYNC_CS2_CR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SS	EW	W_SETUP				W_STROBE						W_HOLD			R_SETUP
R/W-0h	R/W-0h	R/W-Fh				R/W-3Fh						R/W-7h			R/W-Fh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R_SETUP			R_STROBE						R_HOLD			TA		ASIZE	
R/W-Fh			R/W-3Fh						R/W-7h			R/W-3h		R/W-1h	

Table 12-40. ASYNC_CS2_CR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SS	R/W	0h	Select Strobe mode. Set to 1 if chip selects need to have write or read strobe timing. Reset type: SYSRSn
30	EW	R/W	0h	Extend Wait mode. Set to 1 if extended asynchronous cycles are required based on EMxWAIT. Reset type: SYSRSn
29-26	W_SETUP	R/W	Fh	Write Strobe Setup cycles. Number of EMxCLK cycles from EMxAy, EMxBAy, EMxDQMy, and EMxCS2n being set to EMxWEn asserted, minus one cycle. The reset value is 16 cycles. Reset type: SYSRSn
25-20	W_STROBE	R/W	3Fh	Write Strobe Duration cycles. Number of EMxCLK cycles for which EMxWEn is held active, minus one cycle. The reset value is 64 cycles. This field cannot be zero when ew = 1. Reset type: SYSRSn
19-17	W_HOLD	R/W	7h	Write Strobe Hold cycles. Number of EMxCLK cycles for which EMxAy, EMxBAy, EMxDQMy, and EMxCS2n are held after EMxWEn has been deasserted, minus one cycle. The reset value is 8 cycles. Reset type: SYSRSn
16-13	R_SETUP	R/W	Fh	Read Strobe Setup cycles. Number of EMxCLK cycles from EMxAy, EMxBAy, EMxDQMy, and EMxCS2n being set to EMxOEn asserted, minus one cycle. The reset value is 16 cycles. Reset type: SYSRSn
12-7	R_STROBE	R/W	3Fh	Read Strobe Duration cycles. Number of EMxCLK cycles for which EMxOEn is held active, minus one cycle. The reset value is 64 cycles. This field cannot be zero when ew = 1. Reset type: SYSRSn
6-4	R_HOLD	R/W	7h	Read Strobe Hold cycles. Number of EMxCLK cycles for which EMxAy, EMxBAy, EMxDQMy, and EMxCS2n are held after EMxOEn has been deasserted, minus one cycle. The reset value is 8 cycles. Reset type: SYSRSn
3-2	TA	R/W	3h	Turn Around cycles. Number of EMxCLK cycles between the end of one asynchronous memory access and the start of another asynchronous memory access, minus one cycle. This delay is not incurred between a read followed by a read, or a write followed by a write to the same chip select. The reset value is 4 cycles. Reset type: SYSRSn

Table 12-40. ASYNC_CS2_CR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	ASIZE	R/W	1h	Asynchronous Memory Size. Defines the width of the asynchronous device's data bus : 00: 8 Bit data bus. 01: 16 Bit data bus. 10: 32 Bit data bus. 11: Reserved. Reset type: SYSRSn

12.5.2.6 ASYNC_CS3_CR Register (Offset = Ah) [Reset = 3FFFFFFDh]

ASYNC_CS3_CR is shown in Figure 12-28 and described in Table 12-41.

Return to the Summary Table.

Async 2 (EMxCS3n) Config Register

Figure 12-28. ASYNC_CS3_CR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SS	EW	W_SETUP				W_STROBE						W_HOLD			R_SETUP
R/W-0h	R/W-0h	R/W-Fh				R/W-3Fh						R/W-7h			R/W-Fh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R_SETUP			R_STROBE						R_HOLD			TA		ASIZE	
R/W-Fh			R/W-3Fh						R/W-7h			R/W-3h		R/W-1h	

Table 12-41. ASYNC_CS3_CR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SS	R/W	0h	Select Strobe mode. Set to 1 if chip selects need to have write or read strobe timing. Reset type: SYSRSn
30	EW	R/W	0h	Extend Wait mode. Set to 1 if extended asynchronous cycles are required based on EMxWAIT. Reset type: SYSRSn
29-26	W_SETUP	R/W	Fh	Write Strobe Setup cycles. Number of EMxCLK cycles from EMxAy, EMxBAy, EMxDQMy, and EMxCS3n being set to EMxWEn asserted, minus one cycle. The reset value is 16 cycles. Reset type: SYSRSn
25-20	W_STROBE	R/W	3Fh	Write Strobe Duration cycles. Number of EMxCLK cycles for which EMxWEn is held active, minus one cycle. The reset value is 64 cycles. This field cannot be zero when ew = 1. Reset type: SYSRSn
19-17	W_HOLD	R/W	7h	Write Strobe Hold cycles. Number of EMxCLK cycles for which EMxAy, EMxBAy, EMxDQMy, and EMxCS3n are held after EMxWEn has been deasserted, minus one cycle. The reset value is 8 cycles. Reset type: SYSRSn
16-13	R_SETUP	R/W	Fh	Read Strobe Setup cycles. Number of EMxCLK cycles from EMxAy, EMxBAy, EMxDQMy, and EMxCS3n being set to EMxOEn asserted, minus one cycle. The reset value is 16 cycles. Reset type: SYSRSn
12-7	R_STROBE	R/W	3Fh	Read Strobe Duration cycles. Number of EMxCLK cycles for which EMxOEn is held active, minus one cycle. The reset value is 64 cycles. This field cannot be zero when ew = 1. Reset type: SYSRSn
6-4	R_HOLD	R/W	7h	Read Strobe Hold cycles. Number of EMxCLK cycles for which EMxAy, EMxBAy, EMxDQMy, and EMxCS3n are held after EMxOEn has been deasserted, minus one cycle. The reset value is 8 cycles. Reset type: SYSRSn
3-2	TA	R/W	3h	Turn Around cycles. Number of EMxCLK cycles between the end of one asynchronous memory access and the start of another asynchronous memory access, minus one cycle. This delay is not incurred between a read followed by a read, or a write followed by a write to the same chip select. The reset value is 4 cycles. Reset type: SYSRSn

Table 12-41. ASYNC_CS3_CR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	ASIZE	R/W	1h	Asynchronous Memory Size. Defines the width of the asynchronous device's data bus : 00: 8 Bit data bus. 01: 16 Bit data bus. 10: 32 Bit data bus. 11: Reserved. Reset type: SYSRSn

12.5.2.7 ASYNC_CS4_CR Register (Offset = Ch) [Reset = 3FFFFFFDh]

ASYNC_CS4_CR is shown in Figure 12-29 and described in Table 12-42.

Return to the Summary Table.

Async 3 (EMxCS4n) Config Register

Figure 12-29. ASYNC_CS4_CR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SS	EW	W_SETUP				W_STROBE						W_HOLD			R_SETUP
R/W-0h	R/W-0h	R/W-Fh				R/W-3Fh						R/W-7h			R/W-Fh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R_SETUP			R_STROBE						R_HOLD			TA		ASIZE	
R/W-Fh			R/W-3Fh						R/W-7h			R/W-3h		R/W-1h	

Table 12-42. ASYNC_CS4_CR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SS	R/W	0h	Select Strobe mode. Set to 1 if chip selects need to have write or read strobe timing. Reset type: SYSRSn
30	EW	R/W	0h	Extend Wait mode. Set to 1 if extended asynchronous cycles are required based on EMxWAIT. Reset type: SYSRSn
29-26	W_SETUP	R/W	Fh	Write Strobe Setup cycles. Number of EMxCLK cycles from EMxAy, EMxBAy, EMxDQMy, and EMxCS4n being set to EMxWEn asserted, minus one cycle. The reset value is 16 cycles. Reset type: SYSRSn
25-20	W_STROBE	R/W	3Fh	Write Strobe Duration cycles. Number of EMxCLK cycles for which EMxWEn is held active, minus one cycle. The reset value is 64 cycles. This field cannot be zero when ew = 1. Reset type: SYSRSn
19-17	W_HOLD	R/W	7h	Write Strobe Hold cycles. Number of EMxCLK cycles for which EMxAy, EMxBAy, EMxDQMy, and EMxCS4n are held after EMxWEn has been deasserted, minus one cycle. The reset value is 8 cycles. Reset type: SYSRSn
16-13	R_SETUP	R/W	Fh	Read Strobe Setup cycles. Number of EMxCLK cycles from EMxAy, EMxBAy, EMxDQMy, and EMxCS4n being set to EMxOEn asserted, minus one cycle. The reset value is 16 cycles. Reset type: SYSRSn
12-7	R_STROBE	R/W	3Fh	Read Strobe Duration cycles. Number of EMxCLK cycles for which EMxOEn is held active, minus one cycle. The reset value is 64 cycles. This field cannot be zero when ew = 1. Reset type: SYSRSn
6-4	R_HOLD	R/W	7h	Read Strobe Hold cycles. Number of EMxCLK cycles for which EMxAy, EMxBAy, EMxDQMy, and EMxCS4n are held after EMxOEn has been deasserted, minus one cycle. The reset value is 8 cycles. Reset type: SYSRSn
3-2	TA	R/W	3h	Turn Around cycles. Number of EMxCLK cycles between the end of one asynchronous memory access and the start of another asynchronous memory access, minus one cycle. This delay is not incurred between a read followed by a read, or a write followed by a write to the same chip select. The reset value is 4 cycles. Reset type: SYSRSn

Table 12-42. ASYNC_CS4_CR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	ASIZE	R/W	1h	Asynchronous Memory Size. Defines the width of the asynchronous device's data bus : 00: 8 Bit data bus. 01: 16 Bit data bus. 10: 32 Bit data bus. 11: Reserved. Reset type: SYSRSn

12.5.2.8 SDRAM_TR Register (Offset = 10h) [Reset = 19214610h]

SDRAM_TR is shown in Figure 12-30 and described in Table 12-43.

Return to the Summary Table.

SDRAM Timing Register

Figure 12-30. SDRAM_TR Register

31	30	29	28	27	26	25	24
T_RFC					T_RP		
R/W-3h					R/W-1h		
23	22	21	20	19	18	17	16
RESERVED	T_RCD			RESERVED	T_WR		
R-0h	R/W-2h			R-0h	R/W-1h		
15	14	13	12	11	10	9	8
T_RAS				T_RC			
R/W-4h				R/W-6h			
7	6	5	4	3	2	1	0
RESERVED	T_RRD			RESERVED			
R-0h	R/W-1h			R-0h			

Table 12-43. SDRAM_TR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	T_RFC	R/W	3h	Minimum number of EMxCLK cycles from Refresh or Load Mode to Refresh or Activate, minus one. Reset type: SYSRSn
26-24	T_RP	R/W	1h	Minimum number of EMxCLK cycles from Precharge to Activate or Refresh, minus one. Reset type: SYSRSn
23	RESERVED	R	0h	Reserved
22-20	T_RCD	R/W	2h	Minimum number of EMxCLK cycles from Activate to Read or Write, minus one. Reset type: SYSRSn
19	RESERVED	R	0h	Reserved
18-16	T_WR	R/W	1h	For SDR, this is equal to minimum number of EMxCLK cycles from last Write transfer to Precharge, minus one. Reset type: SYSRSn
15-12	T_RAS	R/W	4h	Minimum number of EMxCLK cycles from Activate to Precharge, minus one. t_ras >= t_rcd. Reset type: SYSRSn
11-8	T_RC	R/W	6h	Minimum number of EMxCLK cycles from Activate to Activate minus one. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6-4	T_RRD	R/W	1h	Minimum number of EMxCLK cycles from Activate to Activate for a different bank, minus one. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

12.5.2.9 TOTAL_SDRAM_AR Register (Offset = 18h) [Reset = 0h]

TOTAL_SDRAM_AR is shown in Figure 12-31 and described in Table 12-44.

Return to the Summary Table.

Total SDRAM Accesses Register

Figure 12-31. TOTAL_SDRAM_AR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOTAL_SDRAM_AR																															
R-0h																															

Table 12-44. TOTAL_SDRAM_AR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TOTAL_SDRAM_AR	R	0h	Indicates the total number of accesses to SDRAM from a master (CPUx/CPUX.DMA). This counter is incremented by two for a single access crossing page boundaries. Reset type: SYSRSn

12.5.2.10 TOTAL_SDRAM_ACTR Register (Offset = 1Ah) [Reset = 0h]

TOTAL_SDRAM_ACTR is shown in Figure 12-32 and described in Table 12-45.

Return to the Summary Table.

Total SDRAM Activate Register

Figure 12-32. TOTAL_SDRAM_ACTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOTAL_SDRAM_ACTR																															
R-0h																															

Table 12-45. TOTAL_SDRAM_ACTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TOTAL_SDRAM_ACTR	R	0h	Indicates the total number of SDRAM accesses which require an activate command. Reset type: SYSRSn

12.5.2.11 SDR_EXT_TMNG Register (Offset = 1Eh) [Reset = 7h]

SDR_EXT_TMNG is shown in Figure 12-33 and described in Table 12-46.

Return to the Summary Table.

SDRAM SR/PD Exit Timing Register

Figure 12-33. SDR_EXT_TMNG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RESERVED											T_XS				
R-0h																R-0h											R/W-7h				

Table 12-46. SDR_EXT_TMNG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-5	RESERVED	R	0h	Reserved
4-0	T_XS	R/W	7h	This is equal to minimum number of EMxCLK cycles from Self Refresh exit to any command, minus one. For SDR SDRAM, this count should satisfy tXSR. Reset type: SYSRSn

12.5.2.12 INT_RAW Register (Offset = 20h) [Reset = 0h]

INT_RAW is shown in Figure 12-34 and described in Table 12-47.

Return to the Summary Table.

Interrupt Raw Register

Figure 12-34. INT_RAW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED										WR				LT	AT
R-0h										R/W1S-0h				R/W1S-0h	R/W1S-0h

Table 12-47. INT_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-6	RESERVED	R	0h	Reserved
5-2	WR	R/W1S	0h	Wait Rise. Set to 1 by hardware to indicate rising edge on the corresponding EMxWAIT has been detected. The WPx bits in the Async Wait Cycle Config register has no effect on these bits. Writing a 1 will clear these bits as well as the wr_masked bits in the Interrupt Masked register. Writing a 0 has no effect. Reset type: SYSRSn
1	LT	R/W1S	0h	Line Trap. Set to 1 by hardware to indicate illegal memory access type or invalid cache line size. Writing a 1 will clear this bit as well as the lt_masked bit in the Interrupt Masked register. Writing a 0 has no effect. Reset type: SYSRSn
0	AT	R/W1S	0h	Asynchronous Timeout. Set to 1 by hardware to indicate that during an extended asynchronous memory access cycle, the EMxWAIT signal did not go inactive within the number of cycles defined by the max_ext_wait field in Async Wait Cycle Config register. Writing a 1 will clear this bit as well as the at_masked bit in the Interrupt Masked register. Writing a 0 has no effect. Reset type: SYSRSn

12.5.2.13 INT_MSK Register (Offset = 22h) [Reset = 0h]

INT_MSK is shown in Figure 12-35 and described in Table 12-48.

Return to the Summary Table.

Interrupt Masked Register

Figure 12-35. INT_MSK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		WR_MASKED				LT_MASKED	AT_MASKED
R-0h		R/W1S-0h				R/W1S-0h	R/W1S-0h

Table 12-48. INT_MSK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-6	RESERVED	R	0h	Reserved
5-2	WR_MASKED	R/W1S	0h	Masked Wait Rise. Set to 1 by hardware to indicate rising edge on the corresponding EMxWAIT has been detected, only if the wr_mask_set bit in the Interrupt Mask Set register is set to 1. The WPx bits in the Async Wait Cycle Config register has no effect on these bits. Writing a 1 will clear these bits as well as the wr bits in the Interrupt Raw register. Writing a 0 has no effect. Reset type: SYSRSn
1	LT_MASKED	R/W1S	0h	Masked Line Trap. Set to 1 by hardware to indicate illegal memory access type or invalid cache line size, only if the lt_mask_set bit in the Interrupt Mask Set register is set to 1. Writing a 1 will clear this bit as well as the lt bit in the Interrupt Raw register. Writing a 0 has no effect. Reset type: SYSRSn
0	AT_MASKED	R/W1S	0h	Masked Asynchronous Timeout. Set to 1 by hardware to indicate that during an extended asynchronous memory access cycle, the EMxWAIT signal did not go inactive within the number of cycles defined by the max_ext_wait field in Async Wait Cycle Config register, only if the at_mask_set bit in the Interrupt Mask Set register is set to 1. Writing a 1 will clear this bit as well as the at bit in the Interrupt Raw register. Writing a 0 has no effect. Reset type: SYSRSn

12.5.2.14 INT_MSK_SET Register (Offset = 24h) [Reset = 0h]

INT_MSK_SET is shown in Figure 12-36 and described in Table 12-49.

Return to the Summary Table.

Interrupt Mask Set Register

Figure 12-36. INT_MSK_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		WR_MASK_SET				LT_MASK_SET	AT_MASK_SET
R-0h		R/W1S-0h				R/W1S-0h	R/W1S-0h

Table 12-49. INT_MSK_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-6	RESERVED	R	0h	Reserved
5-2	WR_MASK_SET	R/W1S	0h	Mask set for wr_masked bits in the Interrupt Masked Register. Writing a 1 will enable the interrupts, and set these bits as well as the wr_mask_clr bits in the Interrupt Mask Clear register. Writing a 0 has no effect. Reset type: SYSRSn
1	LT_MASK_SET	R/W1S	0h	Mask set for lt_masked bit in the Interrupt Masked Register. Writing a 1 will enable the interrupt, and set this bit as well as the lt_mask_clr bit in the Interrupt Mask Clear register. Writing a 0 has no effect. Reset type: SYSRSn
0	AT_MASK_SET	R/W1S	0h	Mask set for at_masked bit in the Interrupt Masked Register. Writing a 1 will enable the interrupt, and set this bit as well as the at_mask_clr bit in the Interrupt Mask Clear register. Writing a 0 has no effect. Reset type: SYSRSn

12.5.2.15 INT_MSK_CLR Register (Offset = 26h) [Reset = 0h]

INT_MSK_CLR is shown in Figure 12-37 and described in Table 12-50.

Return to the Summary Table.

Interrupt Mask Clear Register

Figure 12-37. INT_MSK_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		WR_MASK_CLR				LT_MASK_CLR	AT_MASK_CLR
R-0h		R/W1S-0h				R/W1S-0h	R/W1S-0h

Table 12-50. INT_MSK_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-6	RESERVED	R	0h	Reserved
5-2	WR_MASK_CLR	R/W1S	0h	Mask clear for wr_masked bits in the Interrupt Masked Register. Writing a 1 will disable the interrupts, and clear these bits as well as the wr_mask_set bits in the Interrupt Mask Set register. Writing a 0 has no effect. Reset type: SYSRSn
1	LT_MASK_CLR	R/W1S	0h	Mask clear for lt_masked bit in the Interrupt Masked Register. Writing a 1 will disable the interrupt, and clear this bit as well as the lt_mask_set bit in the Interrupt Mask Set register. Writing a 0 has no effect. Reset type: SYSRSn
0	AT_MASK_CLR	R/W1S	0h	Mask clear for at_masked bit in the Interrupt Masked Register. Writing a 1 will disable the interrupt, and clear this bit as well as the at_mask_set bit in the Interrupt Mask Set register. Writing a 0 has no effect. Reset type: SYSRSn

12.5.3 EMIF1_CONFIG_REGS Registers

Table 12-51 lists the memory-mapped registers for the EMIF1_CONFIG_REGS registers. All register offset addresses not listed in Table 12-51 should be considered as reserved locations and the register contents should not be modified.

Table 12-51. EMIF1_CONFIG_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	EMIF1LOCK	EMIF1 Config Lock Register	EALLOW	Go
2h	EMIF1COMMIT	EMIF1 Config Lock Commit Register	EALLOW	Go
4h	EMIF1MSEL	EMIF1 Master Sel Register	EALLOW	Go
8h	EMIF1ACCPROT0	EMIF1 Config Register 0	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 12-52 shows the codes that are used for access types in this section.

Table 12-52. EMIF1_CONFIG_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

12.5.3.1 EMIF1LOCK Register (Offset = 0h) [Reset = 0h]

EMIF1LOCK is shown in Figure 12-38 and described in Table 12-53.

Return to the Summary Table.

EMIF1 Config Lock Register

Figure 12-38. EMIF1LOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK_EMIF1
R-0h							R/W-0h

Table 12-53. EMIF1LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	LOCK_EMIF1	R/W	0h	Locks the write to access protection and master select fields for EMIF1: 0: Write to ACCPROT and Mselect fields are allowed. 1: Write to ACCPROT and Mselect fields are blocked. Reset type: SYSRSn

12.5.3.2 EMIF1COMMIT Register (Offset = 2h) [Reset = 0h]

EMIF1COMMIT is shown in Figure 12-39 and described in Table 12-54.

Return to the Summary Table.

EMIF1 Config Lock Commit Register

Figure 12-39. EMIF1COMMIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							COMMIT_EMIF1
R-0h							R/WSonce-0h

Table 12-54. EMIF1COMMIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	COMMIT_EMIF1	R/WSonce	0h	Permanently Locks the write to access protection and master select fields for EMIF1: 0: Write to ACCPROT and Mselect fields are allowed based on value of lock field in EMIF1LOCK register. 1: Write to ACCPROT and Mselect fields are permanently blocked. Reset type: SYSRSn

12.5.3.3 EMIF1MSEL Register (Offset = 4h) [Reset = 0h]

EMIF1MSEL is shown in Figure 12-40 and described in Table 12-55.

Return to the Summary Table.

EMIF1 Master Sel Register

Figure 12-40. EMIF1MSEL Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
KEY							
R-0/W-0h							
7	6	5	4	3	2	1	0
KEY				RESERVED		MSEL_EMIF1	
R-0/W-0h				R-0h		R/W-0h	

Table 12-55. EMIF1MSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	KEY	R-0/W	0h	Writing the value 0x93A5CE7 will allow the writing of the EMIF1Mselect bits, else writes are ignored. Reads will return 0. Reset type: CPU1.SYSRSn
3-2	RESERVED	R	0h	Reserved
1-0	MSEL_EMIF1	R/W	0h	Master Select for EMIF1: 00: CPU1 is master but not grabbed. CPU2 can grab the master ownership by changing this value to "10". 01: CPU1 is master. 10: CPU2 is master. 11: CPU1 is master but not grabbed. CPU2 can grab the master ownership by changing this value to "10". Reset type: CPU1.SYSRSn

12.5.3.4 EMIF1ACCPROT0 Register (Offset = 8h) [Reset = 0h]

EMIF1ACCPROT0 is shown in Figure 12-41 and described in Table 12-56.

Return to the Summary Table.

EMIF1 Config Register 0

Figure 12-41. EMIF1ACCPROT0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					DMAWRPROT_EMIF1	CPUWRPROT_EMIF1	FETCHPROT_EMIF1
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 12-56. EMIF1ACCPROT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-3	RESERVED	R	0h	Reserved
2	DMAWRPROT_EMIF1	R/W	0h	DMA WR Protection For EMIF1: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
1	CPUWRPROT_EMIF1	R/W	0h	CPU WR Protection For EMIF1: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
0	FETCHPROT_EMIF1	R/W	0h	Fetch Protection For EMIF1: 0: CPU Fetches are allowed. 1: CPU Fetches are blocked. Reset type: SYSRSn

12.5.4 EMIF2_CONFIG_REGS Registers

Table 12-57 lists the memory-mapped registers for the EMIF2_CONFIG_REGS registers. All register offset addresses not listed in Table 12-57 should be considered as reserved locations and the register contents should not be modified.

Table 12-57. EMIF2_CONFIG_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	EMIF2LOCK	EMIF2 Config Lock Register		Go
2h	EMIF2COMMIT	EMIF2 Config Lock Commit Register	EALLOW	Go
8h	EMIF2ACCPROT0	EMIF2 Config Register 0	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 12-58 shows the codes that are used for access types in this section.

Table 12-58. EMIF2_CONFIG_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

12.5.4.1 EMIF2LOCK Register (Offset = 0h) [Reset = 0h]

EMIF2LOCK is shown in Figure 12-42 and described in Table 12-59.

Return to the Summary Table.

EMIF2 Config Lock Register

Figure 12-42. EMIF2LOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK_EMIF2
R-0h							R/W-0h

Table 12-59. EMIF2LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	LOCK_EMIF2	R/W	0h	Locks the write to access protection fields for EMIF2: 0: Write to ACCPROT and Mselect fields are allowed. 1: Write to ACCPROT and Mselect fields are blocked. Reset type: CPU1.SYSRSn

12.5.4.2 EMIF2COMMIT Register (Offset = 2h) [Reset = 0h]

EMIF2COMMIT is shown in Figure 12-43 and described in Table 12-60.

Return to the Summary Table.

EMIF2 Config Lock Commit Register

Figure 12-43. EMIF2COMMIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							COMMIT_EMIF2
R-0h							R/WSonce-0h

Table 12-60. EMIF2COMMIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	COMMIT_EMIF2	R/WSonce	0h	Permanently Locks the write to access protection fields for EMIF2: 0: Write to ACCPROT fields are allowed based on value of lock field in EMIF2LOCK register. 1: Write to ACCPROT fields are permanently blocked. Reset type: CPU1.SYSRSn

12.5.4.3 EMIF2ACCPROT0 Register (Offset = 8h) [Reset = 0h]

EMIF2ACCPROT0 is shown in Figure 12-44 and described in Table 12-61.

Return to the Summary Table.

EMIF2 Config Register 0

Figure 12-44. EMIF2ACCPROT0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						CPUWRPROT_EMIF2	FETCHPROT_EMIF2
R-0h						R/W-0h	R/W-0h

Table 12-61. EMIF2ACCPROT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-2	RESERVED	R	0h	Reserved
1	CPUWRPROT_EMIF2	R/W	0h	CPU WR Protection For EMIF2: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
0	FETCHPROT_EMIF2	R/W	0h	Fetch Protection For EMIF2 0: CPU Fetches are allowed. 1: CPU Fetches are blocked. Reset type: SYSRSn

12.5.5 EMIF Registers to Driverlib Functions

Table 12-62. EMIF Registers to Driverlib Functions

File	Driverlib Function
RCSR	
-	
ASYNC_WCCR	
emif.h	EMIF_setAsyncWaitPolarity
emif.h	EMIF_setAsyncMaximumWaitCycles
SDRAM_CR	
emif.h	EMIF_setSyncMemoryConfig
emif.h	EMIF_enableSyncSelfRefresh
emif.h	EMIF_disableSyncSelfRefresh
emif.h	EMIF_enableSyncPowerDown
emif.h	EMIF_disableSyncPowerDown
emif.h	EMIF_enableSyncRefreshInPowerDown
emif.h	EMIF_disableSyncRefreshInPowerDown

Table 12-62. EMIF Registers to Driverlib Functions (continued)

File	Driverlib Function
SDRAM_RCR	
emif.h	EMIF_setSyncRefreshRate
ASYNC_CS2_CR	
emif.h	EMIF_setAsyncMode
emif.h	EMIF_enableAsyncExtendedWait
emif.h	EMIF_setAsyncTimingParams
emif.h	EMIF_setAsyncDataBusWidth
ASYNC_CS3_CR	
-	See ASYNC_CS2_CR
ASYNC_CS4_CR	
-	See ASYNC_CS2_CR
SDRAM_TR	
emif.h	EMIF_setSyncTimingParams
TOTAL_SDRAM_AR	
emif.h	EMIF_getSyncTotalAccesses
TOTAL_SDRAM_ACTR	
emif.h	EMIF_getSyncTotalActivateAccesses
SDR_EXT_TMNG	
emif.h	EMIF_setSyncSelfRefreshExitTmng
INT_RAW	
-	
INT_MSK	
emif.h	EMIF_enableAsyncInterrupt
emif.h	EMIF_disableAsyncInterrupt
emif.h	EMIF_getAsyncInterruptStatus
emif.h	EMIF_clearAsyncInterruptStatus
INT_MSK_SET	
emif.h	EMIF_enableAsyncInterrupt
INT_MSK_CLR	
emif.h	EMIF_disableAsyncInterrupt

This page intentionally left blank.

Chapter 13

Flash Module

This chapter describes the Flash module.

13.1 Introduction to Flash and OTP Memory

Flash is an electrically erasable/programmable nonvolatile memory that can be programmed and erased many times to ease code development. Flash memory can be used primarily as a program memory for the core, and secondarily as static data memory.

This section describes the proper sequence to configure the wait states and operating mode of Flash. It also includes information on Flash and OTP power modes, how to improve Flash performance by enabling the Flash prefetch/cache mode, and the SECDED safety feature.

13.1.1 FLASH Related Collateral

Foundational Materials

- C2000 Academy - FLASH
- Embedded Flash Memory (Video)

Getting Started Materials

- Flash API Reference Guide: C2000Ware_VERSION#/libraries/flash_api/DEVICE_GPN/docs
- *Serial Flash Programming of C2000 Microcontrollers Application Report*
- [FAQ] FAQ for Flash ECC usage in C2000 devices - Includes ECC test mode, Linker ECC options:
- [FAQ] FAQ on Flash API usage for C2000 devices
- [FAQ] Flash - How to modify an application from RAM configuration to Flash configuration?
- [FAQ] How can we improve the Flash tool performance?
- [FAQ] TI C2000 Device Programming Tools and Services

13.1.2 Features

Features of Flash memory include:

- Dedicated Flash bank in the CPU1 and CPU2 subsystem (refer to the device data sheet for the size of Flash bank)
- Dedicated Flash bank in the Connectivity Manager (CM) subsystem (refer to the device data sheet for the size of Flash bank)
- Dedicated Flash module controller (FMC) in the CPU1, CPU2, and CM subsystems for each bank
- 128 bits (bank width) can be programmed at a time along with ECC
- Multiple sectors providing the option of leaving some sectors programmed and only erasing specific sectors using sector erase command.
- User-programmable OTP locations for configuring security, OTP boot-mode and boot-mode select pins (if the user is unable to use the factory-default boot-mode select pins)
- Single-Flash pump shared by the CPU1, CPU2, and CM subsystems
- Hardware Flash pump semaphore to control ownership of the pump between the three FMCs.
- Enhanced performance using the code prefetch mechanism and data cache in CPU1-FMC, CPU2-FMC and CM-FMC
- Configurable wait states to give the best performance for a given execution speed
- Safety Features:
  - SECDED - single-error correction and double-error detection is supported in all three FMCs
  - Address bits are included in ECC
  - Test mode to check the health of ECC logic
- Supports low-power modes for Flash bank and pump for power savings
- Built-in power mode control logic
- Integrated Flash program/erase state machine (FSM) in all three FMCs:
  - Simple Flash API algorithms
  - Fast erase and program times (refer to the device data sheet for details)
- Code Security Module (CSM) to prevent unauthorized access to the Flash (refer to Chapter 6 for details)

13.1.3 Flash Tools

Texas Instruments provides the following tools for Flash:

- Code Composer Studio™ (CCS) IDE - the development environment with integrated Flash plugin. TI recommends performing a debug reset and restart after programming the code into Flash using CCS.
- Flash API Library - a set of software peripheral functions to erase/program Flash
- UniFlash - standalone tool to erase/program/verify the Flash content through JTAG. No CCS is required.
- Users must check and install available updates for CCS On-Chip Flash Plugin and UniFlash tools.

13.1.4 Default Flash Configuration

The following are Flash module configuration settings at power-up:

- Dedicated Flash banks are in sleep mode (BNKPWR bit field in the FBFALLBAC register)
- Shared pump is in sleep mode (PMPPWR bit field in the FPAC1 register)
- ECC is enabled
- Wait-states are set to the maximum (0xF)
- Code-prefetch mechanism and data cache are disabled in all three FMCs
- Bank and pump active grace periods are set to 0x0 (refer to the BAGP field in the FBAC register and PAGP bit field in the FPAC2 register)

During the boot process, the boot ROM performs a dummy read of the Code Security Module (CSM) password locations in the OTP. This read is performed to unlock a new device that has no password stored in it, so that Flash programming or loading of code into CSM-protected SRAM can be performed. On devices with a password, this read has no effect and the device remains locked. One effect of this read is that the Flash will transition from the sleep (reset) state to the active state.

User application software must initialize wait-states using the FRDCNTL register, and configure cache/prefetch features using the FRD_INTF_CTRL register, to achieve optimum system performance. Software that configures Flash settings like wait-states, cache/prefetch features, and so on, must be executed only from RAM memory, **not** from Flash memory.

Note

Before initializing wait-states, turn off the prefetch and data caching in the FRD_INTF_CTRL register.

13.2 Flash Bank, OTP, and Pump

There is a dedicated Flash bank for CPU subsystem(CPU1, CPU2, and CM) called CPU1 Flash bank, CPU2 Flash bank, and CM Flash bank. Also, there is a one-time programmable (OTP) memory on the CPU1, CPU2, and CM subsystems called USER OTP, which the user can program only once and cannot erase. Flash and OTP are uniformly mapped in both program and data memory space.

The CPU subsystem has a TI-OTP that contains manufacturing information like settings used by the Flash state machine for erase and program operations, and so on. Users can read TI-OTP but TI-OTP cannot be programmed or erased. For memory maps and size information for the Flash Banks, TI-OTP, USER OTP, and corresponding ECC locations, refer to the device data sheet.

The CPU1 Flash bank/USER OTP, CPU2 Flash bank/USER OTP and CM Flash bank/USER OTP share a common Flash pump. A hardware semaphore, called the Flash pump semaphore, is provided to control the access of the Flash pump between the CPU1, CPU2 and CM subsystem.

Figure 6-2 shows the user-programmable OTP locations in CPU1 USER-OTP. For more information on the functionality of these fields, refer to Section 6.1 and Chapter 5.

13.3 Flash Module Controller (FMC)

There is a dedicated Flash module controller in CPU1 subsystem (CPU1-FMC), CPU2 subsystem (CPU2-FMC), and CM subsystem (CM-FMC). The CPU1 in the CPU1 subsystem interfaces with the CPU1 Flash module controller (CPU1-FMC), which in turn, interfaces with the CPU1 Flash bank and shared pump to perform erase/program operations as well as to read data/execute code from the CPU1 Flash bank.

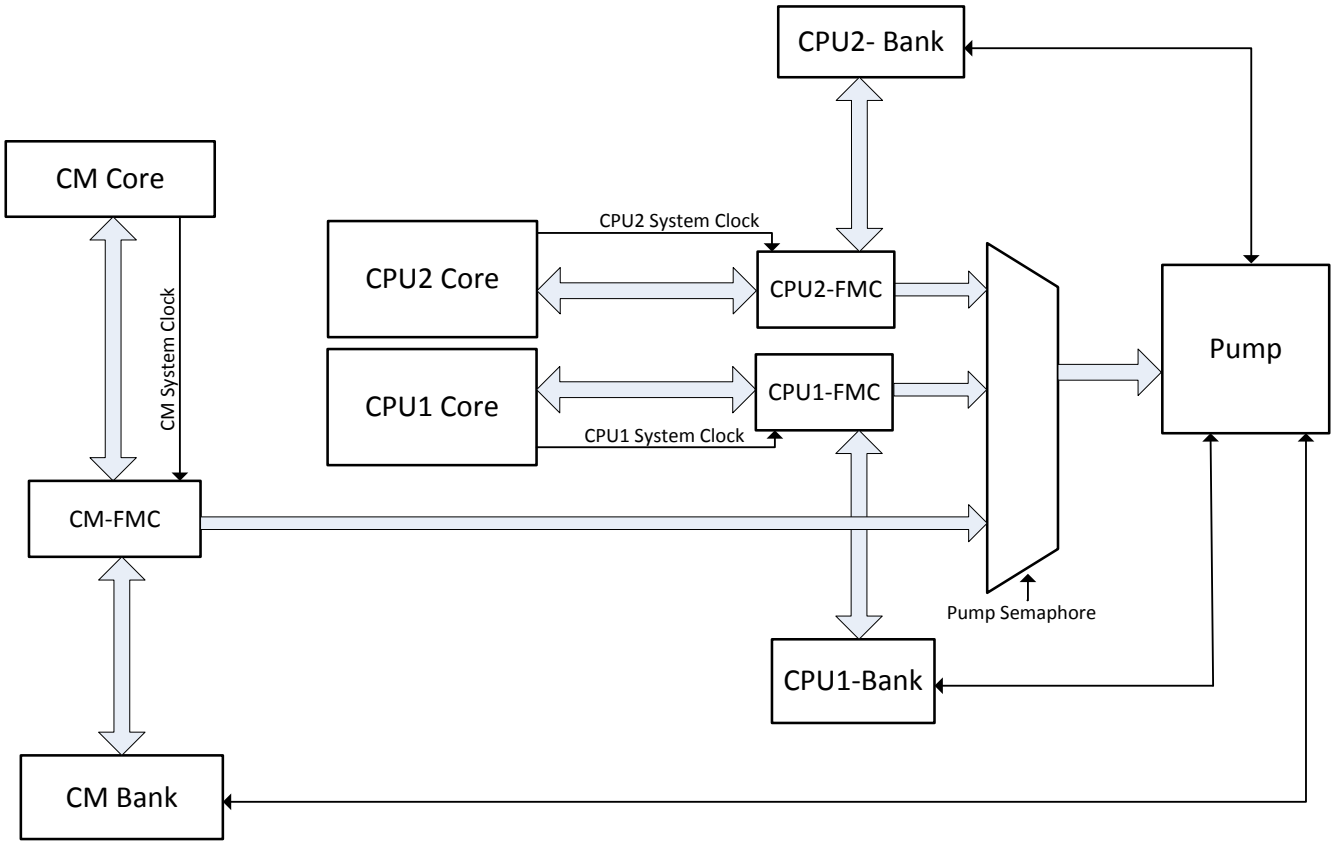


Figure 13-1. FMC Interface with Core, Bank, and Pump

The CPU2 in the CPU2 subsystem interfaces with the CPU2 Flash module controller (CPU2-FMC) that interfaces with the CPU2 Flash bank and shared pump to perform erase/program operations as well as to read data/execute code from the CPU2 Flash bank.

The CM in the CM subsystem interfaces with the CM Flash module controller (CM-FMC) that interfaces with the CM Flash bank and shared pump to perform erase/program operations as well as to read data/execute code from the CM Flash bank. Control signals to the Flash pump are controlled by either CPU1-FMC or CPU2-FMC or CM-FMC, depending on who gains the Flash pump semaphore.

13.4 Flash and OTP Memory Power-Down Modes and Wakeup

The Flash bank and pump consume a significant amount of power when active. The Flash module provides a mechanism to power-down Flash banks and pump. Special timers automatically sequence the power-up of the CPU1 Flash bank, CPU2 Flash bank, and CM Flash bank independently of each other. The shared charge pump module includes an independent power-up timer as well.

The Flash bank and OTP memory operate in three power modes: Sleep (lowest power), Standby, and Active (highest power)

- Sleep State
  This is the state after a device reset. In this state, a CPU data read or opcode fetch automatically initiates a change in power mode to the standby state and then to the active state. During this transition time to the active state, the CPU execution is automatically stalled.
- Standby State
  This state uses more power than the sleep state, but takes a shorter time to transition to the active or read state. In this state, a CPU data read or opcode fetch automatically initiates a change in power mode to the active state. During this transition time to the active state, the CPU execution is automatically stalled. Once the Flash/OTP has reached the active state, the CPU access completes as normal.
- Active or Read State
  In this state, the bank and pump are in active power mode state (highest power)

The charge pump operates in two power modes:

- Sleep (lowest power)
- Active (highest power)

Any access to any Flash bank/OTP causes the charge pump to go into active mode, if in sleep mode. An erase or program command causes the charge pump and bank to become active. If any bank is in active or in standby mode, the charge pump is in active mode, independent of the pump power mode control configuration (PMPPWR bit field in the FPAC1 register).

To power down the Flash pump, all three cores (CPU1, CPU2, and CM) must each power down the Flash Pump without any Flash accesses in between. The Flash Pump does not enter low-power mode if the following sequence of operations is not executed.

1. When the system is ready to power down the Flash completely, synchronize CPU1, CPU2, and CM. CM executes the Flash power-down phase (steps 2, 3, and 4) while CPU1 and CPU2 wait for CM to complete the Flash power-down sequence.
2. Acquire the Pump Semaphore with the CM.
3. Change the CM Flash Bank Fall Back power mode to Sleep: FBFALLBACK.BNKPWR = 0.
4. Change the CM Flash Charge Pump Fall Back power-mode to Sleep: FPAC1.PMPPWR = 0. Using IPC, notify CPU1 and CPU2 that the CM has completed the above sequence. After sending the notification, wait until CPU1 and CPU2 have completed steps 5 to 11, and an IPC notification has been received to confirm that all cores have completed the power-down sequence.
5. Acquire the Pump Semaphore with the CPU2.
6. Change the CPU2 Flash Bank Fall Back power mode to Sleep: FBFALLBACK.BNKPWR = 0.
7. Change the CPU2 Flash Charge Pump Fall Back power mode to Sleep: FPAC1.PMPPWR = 0.
8. Release the Pump Semaphore from the CPU2. Using IPC, notify CPU1 that CPU2 has completed the power-down sequence. After sending the notification, wait until CPU1 completes steps 9, 10 and 11.
9. Acquire the Pump Semaphore with the CPU1.
10. Change the CPU1 Flash Bank Fall Back power mode to Sleep: FBFALLBACK.BNKPWR = 0.
11. Change the CPU1 Flash Charge Pump Fall Back power mode to Sleep: FPAC1.PMPPWR = 0.
12. Release the Pump Semaphore from the CPU1. Using IPC, notify CPU2 and CM that CPU1 has completed the power-down sequence so that all three (CPU1, CPU2, and CM) subsystems can continue.

The previously described procedure can only be executed from RAM, not from Flash.

Note

Before configuring FPAC1.PMPPWR as described above, be sure to gain exclusive control of the Flash pump by using PUMPREQUEST register. Since the charge pump is shared between CPU1-FMC, CPU2-FMC, and CM-FMC, the effective PMPPWR value used when powering down the pump is the FMC (out of these three) that currently owns the pump. The application code can check the current power mode of the Flash bank by reading the FBPRDY register. The effective power mode of the charge pump is a logical OR of the FBPRDY.PUMPRDY register bits in CPU1-FMC, CPU2-FMC, and CM-FMC. A value of 0 in the PUMPRDY bit in all three FMCs indicates that the charge pump is in sleep mode. If the value of FBPRDY.PUMPRDY in any of the FMCs is 1, then the charge pump is in active mode. For more details, see the Flash register descriptions in Section 13.15.

While the pump is in sleep state, a charge pump sleep down counter holds a user configurable value (PSLEEP bit field in the FPAC1 register). When the charge pump exits sleep mode, the device waits for PSLEEP pre-scaled SYSCLK/CMCLK clock cycles (pre-scaled clock is SYSCLK/2 for CPU1 and CPU2, and is CMCLK/2 for CM) before putting the charge pump into active power mode. The configured PSLEEP value must result in a minimum delay of 20 µs before the pump enters active mode. For more details, see the Flash register descriptions in Section 13.15.

Following are the cycle counts required for the bank and pump to wake up from low-power modes.

1. Pump sleep to active = PSLEEP * (SYSCLK or CMCLK)/2 cycles
2. Bank sleep to standby = 425 Flash clock cycles
3. Bank standby to active = 90 Flash clock cycles

Where: Flash clock = (SYSCLK or CMCLK)/(RWAIT+1)

13.5 Active Grace Period

The active grace period (AGP) can be used to optimize the Flash module power consumption versus access time. Faster access times are associated with higher-power modes of operation. At one extreme, the power control logic can attempt to reduce power consumption by putting the bank and charge pump into a low-power mode immediately at the end of every Flash access. However, if accesses are only a few cycles apart, this can actually increase power consumption versus leaving the Flash powered, because the bank and charge pump consume more power during Flash startup and access.

The active grace periods allows the bank and charge pump to be maintained in active mode for a specified period following an access. This is done in anticipation of another read within the AGP time, to allow the subsequent read to have a faster access and spend less time dissipating power, than if the bank went into one of the low-power modes immediately. If the next access does not occur within the AGP time, the power control logic can automatically put the bank and charge pump into a low-power mode to reduce power consumption during long periods of inactivity.

The AGP value is programmed by a set of programmable counters (FBAC and FPAC2) that keep the Flash bank or charge pump in active mode until the counter expires, at which time the bank or charge pump reverts to the fallback power mode as defined in the FBFALLBACK and FPAC1 (refer to PMPPWR bit-field) registers. The application software can configure the fallback power mode to reduce power consumption, or configure the power mode to be active mode to keep the bank active regardless of counter settings (default is SLEEP). The charge pump AGP counter remains in the initialized state when the bank is active, including the AGP counter of the bank. The charge pump AGP counter begins counting when the bank has become inactive.

The application software can check the current power mode of Flash bank and charge pump by reading the FBPRDY register.

13.6 Flash and OTP Memory Performance

Flash read or instruction fetch accesses can be classified either as a Flash access (access to an address location in Flash), or an OTP access (access to an address location in OTP).

When the CPU performs an access to a Flash memory address, data is returned after (RWAIT+1) SYSCLK/CMCLK cycles.

For a USER-OTP access, data is returned after 11 SYSCLK cycles.

For a USER-OTP access, data is returned after 11 SYSCLK cycles for CPU1, RWAIT+2 SYSCLK cycles for CPU2, and RWAIT+2 CMCLK cycles for CM.

RWAIT defines the number of random access wait states, and is configured using the RWAIT field in the FRDCNTL register. At reset, RWAIT defaults to a worst-case wait state count (15), and therefore must be initialized to the appropriate number of wait states to improve performance, based on the CPU clock frequency and the access time of the Flash. The Flash supports zero-wait accesses when RWAIT is set to zero, when the CPU clock frequency is low enough to accommodate the Flash access time.

For a given system clock frequency, configure RWAIT using the following formula:

For C29x Flash Bank: RWAIT = ceiling[(SYSCLK/FCLK)-1]

For CM Flash Bank: RWAIT = ceiling[(CMCLK/FCLK)-1]

where SYSCLK is the system operating frequency for CPU1 and CPU2, where CMCLK is the system operating frequency for CM, and FCLK is the clock frequency for Flash.

FCLK must be ≤ $FCLK_{max}$, the allowed maximum Flash clock frequency at RWAIT=0.

If RWAIT results in a fractional value when calculated using the above formula, round up RWAIT to the nearest integer.

13.7 Flash Read Interface

This section provides details about the data read modes to access Flash bank/OTP and the configuration registers that control the read interface. In addition to a standard read mode, the FMC has a built-in prefetch and cache mechanism to allow increased clock speeds and CPU throughput wherever applicable.

13.7.1 C28x-FMC (CPU1-FMC and CPU2-FMC) Flash Read Interface

13.7.1.1 Standard Read Mode

Standard read mode is the default Flash read mode after reset. In this mode, the code prefetch mechanism and data cache are disabled. When standard read mode is active, every read access to Flash is decoded by the Flash wrapper to fetch the data from the addressed location, and the data is returned after RWAIT+1 cycles (except User OTP).

Flash data buffers associated with the prefetch mechanism and data cache are bypassed in standard read mode; therefore, every access to the Flash/OTP is used by the CPU immediately, and every access creates a unique Flash bank access.

Standard read mode is the recommended mode for lower system frequency operation, where RWAIT can be set to zero to provide single-cycle access operation. The FMC can operate at higher frequencies using standard read mode, at the expense of adding wait states. At higher system frequencies, it is recommended to enable the data cache and prefetch mechanisms to improve performance. Refer to the device data sheet to determine the maximum Flash frequency allowed in standard read mode (that is, maximum Flash clock frequency with RWAIT=0, $FCLK_{MAX}$).

13.7.1.2 Prefetch Mode

Flash memory is typically used to store application code. During code execution, instructions are fetched from contiguous memory addresses, except when a discontinuity occurs. Usually, the portion of the code that resides in contiguous address locations makes up the majority of the application code, and is referred to as linear code. To improve the performance of linear code execution, a Flash prefetch mechanism has been implemented. Figure 13-2 illustrates how this mode functions.

The prefetch mechanism does a look-ahead prefetch on linear address increments, starting from the address of the last instruction fetch. The Flash prefetch mechanism is disabled by default. To enable prefetch mode, set the PREFETCH_EN bit in the FRD_INTF_CTRL register, or call the `Flash_enablePrefetch()` driverlib function.

Each instruction fetch from the Flash or OTP reads out 128 bits. The starting address of the access from Flash is automatically aligned to a 128-bit boundary, such that the instruction location is within the 128 bits to be fetched. When Flash prefetch mode is enabled, the 128 bits read from the instruction fetch are stored in a 128-bit wide by 2-level deep instruction prefetch buffer. The contents of this prefetch buffer are then sent to the CPU for processing as required.

Up to four 32-bit or eight 16-bit instructions can reside within a single 128-bit access. The majority of C28x instructions are 16 bits, so for every 128-bit instruction fetch from the Flash bank, it is likely that there are up to eight instructions in the prefetch buffer ready to process through the CPU. During the time it takes to process these instructions, the Flash prefetch mechanism automatically initiates another access to the Flash bank to prefetch the next 128 bits. In this manner, the Flash prefetch mechanism works in the background to keep the instruction prefetch buffers as full as possible. Using this technique, the overall efficiency of sequential code execution from Flash or OTP is improved significantly.

Note

If the prefetch mechanism is enabled, then the last two rows (16 16-bit words, 256 bits) of a Flash bank can not be used if there are no valid Flash memory addresses beyond the bank boundary. This is necessary to prevent the Flash prefetch mechanism from attempting to pre-load the instruction buffer with data from invalid address locations, causing an ECC error.

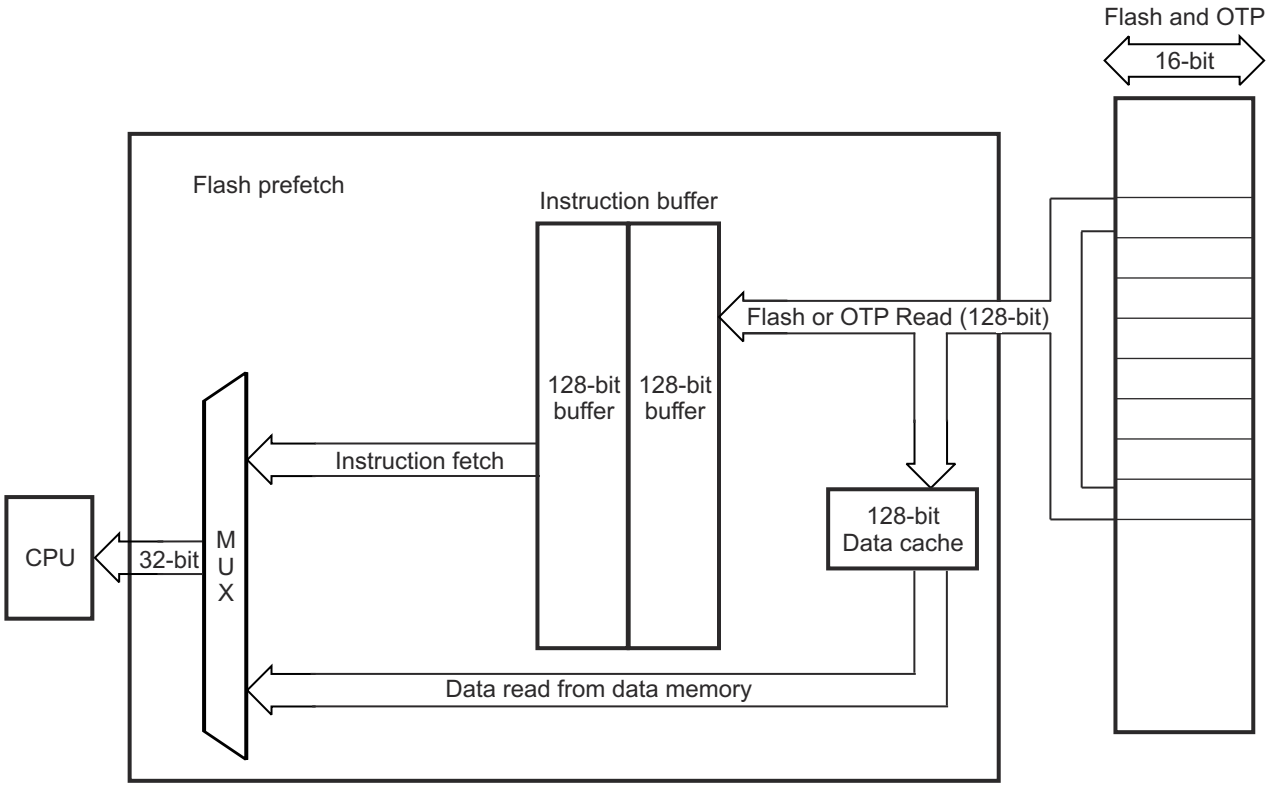


Figure 13-2. Flash Prefetch Mode

The Flash prefetch is aborted only when there is a code discontinuity caused by executing an instruction such as a branch, function call, or loop. When this occurs, the prefetch mechanism is aborted, and the contents of the prefetch buffer are flushed. There are two possible scenarios when this occurs:

1. If the destination address is within the Flash or OTP, the prefetch aborts and then resumes at the destination address.
2. If the destination address is outside of the Flash and OTP, the prefetch is aborted, and begins again only when the code branches back into the Flash or OTP. The Flash prefetch mechanism only applies to instruction fetches from program space. Data reads from data memory and from program memory do not utilize the prefetch mechanism and thus bypass the prefetch buffer. For example, instructions such as MAC, DMAC, and PREAD read a data value from program memory. When such a read happens, the prefetch buffer is bypassed, but the buffer is not flushed. If an instruction prefetch is already in progress when a data read operation is initiated, then the data read is stalled until the prefetch completes.

Note that the prefetch mechanism gets bypassed when RWAIT is configured as zero.

13.7.1.2.1 Data Cache

In addition to the prefetch mechanism, a data cache of 128-bits wide has been implemented to improve data space read performance. This data cache is separate from the instruction prefetch buffer, and is used for data reads only. Whenever a data read access is performed by the CPU to a Flash bank address, if the data located at that address is not presently loaded into the data cache, then the Flash wrapper reads 128 bits of data from the Flash bank and stores it in the data cache. This data is eventually sent to the CPU for processing. The starting address of the Flash bank access is automatically aligned to a 128-bit boundary, such that the requested address location is within the 128 bits to be read from the bank.

The data cache is disabled by default at reset. To enable the data cache, set the DATA_CACHE_EN bit in the FRD_INTF_CTRL register, or call the `Flash_enableCache()` driverlib function. Note that the data cache gets bypassed when RWAIT is set to zero.

Note

The data cache does not get updated on a debugger access, or when a read to the ECC memory-mapped region is performed.

13.7.2 M4-FMC (CM-FMC) Flash Read Interface

13.7.2.1 Standard Read Mode

Standard read mode is defined as the read mode in effect when program cache/prefetch-mechanism and data cache are disabled. Standard read mode is also the default read mode after reset. During this mode, each read access to Flash is decoded by the Flash wrapper to fetch the data from the addressed location and the data is returned after the RWAIT+1 number of cycles.

The program cache/prefetch mecahnism and data cache are bypassed in standard read mode; therefore, every access to the Flash/OTP is used by the CPU immediately and every access creates a unique Flash bank access.

Standard read mode is the recommended mode for lower system frequency operation in which RWAIT can be set to zero to provide single cycle access operation.

FMC can operate at higher frequencies using standard read mode at the expense of adding wait states. At higher system frequencies, it is recommended to enable cache and prefetch mechanisms to improve performance. Refer to the data manual to determine the maximum Flash frequency allowed in standard read mode (that is, the maximum Flash clock frequency with one wait state - FCLKmax).

13.7.2.2 Cache Mode

13.7.2.2.1 Program Cache

Flash memory is typically used to store application code. During code execution, instructions are fetched from sequential memory addresses, except when a discontinuity occurs. Usually, the portion of the code that resides in sequential addresses makes up the majority of the application code, and is referred to as linear code. To improve the performance of linear code execution, the FMC includes a Flash prefetch mechanism. This prefetch mechanism does a look-ahead prefetch on linear address increments starting from the address of the last program access.

Apart from linear code, in general application code, there can be several loops wherein a set of instructions located in sequential addresses are executed repeatedly in a loop, until a condition holds true. To improve the performance of small loop code execution, an 8-level deep 128-bit wide (8 x 128) direct mapped program cache has been implemented in the FMC. Whenever instructions in the cache are fetched for CPU processing, the Flash prefetch mechanism does a look-ahead prefetch of 128 bits from the next linear 128-bit aligned address from last address access, and fills the program cache as shown in Figure 13-3.

Up to four 32-bit instructions or up to eight 16-bit instructions can reside within a single 128-bit access. For every 128-bit instruction fetch from the Flash bank, it is likely that there are up to eight instructions (assuming 16-bit instructions) in each level of cache, ready to process through the CPU. During the time it takes to process these instructions, the Flash prefetch mechanism automatically initiates another access to the Flash bank to prefetch the next 128 bits. In this manner, the Flash prefetch mechanism works in the background to keep the cache as full as possible with data corresponding to the next linear address. Using this technique, the overall efficiency of sequential code execution and small loop code execution from Flash or OTP is improved significantly.

The Flash program cache and prefetch mechanism features are disabled by default. Setting the PROG_CACHE_EN bit in the FRD_INTF_CTRL register enables this cache mode. The Flash prefetch and cache mechanisms are independent of the CPU pipeline.

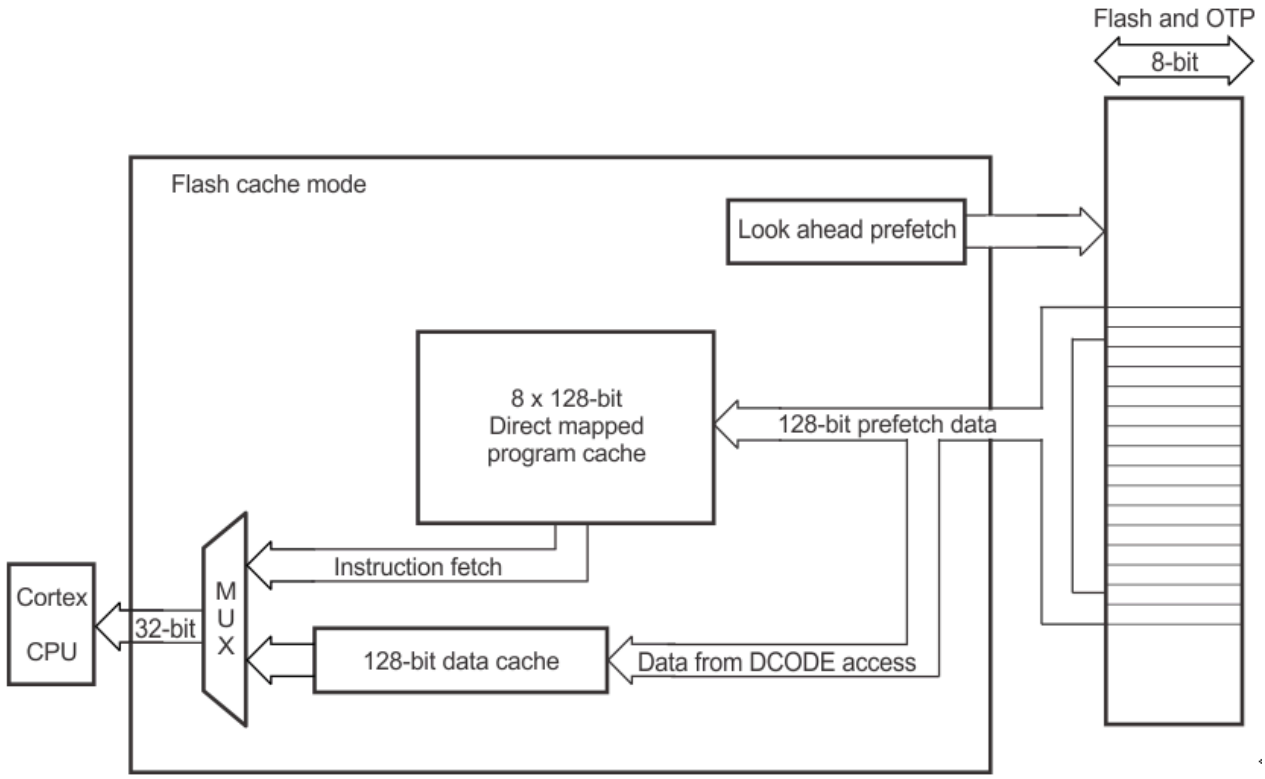


Figure 13-3. Flash Cache Mode

When the Cortex®-M4 CPU (CM) initiates an ICODE access from an address in program space in Flash:

1. The M4-FMC (CM-FMC) checks the program cache, if enabled, to determine the presence of required data.
2. If data corresponding to the requested address is present in the cache, the data is construed as a cache hit, and data is provided to the CPU from cache. At this time, the prefetch mechanism fetches 128 bits from the next linear 128-bit aligned address from last address access, and fills the program cache.
3. If data corresponding to the requested address is not present in the cache, the data is construed as a cache miss. Therefore, the prefetch mechanism fetches 128 bits of the data from the requested address in bank (the starting address of the access from Flash is automatically aligned to a 128-bit boundary such that the instruction location is within the 128 bits to be fetched) and writes that data on to cache, and eventually the requested data is sent to the CPU from cache for processing.

When cache mode is enabled, avoid placing code in the last row of 128 bits in the Flash bank. Doing so can result in an ECC error caused by a look-ahead prefetch access to an address past the bank address boundary.

The Flash prefetch mechanism is aborted only on a PC discontinuity caused by executing an instruction such as a branch, function call, or loop, and so on. When this occurs, the prefetch is aborted. There are two possible scenarios when this occurs:

- If the destination address is within the Flash or OTP, then prefetch aborts and then resumes at the destination address.
- If the destination address is outside of the Flash and OTP, the prefetch is aborted and begins again only when a branch is made back into the Flash or OTP. The Flash prefetch mechanism only applies to instruction fetches (ICODE) from program space. Data space accesses (DCODE) do not utilize the prefetch mechanism. For data space accesses, the program cache and prefetch mechanism are bypassed. If an instruction prefetch is already in progress when a data read operation is initiated, then the data read stalls until the prefetch completes.

13.7.2.2.2 Data Cache

Along with the program cache, a data cache of 128-bits width is also implemented to improve data space access (DCODE) performance. This data cache is not filled by the prefetch mechanism. When any kind of data space access is made by the CPU from an address in the bank, and if the data corresponding to the requested address is not in the data cache, then 128 bits of data are read from the bank and loaded in the data cache. The data is eventually sent to the CPU for processing. The starting address of the access from Flash is automatically aligned to a 128-bit boundary such that the requested address location is within the 128 bits to be read from the bank. By default, this data cache is disabled and can be enabled by setting DATA_CACHE_EN bit in the FRD_INTF_CTRL register.

Some other points to keep in mind when working with M3 Flash:

- Reads of the CSM password locations, GRABRAM, GRABSECT, ECSLKEY and EXEONLY locations are hardwired for 10 wait states. The RWAIT bits have no effect on these locations.
- CPU writes to the Flash or OTP memory map areas are ignored and complete in a single cycle.
- DCODE access to zone-1 and zone-2 password locations return the data to Code Security Module (CSM) and return a value of zero to Cortex®-M4 (CM) when the respective password lock bits are not all 1s and the respective zone is not unlocked.
- ICODE accesses to zone-1 and zone-2 password locations return a 0 when the respective password lock bits are not all 1s and the respective zone is not unlocked.
- When the Code Security Module (CSM) is secured, reads to the Flash/OTP memory map area from outside the secure zone take the same number of cycles as a normal access. However, the read operation returns a zero.
- The arbitration scheme in M3-FMC prioritizes Cortex®-M4 (CM) accesses in the fixed priority order of data space access (DCODE), program space access (ICODE)/program space prefetch.
- When FSM interface is active for erase/program operations, data in the program cache and data cache in FMC is invalidated.

13.8 Flash Erase and Program

Flash memory can be programmed either by using the CCS Flash plug-in or by using the UniFlash application. If these methods are not feasible in an application, the Flash API can be used. The Flash memory can be programmed, erased, and verified only by using the Flash API library. These functions are written, compiled and validated by Texas Instruments. The Flash module contains a Flash state machine (FSM) to perform program and erase operations.

The recommended flow for programming Flash is:
Erase → Program → Verify

13.8.1 Erase

When the target Flash is erased, the Flash reads as all 1's. This state is called 'blank.' The erase function must be executed before programming. The user can not skip erase on sectors that read as 'blank' because these sectors can require additional erasing due to marginally erased bits columns. The FSM provides an Erase Sector command to erase the target sector. The erase function erases the data and the ECC together.

13.8.2 Program

The FSM provides a command to program the and Flash. This command is also used to program ECC check bits.

Note

The main array Flash programming must be aligned to 64-bit address boundaries, and each 64-bit word can only be programmed once per write/erase cycle.

The DCSM OTP programming must be aligned to 128-bit address boundaries and each 128-bit word can only be programmed once. The exceptions are:

- The DCSM Zx-LINKPOINTER1 and Zx-LINKPOINTER2 values in the DCSM OTP can be programmed together and can be programmed one bit at a time as required by the DCSM operation.
- The DCSM Zx-LINKPOINTER3 values in the DCSM OTP can be programmed one bit at a time as required by the DCSM operation.

13.8.3 Verify

After programming, the user must perform verify using API function Fapi_doVerify(). This function verifies the Flash contents against supplied data.

Application software typically perform a CRC check of the Flash memory contents during power-up and at regular intervals during runtime (as needed). Apart from this, ECC logic, when enabled (enabled by default), catches single-bit errors, double-bit errors, and address errors whenever the CPU reads/fetches from a Flash address.

13.9 Error Correction Code (ECC) Protection

CPU1-FMC and CPU2-FMC contain an embedded single error correction and double error detection (SECDED) module. SECDED, when enabled, provides the capability to screen out memory faults. SECDED can detect and correct single-bit data errors and detect address errors/double-bit data errors. For every 64 bits of Flash/OTP data (aligned on a 64-bit memory boundary) that is programmed, eight ECC check bits have to be calculated and programmed in ECC memory space. Refer to the device data sheet for the Flash/OTP ECC memory-map. SECDED works with a total of eight user-calculated error correction code (ECC) check bits associated with each 64-bit wide data word and the corresponding 128-bit memory-aligned address. Users must program ECC check bits along with Flash data. TI recommends using the AutoEccGeneration option available in the Plugin/API to program ECC. Users can use the Flash API to calculate and program ECC data along with Flash data. Flash API uses hardware ECC logic in the device to generate the ECC data for the given Flash data. The Flash Plugin, the Flash programming tool integrated with the Code Composer Studio™ IDE, uses the Flash API to generate and program ECC data.

Figure 13-4 illustrates the ECC logic inputs and outputs.

During an instruction fetch or a data read operation, the 19 most significant address bits (three least significant bits of address are not considered), together with the 64-bit data/8-bit ECC read-out of Flash banks/ECC memory map area, pass through the SECDED logic and the eight checkbits are produced in FMC. These eight calculated ECC check bits are then XORed with the stored check bits (user programmed check bits) associated with the address and the read data. The 8-bit output is decoded inside the SECDED module to determine one of three conditions:

- No error occurred
- A correctable error (single bit data error) occurred
- A non-correctable error (double bit data error or address error) occurred

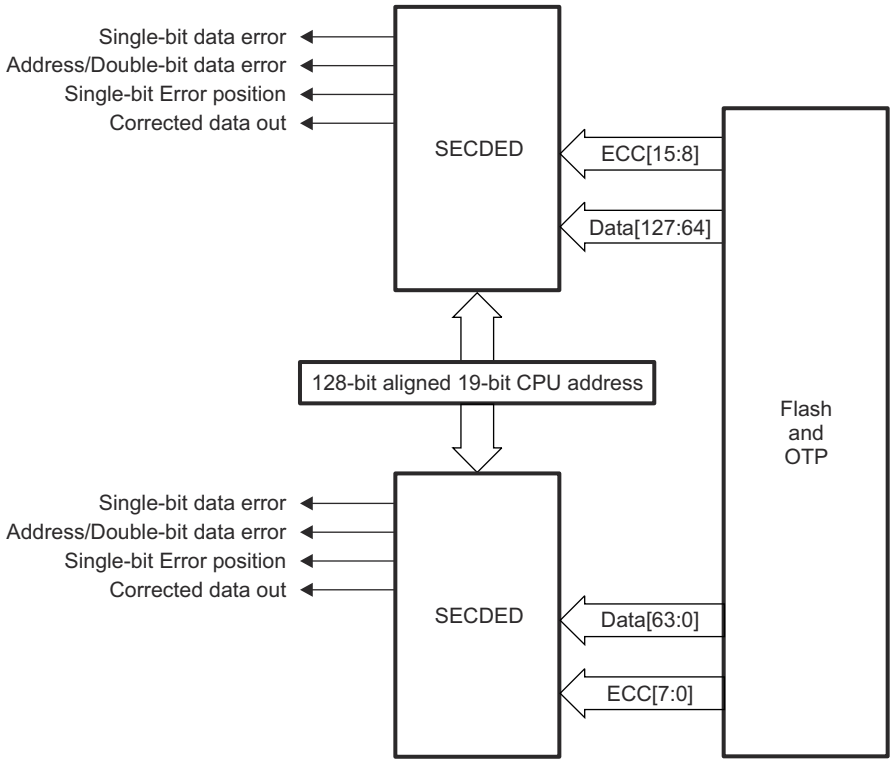


Figure 13-4. ECC Logic Inputs and Outputs

If the SECDED logic finds a single-bit error in the address field, then the error is considered to be a non-correctable error.

Note

Since ECC is calculated for an entire 64-bit data, a non 64-bit read such as a byte read or a half-word read still forces the entire 64-bit data to be read and calculated, but only the byte or half-word are actually used by the CPU.

This ECC (SECDED) feature is enabled at reset. The ECC_ENABLE register can be used to configure (enable/disable) the ECC feature. The ECC for the application code must be programmed. There are two SECDED modules in each FMC. Out of the 128-bit data (aligned on a 128-bit memory boundary) read from the bank/OTP address, the lower 64 bits of data and corresponding 8 ECC bits (read from user programmable ECC memory area) are fed as inputs to one SECDED module along with 128-bit aligned 19-bit address from where data has been read. The upper 64 bits of data and corresponding 8 ECC bits are fed as inputs to another SECDED module in parallel, along with 128-bit aligned 19-bit address. Each of the SECDED modules evaluate their inputs and determine if there is any single-bit data error or double-bit data error/address error.

ECC logic is bypassed when the 64 data bits and the associated ECC bits fetched from the bank are either all ones or zeros.

13.9.1 Single-Bit Data Error

This section provides information for both single-bit data errors and single-bit ECC check bit errors. If there is a single bit flip (0 to 1 or 1 to 0) in Flash data or in ECC data, then the error is considered as a single-bit data error. The SECDED module detects and corrects single-bit errors, if any, in the 64-bit Flash data or eight ECC check bits read from the Flash/ECC memory map before the read data is provided to the CPU.

When SECDED finds and corrects single bit data errors, the following information is logged in the ECC registers if the ECC feature is enabled:

- Address where the error occurred: if the single-bit error occurs in the lower 64 bits of a 128-bit memory-aligned Flash data word, the address of the lower 64-bit word is captured in the SINGLE_ERR_ADDR_LOW register. If the single-bit error occurs in the upper 64 bits of the 128-bit data word, then the address of the upper 64-bit word is captured in the SINGLE_ERR_ADDR_HIGH register.
- Whether the error occurred in data bits or ECC bits: the ERR_TYPE_L and ERR_TYPE_H bit fields in the ERR_POS register indicate whether the error occurred in data bits or ECC bits of the lower 64 bits, or the upper 64 bits respectively, of a 128-bit memory-aligned Flash data word.
- Bit position at which the error occurred: the ERR_POS_L and ERR_POS_H bit fields in the ERR_POS register indicate the bit position of the error in the lower 64 bits/lower 8-bit ECC, or the upper 64 bits/upper 8-bit ECC respectively, of a 128-bit memory-aligned Flash data word.
- Whether the corrected value is 0 (FAIL_0_L, FAIL_0_H flags in ERR_STATUS register).
- Whether the corrected value is 1 (FAIL_1_L, FAIL_1_H flags in ERR_STATUS register).
- A single bit error counter that increments on every single bit error occurrence (ERR_CNT register) until a user-configurable threshold (see ERR_THRESHOLD) is met.
- A flag that gets set when one or more single-bit errors occurs after ERR_CNT equals ERR_THRESHOLD (SINGLE_ERR_INT_FLG flag in the ERR_INTFLG register).

When the ERR_CNT value equals ERR_THRESHOLD+1, and a single bit error occurs, the Flash module sets the SINGLE_ERR_INT flag and generates an interrupt signal. To enable propagation of the generated interrupt pulse to the CPU, the user application must enable the FLASH_CORRECTABLE_ERROR channel in the C28 Peripheral Interrupt Expansion module (PIE). The interrupt signal remains high until the application clears the SINGLE_ERR_INTFLG flag by writing to the SINGLE_ERR_INTCLR bit in the ERR_INTCLR register. The Flash module cannot generate any further FLASH_CORRECTABLE_ERROR interrupt signals to the PIE/CPU until SINGLE_ERR_INTFLG is cleared, as this is an edge-based interrupt.

When multiple single-bit errors have been detected by ECC logic, the contents of the Flash ECC registers reflect the most recent ECC error. When multiple single-bit errors have been detected, both FAIL_0_L and FAIL_1_L

(or FAIL_0_H and FAIL_1_H) can be set, indicating that single-bit fail0/fail1 occurred in different 64-bit aligned addresses.

Although ECC is calculated on 64-bit basis, a read of any address location within a 128-bit aligned Flash data word causes the single-bit error flag to get set, if there is a single-bit error in both or in either the lower 64 or upper 64 bits (or corresponding ECC check bits) of that 128-bit data word.

13.9.2 Uncorrectable Error

Uncorrectable errors include address errors and double-bit errors in data or ECC. When the ECC logic finds uncorrectable errors, the following information is logged in ECC registers if the ECC feature is enabled:

- Address where the error occurred: if the uncorrectable error occurs in the lower 64 bits of a 128-bit memory-aligned Flash data word, the lower 64-bit memory-aligned address is captured in the UNC_ERR_ADDR_LOW register. If the uncorrectable error occurs in the upper 64 bits of a 128-bit memory-aligned Flash data word, the upper 64-bit memory-aligned address is captured in the UNC_ERR_ADDR_HIGH register.
- A flag is set indicating that an uncorrectable error occurred – the UNC_ERR_L and UNC_ERR_H flags in the ERR_STATUS register indicate the uncorrectable error occurrence in the lower 64 bits/lower 8-bit ECC, or the upper 64 bits/upper 8-bit ECC, respectively, of a 128-bit memory-aligned Flash data word.
- A flag is set indicating that an uncorrectable error interrupt is generated (UNC_ERR_INTFLG in ERR_INTFLG register).

When an uncorrectable error occurs, the Flash module sets the UNC_ERR_INTFLG bit and generates an uncorrectable error interrupt. This uncorrectable error interrupt generates a non-maskable interrupt (NMI), if enabled, in the CPU. If an uncorrectable error interrupt flag is not cleared by writing to the UNC_ERR_INTCLR bit in the ERR_INTCLR register, the Flash module cannot generate new uncorrectable interrupt signals, as this is an edge-based interrupt.

Although ECC is calculated on 64-bit basis, a read of any address location within a 128-bit aligned Flash word causes the uncorrectable error flag to get set, and an uncorrectable error interrupt/NMI to occur, when there is a uncorrectable error in both or in either the lower 64 bits or upper 64 bits (or corresponding ECC check bits) of that 128-bit data word.

13.9.3 SECDED Logic Correctness Check

Since error detection and correction logic are part of safety-critical logic, safety applications often need to make sure that the SECDED logic is always working properly. For these safety concerns, to confirm the correctness of the SECDED logic, an ECC test mode is provided to test the correctness of ECC logic periodically. In this ECC test mode, data/ECC and address inputs to the ECC logic are controlled by the ECC test mode registers FDATAH_TEST, FDATAL_TEST, FECC_TEST, and FADDR_TEST, respectively. Using this test mode, users can introduce single-bit errors, double-bit errors, or address errors and check whether or not SECDED logic is catching those errors. Users can also check if SECDED logic is reporting any false errors when no errors are introduced.

This ECC test mode can be enabled by setting the ECC_TEST_EN bit in the FECC_CTRL register. When ECC test mode is enabled, the CPU cannot read the data from Flash. Instead, the CPU gets data from the ECC test mode registers (FDATAH_TEST/FDATAL_TEST). This is because the ECC test mode registers (FDATAH_TEST, FDATAL_TEST, FECC_TEST) are multiplexed with data from the Flash. For this reason, ECC test mode code must be executed from RAM and not from Flash.

Only one of the SECDED modules (out of the two SECDED modules that work on lower 64 bits and upper 64 bits of a read 128-bit data) at a time can be tested. The ECC_SELECT bit in the FECC_CTRL register can be configured by users to select one of the SECDED modules for test.

To test the ECC logic using ECC test mode, follow these steps:

1. Obtain the ECC for a given Flash address (128-bit aligned) and 64-bit data by using the Auto ECC generation option provided in Flash API.
2. Develop an application to test ECC logic using the above data. In this application:
   - Write the 128-bit aligned 19-bit Flash address in FADDR_TEST.
   - Write 64-bit data in FDATAx_TEST (FDATAL_TEST and FDATAH_TEST) registers.
   - Write the corresponding 8-bit ECC in the FECC_TEST register.
   - In any of the above three steps, the application can insert errors (single-bit data error or double-bit data error or address error or single-bit ECC error or double-bit ECC error) to check whether or not ECC logic is able to detect the errors.
   - Select the ECC logic block (lower 64-bits or upper 64-bits) which needs to be tested using the ECC_SELECT bit in the FECC_CTRL register.
   - Enable ECC test mode using the ECC_TEST_EN bit in FECC_CTRL register.
   - Write a value of 1 in the DO_ECC_CALC bit in FECC_CTRL register to enable ECC test logic for a single cycle to evaluate the address, data, ECC in FADDR_TEST, FDATAx_TEST and FECC_TEST registers for ECC errors.

Once the above ECC test mode registers are written by the user:

- The FECC_OUTH register holds the data output bits 63:32 from the SECDED block under test.
- The FECC_OUTL register holds the data output bits 31:0 from the SECDED block under test.
- The FECC_STATUS register holds the status of single-bit error occurrence, uncorrectable error occurrence, and error position of single- bit error in data/check bits.

13.10 Reserved Locations Within Flash and OTP Memory

When allocating code and data to Flash and OTP memory, keep the following reserved locations in mind:

- The entire OTP has reserved user-configurable locations for security and boot process. For more details on the functionality of these fields, refer to Chapter 5 and Chapter 6.
- Refer to Chapter 5 for reserved locations in Flash for real-time operating system usage and a boot-to-Flash entry point. A boot-to-Flash entry point is reserved for an entry-into-Flash branch instruction. When the boot-to-Flash boot option is used, the boot ROM jumps to this address in Flash. If the user programs a branch instruction here, that then redirects code execution to the entry point of the application.

13.11 Migrating an Application from RAM to Flash

To migrate an existing application that is configured to run from RAM to a Flash-based linker configuration, follow the steps:

1. Replace the RAM linker command file with a Flash linker command file. For example Flash-based linker command files, see the device_support\<device>\common\cmd directory.
2. When modifying the Flash-based linker command file, be sure to map any initialized sections to Flash memory regions.
3. Make sure the boot mode pins are configured for Flash boot. This tells the boot ROM to redirect execution to the application programmed into Flash memory after boot code execution is complete. For more information on boot mode configuration, see *"Detailed Description Device Boot Modes"* in the device data sheet.
4. When the device is configured for Flash boot, the boot ROM redirects execution to the Flash entry point location (defined as BEGIN in TI-provided Flash linker command files) at the end of boot code execution. Make sure there is a branch instruction at the Flash entry point to your code initialization (for example, _c_int00) function. In the C2000Ware examples, the entry point code is specified in the CodeStartBranch.asm file.
5. To achieve best performance for Flash execution, configure the Flash wait states as per the device operating clock frequency, as specified in the device data manual. In addition, enable prefetch mode and data cache mode. Calling the Flash_initModule() driverlib function achieves these steps. Note that code that initializes the Flash module must execute from a RAM location. This is accomplished by assigning the Flash initialization function to the .TI.ramfunc section, and then copying the function to RAM at runtime using memcpy() before executing the function. In the linker command file, map this section to Flash for load, and RAM for execution.
6. For any functions that require 0- or 1-wait state performance, be sure to map to RAM for execution in the linker command file, similar to the Flash initialization function. The .TI.ramfunc section in the TI-provided Flash linker command files accomplishes this purpose.
7. Align all code and data sections to 128-bit address boundaries when mapping to Flash memory, using the ALIGN directive in the linker command file.
8. For EABI executable formats, all uninitialized sections mapped to RAM should be defined as NOINIT sections (using the directive "type=NOINIT") in the linker command file.
9. Be sure to program ECC bits correctly for the Flash application image. Keep the AutoEccGeneration option enabled in the Code Composer Studio Flash Plugin or UniFlash GUI.

13.12 Procedure to Change the Flash Control Registers

During Flash configuration, no accesses to the Flash or OTP can be in progress. This includes instructions still in the CPU pipeline, data reads, and instruction prefetch operations. To be sure that no access takes place during the configuration change, follow the procedure shown below for any code that modifies the Flash control registers.

1. Start executing application code from RAM/Flash/OTP.
2. Branch to or call the Flash configuration code (that writes to Flash control registers) in RAM. This is required to properly flush the CPU pipeline before the configuration change. The function that changes the Flash configuration cannot execute from the Flash or OTP and must reside in RAM.
3. Execute the Flash configuration code (located in RAM) that writes to Flash control registers like FRDCNTL, FRD_INTF_CTRL, and so on.
4. At the end of the Flash configuration code execution, wait eight cycles to let the write instructions propagate through the CPU pipeline. This must be done before the return-from-function call is made.
5. Return to the calling function that resides in RAM or Flash/OTP and continue execution.

13.13 Flash Pump Ownership Semaphore

Each CPU subsystem includes a dedicated Flash bank, which the subsystem can read, program, and erase. Both Flash banks share a single charge pump for program and erase operations. Hence, only one CPU can program or erase the Flash at any given time. A CPU can read data and execute code from the Flash even when the other CPU is programming or erasing. The Flash pump ownership semaphore allows one CPU to take control of the pump without being interrupted by the other CPU.

The pump ownership semaphore is implemented as a two-bit field in a PUMPREQUEST register with special write protections. This register requires a key field to be written at the same time as the semaphore bits. The possible semaphore states are:

PUMPREQUEST.SEM	Description
00	CPU1 has control of the pump, but CPU2 and CM can seize control at any time
01	CPU2 has exclusive control of the pump and of these semaphore bits. CPU2 can relinquish control by setting the bits back to 00
10	CPU1 has exclusive control of the pump and of these semaphore bits. CPU1 can relinquish control by setting the bits back to 00
11	CM has exclusive control of the pump and of these semaphore bits. CM can relinquish control by setting the bits back to 00

Semaphore bit changes from 01→10, 01→11, 10→01, 10→11, 11→01, or 11→10 are not allowed. The CPU that currently controls the Pump semaphore must first relinquish control by setting the bits to 00 before another CPU can assume ownership.

Figure 13-5 describes the allowed states and state transitions.

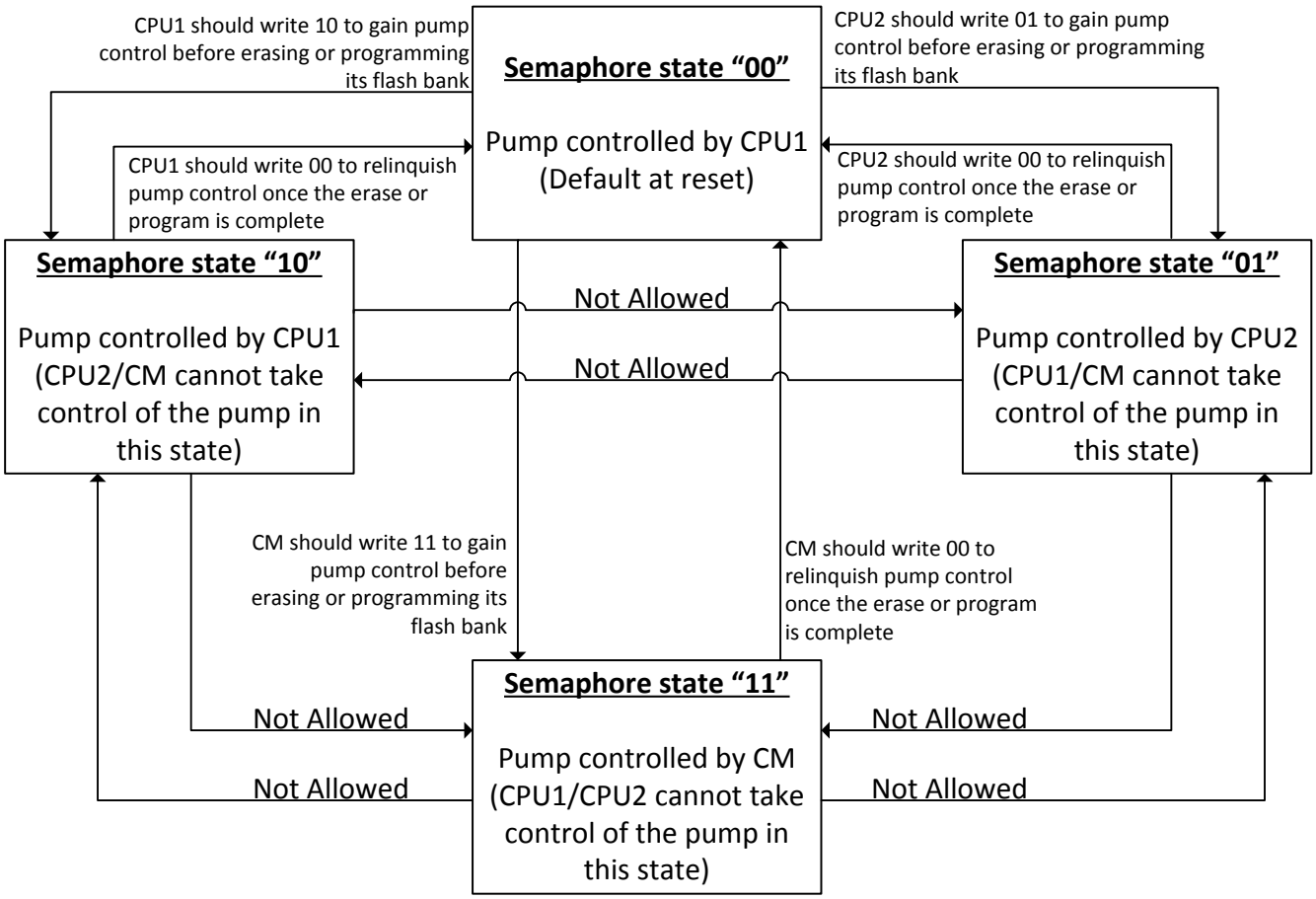


Figure 13-5. Flash Pump Semaphore (PUMPREQUEST) States and State Transitions

Note

Although there are multiple references and descriptions of PUMPREQUEST in this document, all of these reference the same register. There is only one PUMPREQUEST register in the device.

13.14 Software

13.14.1 FLASH Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/flash

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

13.14.1.1 Flash Programming with AutoECC, DataAndECC, DataOnly and EccOnly - CM

FILE: flashapi_cm_ex1_programming.c

This example demonstrates how to program Flash using API's following options
1. AutoEcc generation
2. DataOnly and EccOnly
3. DataAndECC

Before running this example, please run the cm_common_config_c28x Example from the c28x folder. It will initialize the clock, configure CPU1 Flash wait-states, fall back power mode, performance features and ECC.

External Connections
- None.

Watch Variables
- None.

13.14.1.2 Flash Programming with AutoECC, DataAndECC, DataOnly and EccOnly

FILE: flashapi_ex1_programming.c

This example demonstrates how to program Flash using API's following options
1. AutoEcc generation
2. DataOnly and EccOnly
3. DataAndECC

External Connections
- None.

Watch Variables
- None.

13.14.1.3 Flash ECC Test Mode

FILE: flash_ex2_ecc_test_mode.c

This example demonstrates ECC Test mode.

13.14.1.4 Flash ECC Test Mode - CM

FILE: flash_ex2_ecc_test_mode.c

This example demonstrates ECC Test mode.

13.15 Flash Registers

This section describes the Flash Module Registers.

13.15.1 FLASH Base Address Table (C28)

Table 13-1. FLASH Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
Flash0CtrlRegs	FLASH_CTRL_REGS	FLASH0CTRL_BASE	0x0005_F800	YES	YES	-	-	YES
Flash0EccRegs	FLASH_ECC_REGS	FLASH0ECC_BASE	0x0005_FB00	YES	YES	-	-	YES
FlashPumpSemaphoreRegs	FLASH_PUMP_SEMAPHORE_REGS	FLASHPUMPSEMAPHORE_BASE	0x0005_CE24	YES	YES	-	-	YES

13.15.2 CM FLASH Base Address Table (CM)

Table 13-2. CM FLASH Base Address Table (CM)

DriverLib Name	Base Address	µDMA Access	Ethernet DMA Access
FLASH0CTRL_BASE	0x400F_A000	-	-
FLASH0ECC_BASE	0x400F_A600	-	-
FLASHPUMPSEMAPHORE_BASE	0x400F_D048	-	-

13.15.3 FLASH_CTRL_REGS Registers

Table 13-3 lists the memory-mapped registers for the FLASH_CTRL_REGS registers. All register offset addresses not listed in Table 13-3 should be considered as reserved locations and the register contents should not be modified.

Table 13-3. FLASH_CTRL_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	FRDCNTL	Flash Read Control Register	EALLOW	Go
1Eh	FBAC	Flash Bank Access Control Register	EALLOW	Go
20h	FBFALLBACK	Flash Bank Fallback Power Register	EALLOW	Go
22h	FBPRDY	Flash Bank Pump Ready Register	EALLOW	Go
24h	FPAC1	Flash Pump Access Control Register 1	EALLOW	Go
2Ah	FMSTAT	Flash Module Status Register	EALLOW	Go
180h	FRD_INTF_CTRL	Flash Read Interface Control Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 13-4 shows the codes that are used for access types in this section.

Table 13-4. FLASH_CTRL_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

13.15.3.1 FRDCNTL Register (Offset = 0h) [Reset = 300h]

FRDCNTL is shown in Figure 13-6 and described in Table 13-5.

Return to the Summary Table.

Flash Read Control Register

Figure 13-6. FRDCNTL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				RWAIT				RESERVED							
R-0h				R/W-3h				R-0h							

Table 13-5. FRDCNTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-12	RESERVED	R	0h	Reserved
11-8	RWAIT	R/W	3h	Random read waitstate These bits indicate how many waitstates are added to a flash read access. The RWAIT value can be set anywhere from 0 to 0xF. For a flash access, data is returned in RWAIT+1 SYSCLK cycles. Note: The required wait states for each SYSCLK frequency can be found in the device data manual. Note: Following table summarizes the effect of RWAIT on accessing OTP space. There are secutity related overides to RWAIT behavior, refer to security spec for these details. RWAIT Value | Data returned after 0 | 1 cycle 1 to F | RWAIT + 2 cycles Reset type: SYSRSn
7-0	RESERVED	R	0h	Reserved

13.15.3.2 FBAC Register (Offset = 1Eh) [Reset = 15h]

FBAC is shown in Figure 13-7 and described in Table 13-6.

Return to the Summary Table.

Flash Bank Access Control Register

Figure 13-7. FBAC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																BAGP								RESERVED							
R-0h																R/W-0h								R/W-15h							

Table 13-6. FBAC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	BAGP	R/W	0h	Bank Active Grace Period. These bits contain the starting count value for the BAGP down counter. Any access to a given bank causes its BAGP counter to reload the BAGP value for that bank. After the last access to this flash bank, the down counter delays from 0 to 255 prescaled SYSCLK clock cycles before putting the bank into one of the fallback power modes as determined by the FBFALLBACK register. This value must be greater than 1 when the fallback mode is not ACTIVE. Note: The prescaled clock used for the BAGP down counter is a clock divided by 16 from input SYSCLK. Reset type: SYSRSn
7-0	RESERVED	R/W	15h	Reserved

13.15.3.3 FBFALLBACK Register (Offset = 20h) [Reset = 3h]

FBFALLBACK is shown in Figure 13-8 and described in Table 13-7.

Return to the Summary Table.

Flash Bank Fallback Power Register

Figure 13-8. FBFALLBACK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						BNKPWR0	
R-0h						R/W-3h	

Table 13-7. FBFALLBACK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-2	RESERVED	R	0h	Reserved
1-0	BNKPWR0	R/W	3h	Bank Power Mode Control 00 Sleep (Sense amplifiers and sense reference disabled) 01 Standby (Sense amplifiers disabled, but sense reference enabled) 10 Reserved 11 Active (Both sense amplifiers and sense reference enabled) Note: If the bank and pump are not in active mode and an access is made, the value of this register is automatically changed to active. Reset type: SYSRSn

13.15.3.4 FBPRDY Register (Offset = 22h) [Reset = 8001h]

FBPRDY is shown in Figure 13-9 and described in Table 13-8.

Return to the Summary Table.

Flash Bank Pump Ready Register

Figure 13-9. FBPRDY Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
PUMPRDY	RESERVED						
R-1h	R-0h						
7	6	5	4	3	2	1	0
RESERVED							BANKRDY
R-0h							R-1h

Table 13-8. FBPRDY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	PUMPRDY	R	1h	Pump Ready. This is a read-only bit which allows software to determine if the pump is ready for flash access before attempting the actual access. If an access is made to a bank when the pump is not ready, wait states are asserted until it becomes ready. 0 Pump is not ready. 1 Pump is ready, in active power state. Reset type: SYSRSn
14-1	RESERVED	R	0h	Reserved
0	BANKRDY	R	1h	Bank Ready. This is a read-only register which allows software to determine if the bank is ready for Flash access before the access is attempted. Note: The user should wait for both the pump and the bank to be ready before attempting an access. 0 Bank is not ready. 1 Bank is in active power mode and is ready for access. Reset type: SYSRSn

13.15.3.5 FPAC1 Register (Offset = 24h) [Reset = X]

FPAC1 is shown in Figure 13-10 and described in Table 13-9.

Return to the Summary Table.

Flash Pump Access Control Register 1

Figure 13-10. FPAC1 Register

31	30	29	28	27	26	25	24
RESERVED				PSLEEP			
R-0h				R/W-X			
23	22	21	20	19	18	17	16
PSLEEP							
R/W-X							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							PMPPWR
R-0h							R/W-1h

Table 13-9. FPAC1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-16	PSLEEP	R/W	X	Pump sleep. These bits contain the starting count value for the charge pump sleep down counter. While the charge pump is in sleep mode, the power mode management logic holds the charge pump sleep counter at this value. When the charge pump exits sleep power mode, the down counter delays from 0 to PSLEEP prescaled SYSCLK clock cycles before putting the charge pump into active power mode. Note: The pump sleep down counter uses the same prescaled clock as Bank sleep down counter which is divided by 2 of input SYSCLK. HW Reset Value for CPU1-FMC = 0xA0 SW Reset Value for CPU1-FMC = 0xA41 HW & SW Reset Value for CPU2-FMC= 0x834 Reset type: SYSRSn
15-1	RESERVED	R	0h	Reserved
0	PMPPWR	R/W	1h	Flash Charge Pump Fallback Power Mode. This bit selects what power mode the charge pump enters after the pump active grace period (PAGP) counter has timed out. 0 Sleep (all pump circuits disabled) 1 Active (all pump circuits active) Note for devices with multiple flash banks: As the pump is shared between flash banks, if an access is made either bank, the value of this bit changes to 1 (active). Reset type: SYSRSn

13.15.3.6 FMSTAT Register (Offset = 2Ah) [Reset = 0h]

FMSTAT is shown in Figure 13-11 and described in Table 13-10.

Return to the Summary Table.

Flash Module Status Register

Figure 13-11. FMSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						RESERVED	RESERVED
R-0h						R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	PGV	RESERVED	EV	RESERVED	Busy
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
ERS	PGM	INVDAT	CSTAT	VOLTSTAT	ESUSP	PSUSP	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 13-10. FMSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17	RESERVED	R	0h	Reserved
16	RESERVED	R	0h	Reserved
15	RESERVED	R	0h	Reserved
14	RESERVED	R	0h	Reserved
13	RESERVED	R	0h	Reserved
12	PGV	R	0h	When set, indicates that a word is not successfully programmed after the maximum allowed number of program pulses are given for program operation. Reset type: SYSRSn
11	RESERVED	R	0h	Reserved
10	EV	R	0h	When set, indicates that a sector is not successfully erased after the maximum allowed number of erase pulses are given for erase operation. During Erase verify command, this flag is set immediately if a bit is found to be 0. Reset type: SYSRSn
9	RESERVED	R	0h	Reserved
8	Busy	R	0h	When set, this bit indicates that a program, erase, or suspend operation is being processed. Reset type: SYSRSn
7	ERS	R	0h	When set, this bit indicates that the flash module is actively performing an erase operation. This bit is set when erasing starts and is cleared when erasing is complete. It is also cleared when the erase is suspended and set when the erase resumes. Reset type: SYSRSn
6	PGM	R	0h	When set, this bit indicates that the flash module is currently performing a program operation. This bit is set when programming starts and is cleared when programming is complete. It is also cleared when programming is suspended and set when programming is resumes. Reset type: SYSRSn

Table 13-10. FMSTAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	INVDAT	R	0h	When set, this bit indicates that the user attempted to program a 1 where a 0? was already present. This bit is cleared by the Clear Status command. Reset type: SYSRSn
4	CSTAT	R	0h	Once the FSM starts any failure will set this bit. When set, this bit informs the host that the program, erase, or validate sector command failed and the command was stopped. This bit is cleared by the Clear Status command. For some errors, this will be the only indication of an FSM error because the cause does not fall within the other error bit types. Reset type: SYSRSn
3	VOLTSTAT	R	0h	When set, this bit indicates that the core voltage generator of the pump power supply dipped below the lower limit allowable during a program or erase operation. This bit is cleared by the Clear Status command. Reset type: SYSRSn
2	ESUSP	R	0h	When set, this bit indicates that the flash module has received and processed an erase suspend operation. This bit remains set until the erase resume command has been issued or until the Clear_More command is run. Reset type: SYSRSn
1	PSUSP	R	0h	When set, this bit indicates that the flash module has received and processed a program suspend operation. This bit remains set until the program resume command has been issued or until the Clear_More command is run. Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

13.15.3.7 FRD_INTF_CTRL Register (Offset = 180h) [Reset = 0h]

FRD_INTF_CTRL is shown in Figure 13-12 and described in Table 13-11.

Return to the Summary Table.

Flash Read Interface Control Register

Figure 13-12. FRD_INTF_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						DATA_CACHE_EN	PREFETCH_EN
R-0h						R/W-0h	R/W-0h

Table 13-11. FRD_INTF_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-2	RESERVED	R	0h	Reserved
1	DATA_CACHE_EN	R/W	0h	Data cache enable. 0 A value of 0 disables the data cache. 1 A value of 1 enables the data cache. Reset type: SYSRSn
0	PREFETCH_EN	R/W	0h	Prefetch enable. 0 A value of 0 disables program cache and prefetch mechanism. 1 A value of 1 enables program cache and pre-fetch mechanism. Reset type: SYSRSn

13.15.4 FLASH_ECC_REGS Registers

Table 13-12 lists the memory-mapped registers for the FLASH_ECC_REGS registers. All register offset addresses not listed in Table 13-12 should be considered as reserved locations and the register contents should not be modified.

Table 13-12. FLASH_ECC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	ECC_ENABLE	ECC Enable	EALLOW	Go
2h	SINGLE_ERR_ADDR_LOW	Single Error Address Low	EALLOW	Go
4h	SINGLE_ERR_ADDR_HIGH	Single Error Address High	EALLOW	Go
6h	UNC_ERR_ADDR_LOW	Uncorrectable Error Address Low	EALLOW	Go
8h	UNC_ERR_ADDR_HIGH	Uncorrectable Error Address High	EALLOW	Go
Ah	ERR_STATUS	Error Status	EALLOW	Go
Ch	ERR_POS	Error Position	EALLOW	Go
Eh	ERR_STATUS_CLR	Error Status Clear	EALLOW	Go
10h	ERR_CNT	Error Control	EALLOW	Go
12h	ERR_THRESHOLD	Error Threshold	EALLOW	Go
14h	ERR_INTFLG	Error Interrupt Flag	EALLOW	Go
16h	ERR_INTCLR	Error Interrupt Flag Clear	EALLOW	Go
18h	FDATAH_TEST	Data High Test	EALLOW	Go
1Ah	FDATAL_TEST	Data Low Test	EALLOW	Go
1Ch	FADDR_TEST	ECC Test Address	EALLOW	Go
1Eh	FECC_TEST	ECC Test Address	EALLOW	Go
20h	FECC_CTRL	ECC Control	EALLOW	Go
22h	FOUTH_TEST	Test Data Out High	EALLOW	Go
24h	FOUTL_TEST	Test Data Out Low	EALLOW	Go
26h	FECC_STATUS	ECC Status	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 13-13 shows the codes that are used for access types in this section.

Table 13-13. FLASH_ECC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 13-13. FLASH_ECC_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

13.15.4.1 ECC_ENABLE Register (Offset = 0h) [Reset = Ah]

ECC_ENABLE is shown in Figure 13-13 and described in Table 13-14.

Return to the Summary Table.

ECC Enable

Figure 13-13. ECC_ENABLE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												ENABLE			
R-0h												R/W-Ah			

Table 13-14. ECC_ENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3-0	ENABLE	R/W	Ah	ECC enable. A value of 0xA would enable ECC. Any other value would disable ECC. Reset type: SYSRSn

13.15.4.2 SINGLE_ERR_ADDR_LOW Register (Offset = 2h) [Reset = 0h]

SINGLE_ERR_ADDR_LOW is shown in Figure 13-14 and described in Table 13-15.

Return to the Summary Table.

Single Error Address Low

Figure 13-14. SINGLE_ERR_ADDR_LOW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERR_ADDR_L																															
R/W-0h																															

Table 13-15. SINGLE_ERR_ADDR_LOW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ERR_ADDR_L	R/W	0h	64-bit aligned address at which a single bit error occurred in the lower 64-bits of a 128-bit aligned memory. Reset type: SYSRSn

13.15.4.3 SINGLE_ERR_ADDR_HIGH Register (Offset = 4h) [Reset = 0h]

SINGLE_ERR_ADDR_HIGH is shown in Figure 13-15 and described in Table 13-16.

Return to the Summary Table.

Single Error Address High

Figure 13-15. SINGLE_ERR_ADDR_HIGH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERR_ADDR_H																															
R/W-0h																															

Table 13-16. SINGLE_ERR_ADDR_HIGH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ERR_ADDR_H	R/W	0h	64-bit aligned address at which a single bit error occurred in the upper 64-bits of a 128-bit aligned memory. Reset type: SYSRSn

13.15.4.4 UNC_ERR_ADDR_LOW Register (Offset = 6h) [Reset = 0h]

UNC_ERR_ADDR_LOW is shown in Figure 13-16 and described in Table 13-17.

Return to the Summary Table.

Uncorrectable Error Address Low

Figure 13-16. UNC_ERR_ADDR_LOW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNC_ERR_ADDR_L																															
R/W-0h																															

Table 13-17. UNC_ERR_ADDR_LOW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UNC_ERR_ADDR_L	R/W	0h	64-bit aligned address at which an uncorrectable error occurred in the lower 64-bits of a 128-bit aligned memory. Reset type: SYSRSn

13.15.4.5 UNC_ERR_ADDR_HIGH Register (Offset = 8h) [Reset = 0h]

UNC_ERR_ADDR_HIGH is shown in Figure 13-17 and described in Table 13-18.

Return to the Summary Table.

Uncorrectable Error Address High

Figure 13-17. UNC_ERR_ADDR_HIGH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNC_ERR_ADDR_H																															
R/W-0h																															

Table 13-18. UNC_ERR_ADDR_HIGH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UNC_ERR_ADDR_H	R/W	0h	64-bit aligned address at which an uncorrectable error occurred in the upper 64-bits of a 128-bit aligned memory. Reset type: SYSRSn

13.15.4.6 ERR_STATUS Register (Offset = Ah) [Reset = 0h]

ERR_STATUS is shown in Figure 13-18 and described in Table 13-19.

Return to the Summary Table.

Error Status

Figure 13-18. ERR_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED					UNC_ERR_H	FAIL_1_H	FAIL_0_H
R-0h					R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					UNC_ERR_L	FAIL_1_L	FAIL_0_L
R-0h					R-0h	R-0h	R-0h

Table 13-19. ERR_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved
18	UNC_ERR_H	R	0h	Uncorrectable error. A value of 1 indicates that an un-correctable error occurred in upper 64bits of a 128-bit aligned address. Cleared by writing a 1 to UNC_ERR_H_CLR bit of ERR_STATUS_CLR register. Reset type: SYSRSn
17	FAIL_1_H	R	0h	Fail on 1. 0 Fail on 1 single bit error did not occur in upper 64bits of a 128-bit aligned address. 1 A value of 1 would indicate that a single bit error occurred in upper 64bits of a 128-bit aligned address and the corrected value was 1. Cleared by writing a 1 to FAIL_1_H_CLR bit of ERR_STATUS_CLR register. Note: This bit is updated on every flash access which results in a single bit error, So, in case of multiple single bit error, the status would correspond to the last error which occured. Reset type: SYSRSn
16	FAIL_0_H	R	0h	Fail on 0. 0 Fail on 0 single bit error did not occur in upper 64bits of a 128-bit aligned address. 1 A value of 1 would indicate that a single bit error occurred in upper 64bits of a 128-bit aligned address and the corrected value was 0. Cleared by writing a 1 to FAIL_0_H_CLR bit of ERR_STATUS_CLR register. Note: This bit is updated on every flash access which results in a single bit error, So, in case of multiple single bit error, the status would correspond to the last error which occurred. Reset type: SYSRSn
15-3	RESERVED	R	0h	Reserved
2	UNC_ERR_L	R	0h	Uncorrectable error. A value of 1 indicates that an un-correctable error occurred in lower 64bits of a 128-bit aligned address. Cleared by writing a 1 to UNC_ERR_L_CLR bit of ERR_STATUS_CLR register. Reset type: SYSRSn

Table 13-19. ERR_STATUS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	FAIL_1_L	R	0h	Fail on 1. 0 Fail on 1 single bit error did not occur in lower 64bits of a 128-bit aligned address. 1 A value of 1 would indicate that a single bit error occurred in lower 64bits of a 128-bit aligned address and the corrected value was 1. Cleared by writing a 1 to FAIL_1_L_CLR bit of ERR_STATUS_CLR register. Note: This bit is updated on every flash access which results in a single bit error, So, in case of multiple single bit error, the status would correspond to the last error which occured. Reset type: SYSRSn
0	FAIL_0_L	R	0h	Fail on 0. 0 Fail on 0 single bit error did not occur in lower 64bits of a 128-bit aligned address. 1 Would indicate that a single bit error occurred in lower 64bits of a 128-bit aligned address and the corrected value was 0. Cleared by writing a 1 to FAIL_0_L_CLR bit of ERR_STATUS_CLR register. Note: This bit is updated on every flash access which results in a single bit error, So, in case of multiple single bit error, the status would correspond to the last error which occured. Reset type: SYSRSn

13.15.4.7 ERR_POS Register (Offset = Ch) [Reset = 0h]

ERR_POS is shown in Figure 13-19 and described in Table 13-20.

Return to the Summary Table.

Error Position

Figure 13-19. ERR_POS Register

31	30	29	28	27	26	25	24
RESERVED							ERR_TYPE_H
R-0h							R-0h
23	22	21	20	19	18	17	16
RESERVED		ERR_POS_H					
R-0h		R-0h					
15	14	13	12	11	10	9	8
RESERVED							ERR_TYPE_L
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED		ERR_POS_L					
R-0h		R-0h					

Table 13-20. ERR_POS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24	ERR_TYPE_H	R	0h	Error type 0 Indicates that a single bit error occured in upper 64 data bits of a 128-bit aligned address. 1 Indicates that a single bit error occured in ECC check bits of upper 64bits of a 128-bit aligned address. Reset type: SYSRSn
23-22	RESERVED	R	0h	Reserved
21-16	ERR_POS_H	R	0h	Error position. Bit position of the single bit error in upper 64bits of a 128-bit aligned address. The position is interpreted depending on whether the ERR_TYPE bit indicates a check bit or a data bit. If ERR_TYPE indicates a check bit error, the error position could range from 0 to 7, else it could range from 0 to 63. Reset type: SYSRSn
15-9	RESERVED	R	0h	Reserved
8	ERR_TYPE_L	R	0h	Error type 0 Indicates that a single bit error occured in lower 64 data bits of a 128-bit aligned address. 1 Indicates that a single bit error occured in ECC check bits of lower 64bits of a 128-bit aligned address. Reset type: SYSRSn
7-6	RESERVED	R	0h	Reserved
5-0	ERR_POS_L	R	0h	Error position. Bit position of the single bit error in lower 64bits of a 128-bit aligned address. The position is interpreted depending on whether the ERR_TYPE bit indicates a check bit or a data bit. If ERR_TYPE indicates a check bit error, the error position could range from 0 to 7, else it could range from 0 to 63. Reset type: SYSRSn

13.15.4.8 ERR_STATUS_CLR Register (Offset = Eh) [Reset = 0h]

ERR_STATUS_CLR is shown in Figure 13-20 and described in Table 13-21.

Return to the Summary Table.

Error Status Clear

Figure 13-20. ERR_STATUS_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED					UNC_ERR_H_CLR	FAIL_1_H_CLR	FAIL_0_H_CLR
R-0h					R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					UNC_ERR_L_CLR	FAIL_1_L_CLR	FAIL_0_L_CLR
R-0h					R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 13-21. ERR_STATUS_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved
18	UNC_ERR_H_CLR	R-0/W1S	0h	Uncorrectable error clear. Writing a 1 to this bit will clear the UNC_ERR_H bit of ERR_STATUS register. Writes of 0 have no effect. Read returns 0. Reset type: SYSRSn
17	FAIL_1_H_CLR	R-0/W1S	0h	Fail on 1 clear. Writing a 1 to this bit will clear the FAIL_1_H bit of ERR_STATUS register. Writes of 0 have no effect. Read returns 0. Reset type: SYSRSn
16	FAIL_0_H_CLR	R-0/W1S	0h	Fail on 0 clear. Writing a 1 to this bit will clear the FAIL_0_H bit of ERR_STATUS register. Writes of 0 have no effect. Read returns 0. Reset type: SYSRSn
15-3	RESERVED	R	0h	Reserved
2	UNC_ERR_L_CLR	R-0/W1S	0h	Uncorrectable error clear. Writing a 1 to this bit will clear the UNC_ERR_L bit of ERR_STATUS register. Writes of 0 have no effect. Read returns 0. Reset type: SYSRSn
1	FAIL_1_L_CLR	R-0/W1S	0h	Fail on 1 clear. Writing a 1 to this bit will clear the FAIL_1_L bit of ERR_STATUS register. Writes of 0 have no effect. Read returns 0. Reset type: SYSRSn
0	FAIL_0_L_CLR	R-0/W1S	0h	Fail on 0 clear. Writing a 1 to this bit will clear the FAIL_0_L bit of ERR_STATUS register. Writes of 0 have no effect. Read returns 0. Reset type: SYSRSn

13.15.4.9 ERR_CNT Register (Offset = 10h) [Reset = 0h]

ERR_CNT is shown in Figure 13-21 and described in Table 13-22.

Return to the Summary Table.

Error Control

Figure 13-21. ERR_CNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ERR_CNT															
R-0h																R/W-0h															

Table 13-22. ERR_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	ERR_CNT	R/W	0h	Single bit error count. This counter increments with every single bit ECC error occurrence. Upon reaching the threshold value counter stops counting on single bit errors. ERR_CNT can be cleared (irrespective of whether threshold is met or not) using "Single Err Int Clear" bit. This is applicable for ECC logic test mode and normal operational mode. In ECC logic test mode, ERR_CNT will keep incrementing for every cycle after a single bit error occurrence. So, in order to clear ERR_CNT in ECC logic test mode, ECC logic test mode has to be disabled prior to clearing the ERR_CNT using "Single Err Int Clear" bit. Reset type: SYSRSn

13.15.4.10 ERR_THRESHOLD Register (Offset = 12h) [Reset = 0h]

ERR_THRESHOLD is shown in Figure 13-22 and described in Table 13-23.

Return to the Summary Table.

Error Threshold

Figure 13-22. ERR_THRESHOLD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ERR_THRESHOLD															
R-0h																R/W-0h															

Table 13-23. ERR_THRESHOLD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	ERR_THRESHOLD	R/W	0h	Single bit error threshold. Sets the threshold for single bit errors. When the ERR_CNT value equals the THRESHOLD value and a single bit error occurs, SINGLE_ERR_INT flag is set, and an interrupt is fired. Reset type: SYSRSn

13.15.4.11 ERR_INTFLG Register (Offset = 14h) [Reset = 0h]

ERR_INTFLG is shown in Figure 13-23 and described in Table 13-24.

Return to the Summary Table.

Error Interrupt Flag

Figure 13-23. ERR_INTFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						UNC_ERR_INTFLG	SINGLE_ERR_INTFLG
R-0h						R-0h	R-0h

Table 13-24. ERR_INTFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-2	RESERVED	R	0h	Reserved
1	UNC_ERR_INTFLG	R	0h	Uncorrectable bit error interrupt flag. When a Un-correctable error occurs, this bit is set and the UNC_ERR_INT interrupt is fired. When UNC_ERR_INTCLR bit of ERR_INTCLR register is written a value of 1 this bit is cleared. Reset type: SYSRSn
0	SINGLE_ERR_INTFLG	R	0h	Single bit error interrupt flag. When the ERR_CNT value equals the ERR_THRESHOLD value and a single bit error occurs then SINGLE_ERR_INT flag is set and SINGLE_ERR_INT interrupt is fired. When SINGLE_ERR_INTCLR bit of ERR_INTCLR register is written a value of 1 this bit is cleared. Reset type: SYSRSn

13.15.4.12 ERR_INTCLR Register (Offset = 16h) [Reset = 0h]

ERR_INTCLR is shown in Figure 13-24 and described in Table 13-25.

Return to the Summary Table.

Error Interrupt Flag Clear

Figure 13-24. ERR_INTCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						UNC_ERR_INTCLR	SINGLE_ERR_INTCLR
R-0h						R-0/W1S-0h	R-0/W1S-0h

Table 13-25. ERR_INTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-2	RESERVED	R	0h	Reserved
1	UNC_ERR_INTCLR	R-0/W1S	0h	Uncorrectable bit error interrupt flag clear. Writing a 1 to this bit will clear UNC_ERR_INT_FLG. Writes of 0 have no effect. Reset type: SYSRSn
0	SINGLE_ERR_INTCLR	R-0/W1S	0h	Single bit error interrupt flag clear. Writing a 1 to this bit will clear SINGLE_ERR_INT_FLG. Writes of 0 have no effect. Reset type: SYSRSn

13.15.4.13 FDATAH_TEST Register (Offset = 18h) [Reset = 0h]

FDATAH_TEST is shown in Figure 13-25 and described in Table 13-26.

Return to the Summary Table.

Data High Test

Figure 13-25. FDATAH_TEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FDATAH																															
R/W-0h																															

Table 13-26. FDATAH_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FDATAH	R/W	0h	High double word of selected 64-bit data. User-configurable bits 63:32 of the selected data block in ECC test mode. Reset type: SYSRSn

13.15.4.14 FDATAL_TEST Register (Offset = 1Ah) [Reset = 0h]

FDATAL_TEST is shown in Figure 13-26 and described in Table 13-27.

Return to the Summary Table.

Data Low Test

Figure 13-26. FDATAL_TEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FDATAL																															
R/W-0h																															

Table 13-27. FDATAL_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FDATAL	R/W	0h	Low double word of selected 64-bit data. User-configurable bits 31:0 of the selected data block in ECC test mode. Reset type: SYSRSn

13.15.4.15 FADDR_TEST Register (Offset = 1Ch) [Reset = 0h]

FADDR_TEST is shown in Figure 13-27 and described in Table 13-28.

Return to the Summary Table.

ECC Test Address

Figure 13-27. FADDR_TEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED										ADDRH					
R-0h										R/W-0h					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRL													RESERVED		
R/W-0h													R-0h		

Table 13-28. FADDR_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reserved
21-16	ADDRH	R/W	0h	Address for selected 64-bit data. User-configurable address bits of the selected data in ECC test mode. Left-shift the address by one bit (to provide byte address) and ignore the three least significant bits of the address and write the bits 21:16 in remaining address bits in this field. Reset type: SYSRSn
15-3	ADDRL	R/W	0h	Address for selected 64-bit data. User-configurable address bits of the selected data in ECC test mode. Left-shift the address by one bit (to provide byte address) and ignore the three least significant bits of the address and write the bits 15:3 in remaining address bits in this field. Reset type: SYSRSn
2-0	RESERVED	R	0h	Reserved

13.15.4.16 FECC_TEST Register (Offset = 1Eh) [Reset = 0h]

FECC_TEST is shown in Figure 13-28 and described in Table 13-29.

Return to the Summary Table.

ECC Test Address

Figure 13-28. FECC_TEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RESERVED								ECC							
R-0h																R-0h								R/W-0h							

Table 13-29. FECC_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7-0	ECC	R/W	0h	8-bit ECC for selected 64-bit data. User-configurable ECC bits of the selected 64-bit data block in ECC test mode. Reset type: SYSRSn

13.15.4.17 FECC_CTRL Register (Offset = 20h) [Reset = 0h]

FECC_CTRL is shown in Figure 13-29 and described in Table 13-30.

Return to the Summary Table.

ECC Control

Figure 13-29. FECC_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					DO_ECC_CALC	ECC_SELECT	ECC_TEST_EN
R-0h					R-0/W1S-0h	R/W-0h	R/W-0h

Table 13-30. FECC_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-3	RESERVED	R	0h	Reserved
2	DO_ECC_CALC	R-0/W1S	0h	Enable ECC calculation. ECC logic will calculate ECC in one cycle for the data and address written in ECC test registers when ECC test logic is enabled by setting ECC_TEST_EN. Reset type: SYSRSn
1	ECC_SELECT	R/W	0h	ECC block select. 0 Selects the ECC block on bits [63:0] of bank data. 1 Selects the ECC block on bits [127:64] of bank data. Reset type: SYSRSn
0	ECC_TEST_EN	R/W	0h	ECC test mode enable. 0 ECC test mode disabled 1 ECC test mode enabled Reset type: SYSRSn

13.15.4.18 FOUTH_TEST Register (Offset = 22h) [Reset = 0h]

FOUTH_TEST is shown in Figure 13-30 and described in Table 13-31.

Return to the Summary Table.

Test Data Out High

Figure 13-30. FOUTH_TEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATAOUTH																															
R-0h																															

Table 13-31. FOUTH_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATAOUTH	R	0h	High double word test data out. Holds bits 63:32 of the data out of the selected ECC block. Reset type: SYSRSn

13.15.4.19 FOUTL_TEST Register (Offset = 24h) [Reset = 0h]

FOUTL_TEST is shown in Figure 13-31 and described in Table 13-32.

Return to the Summary Table.

Test Data Out Low

Figure 13-31. FOUTL_TEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATAOUTL																															
R-0h																															

Table 13-32. FOUTL_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATAOUTL	R	0h	Low double word test data out. Holds bits 31:0 of the data out of the selected ECC block. Reset type: SYSRSn

13.15.4.20 FECC_STATUS Register (Offset = 26h) [Reset = 0h]

FECC_STATUS is shown in Figure 13-32 and described in Table 13-33.

Return to the Summary Table.

ECC Status

Figure 13-32. FECC_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							ERR_TYPE
R-0h							R-0h
7	6	5	4	3	2	1	0
DATA_ERR_POS						UNC_ERR	SINGLE_ERR
R-0h						R-0h	R-0h

Table 13-33. FECC_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-9	RESERVED	R	0h	Reserved
8	ERR_TYPE	R	0h	Test mode ECC single bit error indicator. When 1, indicates that the single bit error is in check bits. When 0, indicates that the single bit error is in data bits (If SINGLE_ERR field is also set). Reset type: SYSRSn
7-2	DATA_ERR_POS	R	0h	Test mode single bit error position. Holds the bit position where the single bit error occurred. The position is interpreted depending on whether the CHK_ERR bit indicates a check bit or a data bit. If CHK_ERR indicates a check bit error, the error position could range from 0 to 7, or it could range from 0 to 63. Reset type: SYSRSn
1	UNC_ERR	R	0h	Test mode ECC double bit error. When 1 indicates that the ECC test resulted in an uncorrectable bit error. Reset type: SYSRSn
0	SINGLE_ERR	R	0h	Test mode ECC single bit error. When 1 indicates that the ECC test resulted in a single bit error. Reset type: SYSRSn

13.15.5 CM_FLASH_CTRL_REGS Registers

Table 13-34 lists the memory-mapped registers for the CM_FLASH_CTRL_REGS registers. All register offset addresses not listed in Table 13-34 should be considered as reserved locations and the register contents should not be modified.

Table 13-34. CM_FLASH_CTRL_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	FRDCNTL	Flash Read Control Register	Yes	Go
1Eh	FBAC	Flash Bank Access Control Register	Yes	Go
20h	FBFALLBACK	Flash Bank Fallback Power Register	Yes	Go
22h	FBPRDY	Flash Bank Pump Ready Register	Yes	Go
24h	FPAC1	Flash Pump Access Control Register 1	Yes	Go
2Ah	FMSTAT	Flash Module Status Register	Yes	Go
17Eh	FRD_INTF_CTRL_LOCK	Lock register for FLASH_CTRL_REGS (Not including FRD_INTF_CTRL_LOCK).	No	Go
180h	FRD_INTF_CTRL	Flash Read Interface Control Register	Yes	Go

Complex bit access types are encoded to fit into small table cells. Table 13-35 shows the codes that are used for access types in this section.

Table 13-35. CM_FLASH_CTRL_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

13.15.5.1 FRDCNTL Register (Offset = 0h) [Reset = F00h]

FRDCNTL is shown in Figure 13-33 and described in Table 13-36.

Return to the Summary Table.

Flash Read Control Register

Figure 13-33. FRDCNTL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				RWAIT				RESERVED							
R-0h				R/W-Fh				R-0h							

Table 13-36. FRDCNTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-12	RESERVED	R	0h	Reserved
11-8	RWAIT	R/W	Fh	Random read waitstate These bits indicate how many waitstates are added to a flash read access. The RWAIT value can be set anywhere from 0 to 0xF. For a flash access, data is returned in RWAIT+1 CMCLK cycles. Note: Following table summarizes the effect of accessing TI or USER OTP space. RWAIT Value | Data returned after 0 | 1 cycle 1 to F | RWAIT + 2 cycles Reset type: CM.RESETn
7-0	RESERVED	R	0h	Reserved

13.15.5.2 FBAC Register (Offset = 1Eh) [Reset = Fh]

FBAC is shown in Figure 13-34 and described in Table 13-37.

Return to the Summary Table.

Flash Bank Access Control Register

Figure 13-34. FBAC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																BAGP								RESERVED							
R-0h																R/W-0h								R/W-Fh							

Table 13-37. FBAC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	BAGP	R/W	0h	Bank Active Grace Period. These bits contain the starting count value for the BAGP down counter. Any access to a given bank causes its BAGP counter to reload the BAGP value for that bank. After the last access to this flash bank, the down counter delays from 0 to 255 prescaled CMCLK clock cycles before putting the bank into one of the fallback power modes as determined by the FBFALLBACK register. This value must be greater than 1 when the fallback mode is not ACTIVE. Note: The prescaled clock used for the BAGP down counter is a clock divided by 16 from input CMCLK. Reset type: CM.RESETn
7-0	RESERVED	R/W	Fh	Reserved

13.15.5.3 FBFALLBACK Register (Offset = 20h) [Reset = 0h]

FBFALLBACK is shown in Figure 13-35 and described in Table 13-38.

Return to the Summary Table.

Flash Bank Fallback Power Register

Figure 13-35. FBFALLBACK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED		BNKPWR0	
R-0h				R/W-0h		R/W-0h	

Table 13-38. FBFALLBACK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3-2	RESERVED	R/W	0h	Reserved
1-0	BNKPWR0	R/W	0h	Fall Back power mode 00 Sleep (Sense amplifiers and sense reference disabled) 01 Standby (Sense amplifiers disabled, but sense reference enabled) 10 Reserved 11 Active (Both sense amplifiers and sense reference enabled) Note: If the bank and pump are not in active mode and an access is made, the value of this register is automatically changed to active. Reset type: CM.RESETn

13.15.5.4 FBPRDY Register (Offset = 22h) [Reset = 0h]

FBPRDY is shown in Figure 13-36 and described in Table 13-39.

Return to the Summary Table.

Flash Bank Pump Ready Register

Figure 13-36. FBPRDY Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
PUMPRDY	RESERVED						
R-0h	R-0h						
7	6	5	4	3	2	1	0
RESERVED							BANKRDY
R-0h							R-0h

Table 13-39. FBPRDY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	PUMPRDY	R	0h	Pump Ready. This is a read-only bit which allows software to determine if the pump is ready for flash access before attempting the actual access. If an access is made to a bank when the pump is not ready, wait states are asserted until it becomes ready. 0 Pump is not ready. 1 Pump is ready, in active power state. Reset type: CM.RESETn
14-1	RESERVED	R	0h	Reserved
0	BANKRDY	R	0h	Bank Ready. This is a read-only register which allows software to determine if the bank is ready for Flash access before the access is attempted. Note: The user should wait for both the pump and the bank to be ready before attempting an access. 0 Bank is not ready. 1 Bank is in active power mode and is ready for access. Reset type: CM.RESETn

13.15.5.5 FPAC1 Register (Offset = 24h) [Reset = 04E20000h]

FPAC1 is shown in Figure 13-37 and described in Table 13-40.

Return to the Summary Table.

Flash Pump Access Control Register 1

Figure 13-37. FPAC1 Register

31	30	29	28	27	26	25	24
RESERVED				PSLEEP			
R-0h				R/W-4E2h			
23	22	21	20	19	18	17	16
PSLEEP							
R/W-4E2h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							PMPPWR
R-0h							R/W-0h

Table 13-40. FPAC1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-16	PSLEEP	R/W	4E2h	Pump sleep. These bits contain the starting count value for the charge pump sleep down counter. While the charge pump is in sleep mode, the power mode management logic holds the charge pump sleep counter at this value. When the charge pump exits sleep power mode, the down counter delays from 0 to PSLEEP prescaled CMCLK clock cycles before putting the charge pump into active power mode. Note: The pump sleep down counter uses the same prescaled clock as Bank sleep down counter which is divided by 2 of input CMCLK. Reset type: CM.RESETn
15-1	RESERVED	R	0h	Reserved
0	PMPPWR	R/W	0h	Flash Charge Pump Fallback Power Mode. This bit selects what power mode the charge pump enters after the pump active grace period (PAGP) counter has timed out. 0 Sleep (all pump circuits disabled) 1 Active (all pump circuits active) Note for devices with multiple flash banks: As the pump is shared between flash banks, if an access is made either bank, the value of this bit changes to 1 (active). Reset type: CM.RESETn

13.15.5.6 FMSTAT Register (Offset = 2Ah) [Reset = 0h]

FMSTAT is shown in Figure 13-38 and described in Table 13-41.

Return to the Summary Table.

Flash Module Status Register

Figure 13-38. FMSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						RESERVED	RESERVED
R-0h						R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	PGV	RESERVED	EV	RESERVED	Busy
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
ERS	PGM	INVDAT	CSTAT	VOLTSTAT	ESUSP	PSUSP	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 13-41. FMSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17	RESERVED	R	0h	Reserved
16	RESERVED	R	0h	Reserved
15	RESERVED	R	0h	Reserved
14	RESERVED	R	0h	Reserved
13	RESERVED	R	0h	Reserved
12	PGV	R	0h	When set, indicates that a word is not successfully programmed after the maximum allowed number of program pulses are given for program operation. Reset type: CM.RESETn
11	RESERVED	R	0h	Reserved
10	EV	R	0h	When set, indicates that a sector is not successfully erased after the maximum allowed number of erase pulses are given for erase operation. During Erase verify command, this flag is set immediately if a bit is found to be 0. Reset type: CM.RESETn
9	RESERVED	R	0h	Reserved
8	Busy	R	0h	When set, this bit indicates that a program, erase, or suspend operation is being processed. Reset type: CM.RESETn
7	ERS	R	0h	When set, this bit indicates that the flash module is actively performing an erase operation. This bit is set when erasing starts and is cleared when erasing is complete. It is also cleared when the erase is suspended and set when the erase resumes. Reset type: CM.RESETn
6	PGM	R	0h	When set, this bit indicates that the flash module is currently performing a program operation. This bit is set when programming starts and is cleared when programming is complete. It is also cleared when programming is suspended and set when programming is resumes. Reset type: CM.RESETn

Table 13-41. FMSTAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	INVDAT	R	0h	When set, this bit indicates that the user attempted to program a 1 where a 0 was already present. This bit is cleared by the Clear Status command. Reset type: CM.RESETn
4	CSTAT	R	0h	Once the FSM starts any failure will set this bit. When set, this bit informs the host that the program, erase, or validate sector command failed and the command was stopped. This bit is cleared by the Clear Status command. For some errors, this will be the only indication of an FSM error because the cause does not fall within the other error bit types. Reset type: CM.RESETn
3	VOLTSTAT	R	0h	When set, this bit indicates that the core voltage generator of the pump power supply dipped below the lower limit allowable during a program or erase operation. This bit is cleared by the Clear Status command. Reset type: CM.RESETn
2	ESUSP	R	0h	When set, this bit indicates that the flash module has received and processed an erase suspend operation. This bit remains set until the erase resume command has been issued or until the Clear_More command is run. Reset type: CM.RESETn
1	PSUSP	R	0h	When set, this bit indicates that the flash module has received and processed a program suspend operation. This bit remains set until the program resume command has been issued or until the Clear_More command is run. Reset type: CM.RESETn
0	RESERVED	R	0h	Reserved

13.15.5.7 FRD_INTF_CTRL_LOCK Register (Offset = 17Eh) [Reset = 0h]

FRD_INTF_CTRL_LOCK is shown in Figure 13-39 and described in Table 13-42.

Return to the Summary Table.

Lock register for FLASH_CTRL_REGS (Not including FRD_INTF_CTRL_LOCK).

Figure 13-39. FRD_INTF_CTRL_LOCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOCK																															
R/W-0h																															

Table 13-42. FRD_INTF_CTRL_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	LOCK	R/W	0h	These bits when set to "0xA5A5A5A5" in FRD_INTF_CTRL_LOCK register enable writes to FRD_CTRL_REGS registers block (Not including FRD_INTF_CTRL_LOCK register).Any other value will disable register writes. Reset type: CM.RESETn

13.15.5.8 FRD_INTF_CTRL Register (Offset = 180h) [Reset = 0h]

FRD_INTF_CTRL is shown in Figure 13-40 and described in Table 13-43.

Return to the Summary Table.

Flash Read Interface Control Register

Figure 13-40. FRD_INTF_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						DATA_CACHE_EN	PROG_CACHE_EN
R-0h						R/W-0h	R/W-0h

Table 13-43. FRD_INTF_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-2	RESERVED	R	0h	Reserved
1	DATA_CACHE_EN	R/W	0h	Data cache enable. 0 A value of 0 disables the data cache. 1 A value of 1 enables the data cache. Reset type: CM.RESETn
0	PROG_CACHE_EN	R/W	0h	Prefetch enable. 0 A value of 0 disables program cache and prefetch mechanism. 1 A value of 1 enables program cache and pre-fetch mechanism. Reset type: CM.RESETn

13.15.6 CM_FLASH_ECC_REGS Registers

Table 13-44 lists the memory-mapped registers for the CM_FLASH_ECC_REGS registers. All register offset addresses not listed in Table 13-44 should be considered as reserved locations and the register contents should not be modified.

Table 13-44. CM_FLASH_ECC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	ECC_ENABLE	ECC Enable	Yes	Go
2h	SINGLE_ERR_ADDR_LOW	Single Error Address Low	Yes	Go
4h	SINGLE_ERR_ADDR_HIGH	Single Error Address High	Yes	Go
6h	UNC_ERR_ADDR_LOW	Uncorrectable Error Address Low	Yes	Go
8h	UNC_ERR_ADDR_HIGH	Uncorrectable Error Address High	Yes	Go
Ah	ERR_STATUS	Error Status	Yes	Go
Ch	ERR_POS	Error Position	Yes	Go
Eh	ERR_STATUS_CLR	Error Status Clear	Yes	Go
10h	ERR_CNT	Error Control	Yes	Go
12h	ERR_THRESHOLD	Error Threshold	Yes	Go
14h	ERR_INTFLG	Error Interrupt Flag	Yes	Go
16h	ERR_INTCLR	Error Interrupt Flag Clear	Yes	Go
18h	FDATAH_TEST	Data High Test	Yes	Go
1Ah	FDATAL_TEST	Data Low Test	Yes	Go
1Ch	FADDR_TEST	ECC Test Address	Yes	Go
1Eh	FECC_TEST	ECC Test Address	Yes	Go
20h	FECC_CTRL	ECC Control	Yes	Go
22h	FOUTH_TEST	Test Data Out High	Yes	Go
24h	FOUTL_TEST	Test Data Out Low	Yes	Go
26h	FECC_STATUS	ECC Status	Yes	Go
3Eh	FLASH_ECC_REGS_LOCK	Lock register for FLASH_ECC_REGS (Not including FLASH_ECC_REGS_LOCK).	No	Go

Complex bit access types are encoded to fit into small table cells. Table 13-45 shows the codes that are used for access types in this section.

Table 13-45. CM_FLASH_ECC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 13-45. CM_FLASH_ECC_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

13.15.6.1 ECC_ENABLE Register (Offset = 0h) [Reset = Ah]

ECC_ENABLE is shown in Figure 13-41 and described in Table 13-46.

Return to the Summary Table.

ECC Enable

Figure 13-41. ECC_ENABLE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												ENABLE			
R-0h												R/W-Ah			

Table 13-46. ECC_ENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3-0	ENABLE	R/W	Ah	ECC enable. A value of 0xA would enable ECC. Any other value would disable ECC. Reset type: CM.RESETn

13.15.6.2 SINGLE_ERR_ADDR_LOW Register (Offset = 2h) [Reset = 0h]

SINGLE_ERR_ADDR_LOW is shown in Figure 13-42 and described in Table 13-47.

Return to the Summary Table.

Single Error Address Low

Figure 13-42. SINGLE_ERR_ADDR_LOW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERR_ADDR_L																															
R/W-0h																															

Table 13-47. SINGLE_ERR_ADDR_LOW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ERR_ADDR_L	R/W	0h	64-bit aligned address at which a single bit error occurred in the lower 64-bits of a 128-bit aligned memory. Reset type: CM.RESETn

13.15.6.3 SINGLE_ERR_ADDR_HIGH Register (Offset = 4h) [Reset = 0h]

SINGLE_ERR_ADDR_HIGH is shown in Figure 13-43 and described in Table 13-48.

Return to the Summary Table.

Single Error Address High

Figure 13-43. SINGLE_ERR_ADDR_HIGH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERR_ADDR_H																															
R/W-0h																															

Table 13-48. SINGLE_ERR_ADDR_HIGH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ERR_ADDR_H	R/W	0h	64-bit aligned address at which a single bit error occurred in the upper 64-bits of a 128-bit aligned memory. Reset type: CM.RESETn

13.15.6.4 UNC_ERR_ADDR_LOW Register (Offset = 6h) [Reset = 0h]

UNC_ERR_ADDR_LOW is shown in Figure 13-44 and described in Table 13-49.

Return to the Summary Table.

Uncorrectable Error Address Low

Figure 13-44. UNC_ERR_ADDR_LOW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNC_ERR_ADDR_L																															
R/W-0h																															

Table 13-49. UNC_ERR_ADDR_LOW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UNC_ERR_ADDR_L	R/W	0h	64-bit aligned address at which an uncorrectable error occurred in the lower 64-bits of a 128-bit aligned memory. Reset type: CM.RESETn

13.15.6.5 UNC_ERR_ADDR_HIGH Register (Offset = 8h) [Reset = 0h]

UNC_ERR_ADDR_HIGH is shown in Figure 13-45 and described in Table 13-50.

Return to the Summary Table.

Uncorrectable Error Address High

Figure 13-45. UNC_ERR_ADDR_HIGH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNC_ERR_ADDR_H																															
R/W-0h																															

Table 13-50. UNC_ERR_ADDR_HIGH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UNC_ERR_ADDR_H	R/W	0h	64-bit aligned address at which an uncorrectable error occurred in the upper 64-bits of a 128-bit aligned memory. Reset type: CM.RESETn

13.15.6.6 ERR_STATUS Register (Offset = Ah) [Reset = 0h]

ERR_STATUS is shown in Figure 13-46 and described in Table 13-51.

Return to the Summary Table.

Error Status

Figure 13-46. ERR_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED					UNC_ERR_H	FAIL_1_H	FAIL_0_H
R-0h					R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					UNC_ERR_L	FAIL_1_L	FAIL_0_L
R-0h					R-0h	R-0h	R-0h

Table 13-51. ERR_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved
18	UNC_ERR_H	R	0h	Uncorrectable error. A value of 1 indicates that an un-correctable error occurred in upper 64bits of a 128-bit aligned address. Cleared by writing a 1 to UNC_ERR_H_CLR bit of ERR_STATUS_CLR register. Reset type: CM.RESETn
17	FAIL_1_H	R	0h	Fail on 1. 0 Fail on 1 single bit error did not occur in upper 64bits of a 128-bit aligned address. 1 A value of 1 would indicate that a single bit error occurred in upper 64bits of a 128-bit aligned address and the corrected value was 1. Cleared by writing a 1 to FAIL_1_H_CLR bit of ERR_STATUS_CLR register. Note: This bit is updated on every flash access which results in a single bit error, So, in case of multiple single bit error, the status would correspond to the last error which occured. Reset type: CM.RESETn
16	FAIL_0_H	R	0h	Fail on 0. 0 Fail on 0 single bit error did not occur in upper 64bits of a 128-bit aligned address. 1 A value of 1 would indicate that a single bit error occurred in upper 64bits of a 128-bit aligned address and the corrected value was 0. Cleared by writing a 1 to FAIL_0_H_CLR bit of ERR_STATUS_CLR register. Note: This bit is updated on every flash access which results in a single bit error, So, in case of multiple single bit error, the status would correspond to the last error which occured. Reset type: CM.RESETn
15-3	RESERVED	R	0h	Reserved
2	UNC_ERR_L	R	0h	Uncorrectable error. A value of 1 indicates that an un-correctable error occurred in lower 64bits of a 128-bit aligned address. Cleared by writing a 1 to UNC_ERR_L_CLR bit of ERR_STATUS_CLR register. Reset type: CM.RESETn

Table 13-51. ERR_STATUS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	FAIL_1_L	R	0h	Fail on 1. 0 Fail on 1 single bit error did not occur in lower 64bits of a 128-bit aligned address. 1 A value of 1 would indicate that a single bit error occurred in lower 64bits of a 128-bit aligned address and the corrected value was 1. Cleared by writing a 1 to FAIL_1_L_CLR bit of ERR_STATUS_CLR register. Note: This bit is updated on every flash access which results in a single bit error, So, in case of multiple single bit error, the status would correspond to the last error which occured. Reset type: CM.RESETn
0	FAIL_0_L	R	0h	Fail on 0. 0 Fail on 0 single bit error did not occur in lower 64bits of a 128-bit aligned address. 1 Would indicate that a single bit error occurred in lower 64bits of a 128-bit aligned address and the corrected value was 0. Cleared by writing a 1 to FAIL_0_L_CLR bit of ERR_STATUS_CLR register. Note: This bit is updated on every flash access which results in a single bit error, So, in case of multiple single bit error, the status would correspond to the last error which occured. Reset type: CM.RESETn

13.15.6.7 ERR_POS Register (Offset = Ch) [Reset = 0h]

ERR_POS is shown in Figure 13-47 and described in Table 13-52.

Return to the Summary Table.

Error Position

Figure 13-47. ERR_POS Register

31	30	29	28	27	26	25	24
RESERVED							ERR_TYPE_H
R-0h							R/W-0h
23	22	21	20	19	18	17	16
RESERVED		ERR_POS_H					
R-0h		R/W-0h					
15	14	13	12	11	10	9	8
RESERVED							ERR_TYPE_L
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED		ERR_POS_L					
R-0h		R/W-0h					

Table 13-52. ERR_POS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24	ERR_TYPE_H	R/W	0h	Error type 0 Indicates that a single bit error occured in upper 64 data bits of a 128-bit aligned address. 1 Indicates that a single bit error occured in ECC check bits of upper 64bits of a 128-bit aligned address. Reset type: CM.RESETn
23-22	RESERVED	R	0h	Reserved
21-16	ERR_POS_H	R/W	0h	Error position. Bit position of the single bit error in upper 64bits of a 128-bit aligned address. The position is interpreted depending on whether the ERR_TYPE bit indicates a check bit or a data bit. If ERR_TYPE indicates a check bit error, the error position could range from 0 to 7, else it could range from 0 to 63. Reset type: CM.RESETn
15-9	RESERVED	R	0h	Reserved
8	ERR_TYPE_L	R/W	0h	Error type 0 Indicates that a single bit error occured in lower 64 data bits of a 128-bit aligned address. 1 Indicates that a single bit error occured in ECC check bits of lower 64bits of a 128-bit aligned address. Reset type: CM.RESETn
7-6	RESERVED	R	0h	Reserved
5-0	ERR_POS_L	R/W	0h	Error position. Bit position of the single bit error in lower 64bits of a 128-bit aligned address. The position is interpreted depending on whether the ERR_TYPE bit indicates a check bit or a data bit. If ERR_TYPE indicates a check bit error, the error position could range from 0 to 7, else it could range from 0 to 63. Reset type: CM.RESETn

13.15.6.8 ERR_STATUS_CLR Register (Offset = Eh) [Reset = 0h]

ERR_STATUS_CLR is shown in Figure 13-48 and described in Table 13-53.

Return to the Summary Table.

Error Status Clear

Figure 13-48. ERR_STATUS_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED					UNC_ERR_H_CLR	FAIL_1_H_CLR	FAIL_0_H_CLR
R-0h					R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					UNC_ERR_L_CLR	FAIL_1_L_CLR	FAIL_0_L_CLR
R-0h					R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 13-53. ERR_STATUS_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved
18	UNC_ERR_H_CLR	R-0/W1S	0h	Uncorrectable error clear. Writing a 1 to this bit will clear the UNC_ERR_H bit of ERR_STATUS register. Writes of 0 have no effect. Read returns 0. Reset type: CM.RESETn
17	FAIL_1_H_CLR	R-0/W1S	0h	Fail on 1 clear. Writing a 1 to this bit will clear the FAIL_1_H bit of ERR_STATUS register. Writes of 0 have no effect. Read returns 0. Reset type: CM.RESETn
16	FAIL_0_H_CLR	R-0/W1S	0h	Fail on 0 clear. Writing a 1 to this bit will clear the FAIL_0_H bit of ERR_STATUS register. Writes of 0 have no effect. Read returns 0. Reset type: CM.RESETn
15-3	RESERVED	R	0h	Reserved
2	UNC_ERR_L_CLR	R-0/W1S	0h	Uncorrectable error clear. Writing a 1 to this bit will clear the UNC_ERR_L bit of ERR_STATUS register. Writes of 0 have no effect. Read returns 0. Reset type: CM.RESETn
1	FAIL_1_L_CLR	R-0/W1S	0h	Fail on 1 clear. Writing a 1 to this bit will clear the FAIL_1_L bit of ERR_STATUS register. Writes of 0 have no effect. Read returns 0. Reset type: CM.RESETn
0	FAIL_0_L_CLR	R-0/W1S	0h	Fail on 0 clear. Writing a 1 to this bit will clear the FAIL_0_L bit of ERR_STATUS register. Writes of 0 have no effect. Read returns 0. Reset type: CM.RESETn

13.15.6.9 ERR_CNT Register (Offset = 10h) [Reset = 0h]

ERR_CNT is shown in Figure 13-49 and described in Table 13-54.

Return to the Summary Table.

Error Control

Figure 13-49. ERR_CNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ERR_CNT															
R-0h																R/W-0h															

Table 13-54. ERR_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	ERR_CNT	R/W	0h	Single bit error count. This counter increments with every single bit ECC error occurrence. Upon reaching the threshold value counter stops counting on single bit errors. ERR_CNT can be cleared (irrespective of whether threshold is met or not) using "Single Err Int Clear" bit. This is applicable for ECC logic test mode and normal operational mode. In ECC logic test mode, ERR_CNT will keep incrementing for every cycle after a single bit error occurrence. So, in order to clear ERR_CNT in ECC logic test mode, ECC logic test mode has to be disabled prior to clearing the ERR_CNT using "Single Err Int Clear" bit. Reset type: CM.RESETn

13.15.6.10 ERR_THRESHOLD Register (Offset = 12h) [Reset = 0h]

ERR_THRESHOLD is shown in Figure 13-50 and described in Table 13-55.

Return to the Summary Table.

Error Threshold

Figure 13-50. ERR_THRESHOLD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ERR_THRESHOLD															
R-0h																R/W-0h															

Table 13-55. ERR_THRESHOLD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	ERR_THRESHOLD	R/W	0h	Single bit error threshold. Sets the threshold for single bit errors. When the ERR_CNT value equals the THRESHOLD value and a single bit error occurs, SINGLE_ERR_INT flag is set, and an interrupt is fired. Reset type: CM.RESETn

13.15.6.11 ERR_INTFLG Register (Offset = 14h) [Reset = 0h]

ERR_INTFLG is shown in Figure 13-51 and described in Table 13-56.

Return to the Summary Table.

Error Interrupt Flag

Figure 13-51. ERR_INTFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						UNC_ERR_INTFLG	SINGLE_ERR_INTFLG
R-0h						R-0h	R-0h

Table 13-56. ERR_INTFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-2	RESERVED	R	0h	Reserved
1	UNC_ERR_INTFLG	R	0h	Uncorrectable bit error interrupt flag. When a Un-correctable error occurs, this bit is set and the UNC_ERR_INT interrupt is fired. When UNC_ERR_INTCLR bit of ERR_INTCLR register is written a value of 1 this bit is cleared. Reset type: CM.RESETn
0	SINGLE_ERR_INTFLG	R	0h	Single bit error interrupt flag. When the ERR_CNT value equals the ERR_THRESHOLD value and a single bit error occurs then SINGLE_ERR_INT flag is set and SINGLE_ERR_INT interrupt is fired. When SINGLE_ERR_INTCLR bit of ERR_INTCLR register is written a value of 1 this bit is cleared. Reset type: CM.RESETn

13.15.6.12 ERR_INTCLR Register (Offset = 16h) [Reset = 0h]

ERR_INTCLR is shown in Figure 13-52 and described in Table 13-57.

Return to the Summary Table.

Error Interrupt Flag Clear

Figure 13-52. ERR_INTCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						UNC_ERR_INTCLR	SINGLE_ERR_INTCLR
R-0h						R-0/W1S-0h	R-0/W1S-0h

Table 13-57. ERR_INTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-2	RESERVED	R	0h	Reserved
1	UNC_ERR_INTCLR	R-0/W1S	0h	Uncorrectable bit error interrupt flag clear. Writing a 1 to this bit will clear UNC_ERR_INT_FLG. Writes of 0 have no effect. Reset type: CM.RESETn
0	SINGLE_ERR_INTCLR	R-0/W1S	0h	Single bit error interrupt flag clear. Writing a 1 to this bit will clear SINGLE_ERR_INT_FLG. Writes of 0 have no effect. Reset type: CM.RESETn

13.15.6.13 FDATAH_TEST Register (Offset = 18h) [Reset = 0h]

FDATAH_TEST is shown in Figure 13-53 and described in Table 13-58.

Return to the Summary Table.

Data High Test

Figure 13-53. FDATAH_TEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FDATAH																															
R/W-0h																															

Table 13-58. FDATAH_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FDATAH	R/W	0h	High double word of selected 64-bit data. User-configurable bits 63:32 of the selected data blocks in ECC test mode. Reset type: CM.RESETn

13.15.6.14 FDATAL_TEST Register (Offset = 1Ah) [Reset = 0h]

FDATAL_TEST is shown in Figure 13-54 and described in Table 13-59.

Return to the Summary Table.

Data Low Test

Figure 13-54. FDATAL_TEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FDATAL																															
R/W-0h																															

Table 13-59. FDATAL_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FDATAL	R/W	0h	Low double word of selected 64-bit data. User-configurable bits 31:0 of the selected data blocks in ECC test mode. Reset type: CM.RESETn

13.15.6.15 FADDR_TEST Register (Offset = 1Ch) [Reset = 0h]

FADDR_TEST is shown in Figure 13-55 and described in Table 13-60.

Return to the Summary Table.

ECC Test Address

Figure 13-55. FADDR_TEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED										ADDRH					
R-0h										R/W-0h					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRL													RESERVED		
R/W-0h													R-0h		

Table 13-60. FADDR_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reserved
21-16	ADDRH	R/W	0h	Address for selected 64-bit data. User-configurable address bits of the selected data in ECC test mode. Ignore the three least significant bits of the address and write the bits 21:16 in remaining address bits in this field. Reset type: CM.RESETn
15-3	ADDRL	R/W	0h	Address for selected 64-bit data. User-configurable address bits of the selected data in ECC test mode.Ignore the three least significant bits of the address and write the bits 15:3 in remaining address bits in this field. Reset type: CM.RESETn
2-0	RESERVED	R	0h	Reserved

13.15.6.16 FECC_TEST Register (Offset = 1Eh) [Reset = 0h]

FECC_TEST is shown in Figure 13-56 and described in Table 13-61.

Return to the Summary Table.

ECC Test Address

Figure 13-56. FECC_TEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RESERVED								ECC							
R-0h																R-0h								R/W-0h							

Table 13-61. FECC_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7-0	ECC	R/W	0h	8-bit ECC for selected 64-bit data. User-configurable ECC bits of the selected 64-bit data block in ECC test mode. Reset type: CM.RESETn

13.15.6.17 FECC_CTRL Register (Offset = 20h) [Reset = 0h]

FECC_CTRL is shown in Figure 13-57 and described in Table 13-62.

Return to the Summary Table.

ECC Control

Figure 13-57. FECC_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					DO_ECC_CALC	ECC_SELECT	ECC_TEST_EN
R-0h					R-0/W1S-0h	R/W-0h	R/W-0h

Table 13-62. FECC_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-3	RESERVED	R	0h	Reserved
2	DO_ECC_CALC	R-0/W1S	0h	Enable ECC calculation. ECC logic will calculate ECC in one cycle for the data and address written in ECC test registers when ECC test logic is enabled by setting ECC_TEST_EN. Reset type: CM.RESETn
1	ECC_SELECT	R/W	0h	ECC block select. 0 Selects the ECC block on bits [63:0] of bank data. 1 Selects the ECC block on bits [127:64] of bank data. Reset type: CM.RESETn
0	ECC_TEST_EN	R/W	0h	ECC test mode enable. 0 ECC test mode disabled 1 ECC test mode enabled Reset type: CM.RESETn

13.15.6.18 FOUTH_TEST Register (Offset = 22h) [Reset = 0h]

FOUTH_TEST is shown in Figure 13-58 and described in Table 13-63.

Return to the Summary Table.

Test Data Out High

Figure 13-58. FOUTH_TEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATAOUTH																															
R-0h																															

Table 13-63. FOUTH_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATAOUTH	R	0h	High double word test data out. Holds bits 63:32 of the data out of the selected ECC block. Reset type: CM.RESETn

13.15.6.19 FOUTL_TEST Register (Offset = 24h) [Reset = 0h]

FOUTL_TEST is shown in Figure 13-59 and described in Table 13-64.

Return to the Summary Table.

Test Data Out Low

Figure 13-59. FOUTL_TEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATAOUTL																															
R-0h																															

Table 13-64. FOUTL_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATAOUTL	R	0h	Low double word test data out. Holds bits 31:0 of the data out of the selected ECC block. Reset type: CM.RESETn

13.15.6.20 FECC_STATUS Register (Offset = 26h) [Reset = 0h]

FECC_STATUS is shown in Figure 13-60 and described in Table 13-65.

Return to the Summary Table.

ECC Status

Figure 13-60. FECC_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							ERR_TYPE
R-0h							R-0h
7	6	5	4	3	2	1	0
DATA_ERR_POS						UNC_ERR	SINGLE_ERR
R-0h						R-0h	R-0h

Table 13-65. FECC_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-9	RESERVED	R	0h	Reserved
8	ERR_TYPE	R	0h	Test mode ECC single bit error indicator. When 1, indicates that the single bit error is in check bits. When 0, indicates that the single bit error is in data bits (If SINGLE_ERR field is also set). Note: This bit is cleared by clearing FECC_CTRL.TEST_EN bit. Reset type: CM.RESETn
7-2	DATA_ERR_POS	R	0h	Test mode single bit error position. Holds the bit position where the single bit error occurred. The position is interpreted depending on whether the ERR_TYPE bit indicates a check bit or a data bit. If ERR_TYPE indicates a check bit error, the error position could range from 0 to 7, or it could range from 0 to 63. Note: This bit field is cleared by clearing FECC_CTRL.TEST_EN bit. Reset type: CM.RESETn
1	UNC_ERR	R	0h	Test mode ECC double bit error. When 1 indicates that the ECC test resulted in an uncorrectable bit error. Note: This bit is cleared by clearing FECC_CTRL.TEST_EN bit. Reset type: CM.RESETn
0	SINGLE_ERR	R	0h	Test mode ECC single bit error. When 1 indicates that the ECC test resulted in a single bit error. Note: This bit is cleared by clearing FECC_CTRL.TEST_EN bit. Reset type: CM.RESETn

13.15.6.21 FLASH_ECC_REGS_LOCK Register (Offset = 3Eh) [Reset = 0h]

FLASH_ECC_REGS_LOCK is shown in Figure 13-61 and described in Table 13-66.

Return to the Summary Table.

Lock register for FLASH_ECC_REGS (Not including FLASH_ECC_REGS_LOCK).

Figure 13-61. FLASH_ECC_REGS_LOCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOCK																															
R/W-0h																															

Table 13-66. FLASH_ECC_REGS_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	LOCK	R/W	0h	These bits when set to "0xA5A5A5A5" in FLASH_ECC_REGS_LOCK register enable writes to FLASH_ECC_REGS registers block (Not including FLASH_ECC_REGS_LOCK register). Any other value will disable register writes. Reset type: CM.RESETn

13.15.7 FLASH_PUMP_SEMAPHORE_REGS Registers

Table 13-67 lists the memory-mapped registers for the FLASH_PUMP_SEMAPHORE_REGS registers. All register offset addresses not listed in Table 13-67 should be considered as reserved locations and the register contents should not be modified.

Table 13-67. FLASH_PUMP_SEMAPHORE_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	PUMPREQUEST	Flash programming semaphore PUMP request register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 13-68 shows the codes that are used for access types in this section.

Table 13-68. FLASH_PUMP_SEMAPHORE_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

13.15.7.1 PUMPREQUEST Register (Offset = 0h) [Reset = 0h]

PUMPREQUEST is shown in Figure 13-62 and described in Table 13-69.

Return to the Summary Table.

Flash programming semaphore PUMP request register

Figure 13-62. PUMPREQUEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
R-0/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														SEM	
R-0-0h														R/W-0h	

Table 13-69. PUMPREQUEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	In order to write to the semaphore bits, 0x5a5a must be written to these key bits at the same time. Otherwise, writes are ignored. The key is cleared immediately after writing, so it must be written again for every semaphore change. Reset type: CPU1.SYSRSn
15-2	RESERVED	R-0	0h	Reserved
1-0	SEM	R/W	0h	These bits decide which CPU has control of the flash pump, which allows write access to the flash memory. The possible values are: 00: Read-only state. CPU1 has control of the pump, but CPU2 and CM may seize control at any time. 01: CPU2 has exclusive control of the pump and of these semaphore bits. CPU2 can relinquish control by setting the bits back to 00. 10: CPU1 has exclusive control of the pump and of these semaphore bits. CPU1 can relinquish control by setting the bits back to 00. 11: CM has exclusive control of the pump and of these semaphore bits. CM can relinquish control by setting the bits back to 00. Going from 01->10/11 or 10->01/11 or 11->01/10 is not allowed. The semaphore bits [1:0] must be written along with the correct key in bits [31:16]. Note: This field will be reset by the respective CPU resets depending on who owns the PUMP. For example if CPU2 is owning the pump, then CPU2SYSRSN would reset this field. Reset type: CPU1.SYSRSn, CPU2.SYSRSn, CM.RESETn

13.15.8 FLASH Registers to Driverlib Functions

Table 13-70. FLASH Registers to Driverlib Functions

File	Driverlib Function
FRDCNTL	
flash.h	Flash_setWaitstates
FBAC	
flash.h	Flash_setBankActiveGracePeriod
FBFALLBACK	
flash.h	Flash_setBankPowerMode
FBPRDY	
flash.h	Flash_isBankReady
flash.h	Flash_isPumpReady
FPAC1	
flash.h	Flash_setPumpPowerMode

Table 13-70. FLASH Registers to Driverlib Functions (continued)

File	Driverlib Function
flash.h	Flash_setPumpWakeupTime
FMSTAT	
-	
FRD_INTF_CTRL	
flash.h	Flash_enablePrefetch
flash.h	Flash_disablePrefetch
flash.h	Flash_enableCache
flash.h	Flash_disableCache
ECC_ENABLE	
flash.h	Flash_enableECC
flash.h	Flash_disableECC
SINGLE_ERR_ADDR_LOW	
flash.h	Flash_getSingleBitErrorAddressLow
SINGLE_ERR_ADDR_HIGH	
flash.h	Flash_getSingleBitErrorAddressHigh
UNC_ERR_ADDR_LOW	
flash.h	Flash_getUncorrectableErrorAddressLow
UNC_ERR_ADDR_HIGH	
flash.h	Flash_getUncorrectableErrorAddressHigh
ERR_STATUS	
flash.h	Flash_getLowErrorStatus
flash.h	Flash_getHighErrorStatus
flash.h	Flash_clearLowErrorStatus
flash.h	Flash_clearHighErrorStatus
ERR_POS	
flash.h	Flash_getLowErrorPosition
flash.h	Flash_getHighErrorPosition
flash.h	Flash_clearLowErrorPosition
flash.h	Flash_clearHighErrorPosition
flash.h	Flash_getLowErrorType
flash.h	Flash_getHighErrorType
ERR_STATUS_CLR	
flash.h	Flash_clearLowErrorStatus
flash.h	Flash_clearHighErrorStatus
ERR_CNT	
flash.h	Flash_getErrorCount
ERR_THRESHOLD	
flash.h	Flash_setErrorThreshold
ERR_INTFLG	
flash.h	Flash_getInterruptFlag
ERR_INTCLR	
flash.h	Flash_clearSingleErrorInterruptFlag
flash.h	Flash_clearUncorrectableInterruptFlag
FDATAH_TEST	
flash.h	Flash_setDataHighECCTest

Table 13-70. FLASH Registers to Driverlib Functions (continued)

File	Driverlib Function
FDATAL_TEST	
flash.h	Flash_setDataLowECCTest
FADDR_TEST	
flash.h	Flash_setECCTestAddress
FECC_TEST	
flash.h	Flash_setECCTestECCBits
FECC_CTRL	
flash.h	Flash_enableECCTestMode
flash.h	Flash_disableECCTestMode
flash.h	Flash_selectLowECCBlock
flash.h	Flash_selectHighECCBlock
flash.h	Flash_performECCCalculation
FOUTH_TEST	
flash.h	Flash_getTestDataOutHigh
FOUTL_TEST	
flash.h	Flash_getTestDataOutLow
FECC_STATUS	
flash.h	Flash_getECCTestStatus
flash.h	Flash_getECCTestErrorPosition
flash.h	Flash_getECCTestSingleBitErrorType

Chapter 14

Embedded Real-time Analysis and Diagnostic (ERAD)

This chapter describes the features and operation of the embedded real-time analysis and diagnostic (ERAD) module. The ERAD module enhances the debug and system analysis capabilities of the device. The debug and system analysis enhancements provided by the ERAD module are implemented external to the CPU. The ERAD module consists of the enhanced bus comparator (EBC) units, the system event counter (SEC) units, and the cyclic redundancy check (CRC) units. The EBC units are used to generate hardware breakpoints, hardware watch points, and other output events. The SEC units are used to analyze and profile the system. The CRC units monitor CPU buses and compute the CRC when the self-test code is executed. This capability aids in achieving simpler, non-intrusive and interruptible self-test mechanisms with the Software Test Library (STL). The ERAD module is accessible both by the debugger and the application software, which significantly increases the debug capabilities of many real-time systems, especially in situations where the debugger is not connected.

14.1 Introduction

The ERAD module is shown in Figure 14-1.

The ERAD enhances the debug and system analysis capabilities of the device external to the CPU. The CPU has two analysis resources; Analysis Unit 1 (AU1) and Analysis Unit 2 (AU2). The first analysis unit counts events or monitors address buses. The second analysis unit monitors address and data buses. The two analysis units can be configured for hardware breakpoints or hardware watch points, and additionally the first analysis unit can be configured as a benchmark counter or event counter. The ERAD module further expands this capability to provide additional hardware breakpoints, hardware watch points, and counters for profiling, as well as other advanced features. The ERAD module can be utilized by the debugger, and also by the application software. For many real-time systems, it is not always possible to connect a debugger and perform an intrusive debug. Under these situations, the user's code has the ability to set up and control the ERAD module to debug and profile the system without disturbing the end application.

The ERAD module consists of eight enhanced bus comparator (EBC) units and four system event counter (SEC) units. The EBC units monitor buses and generate output events. The SEC units can be used with EBC units to profile and analyze the system. These units are described in detail in the following sections.

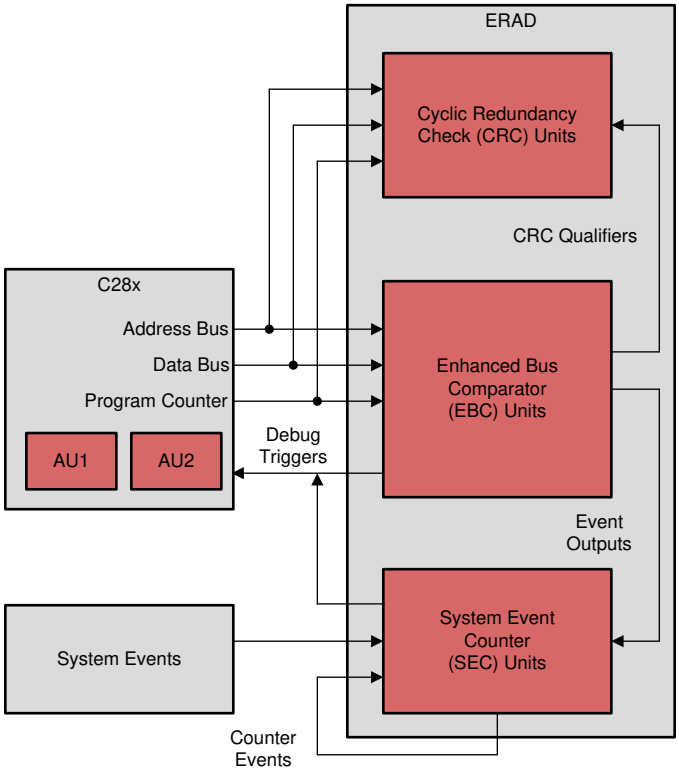


Figure 14-1. ERAD Overview

14.1.1 ERAD Related Collateral

Foundational Materials

- C2000 Academy - ERAD
- Embedded Real-Time Analysis & Diagnostics (ERAD) on C2000™ Devices (Video)

Getting Started Materials

- Embedded Real-Time Analysis & Diagnostics (ERAD) on C2000 MCUs (Video)
- *Embedded Real-Time Analysis and Response for Control Applications Application Report*

14.2 Enhanced Bus Comparator Unit

The Enhanced Bus Comparator (EBC) units connect to the CPU using a direct memory interface. This includes the program address and data buses, the data address, write and read data buses, debug qualifiers for memory access, and the ability to set breakpoints, watch points, and trace points on the CPU. Typically, the EBC is owned and controlled by the debugger application (for example, Code Composer Studio™ IDE). A user application running on the CPU can also configure and use the EBC units to generate events and interrupts for real-time diagnostic purposes. Note that ownership is exclusive—the debugger and application software cannot simultaneously control an EBC unit. For more information on EBC unit ownership, see Section 14.4.

The EBC units have the following capabilities:

- Generate hardware breakpoints
- Generate hardware watch points
- Generate trace tags for instruction fetch matches and generate real-time interrupts (RTOSINT)
- Monitor data address and read and write buses and generate real-time interrupts (RTOSINT)
- Generate event outputs that can be used by other modules.

The following features are not supported by the EBC units:

- Chained breakpoints
- Ability to monitor DMA transfers
- Ability to monitor CLA buses

Each EBC unit has the capability to monitor a range of addresses by defining masks and generating outputs based on greater than, less than, or equal events.

The EBC units can also be used with the System Event Counter (SEC) units for system or code profiling and analysis purposes.

14.2.1 Enhanced Bus Comparator Unit Operations

The following operations are supported by each EBC unit:

- Hardware Breakpoints: The EBC unit generates a breakpoint tag when the specified instruction address is accessed on the program address bus. When the instruction reaches the DECODE-2 (D2) stage of the pipeline, the CPU is halted.
- Watch Points: A watch point detects a read or write to specified locations in data memory, and halts the CPU. Unlike hardware breakpoints, watch points do not have precise timing for halting the CPU—this is entirely dependent on the current state of the CPU pipeline. The CPU halts at the next interruptible boundary.
- Program Trace: Program traces are very similar to hardware breakpoints. The difference here is that instead of halting the CPU, a program trace generates a real-time interrupt (RTOSINT) when the instruction reaches the D2 stage of the pipeline. If the instruction is discarded in the fetch buffer due to discontinuity, no RTOSINT is generated.
- Data Trace: A data trace is similar to a watch point, except that a data trace generates a real-time interrupt (RTOSINT) instead of halting the CPU on an access to the specified data memory.

Note that hardware breakpoints only halt the CPU if a debugger is connected.

14.2.2 Event Masking and Exporting

The events generated by different EBC units can be combined using OR and AND logic to generate new events as required. There are four AND and four OR combinations that can be exported using masks to suppress undesired events. These events can be configured to generate an RTOSINT. In addition, the AND and OR events can also be used to qualify CRC computation using the cyclic redundancy check unit. The AND and OR events are also available as inputs to the system event counter unit for event counting and system profiling.

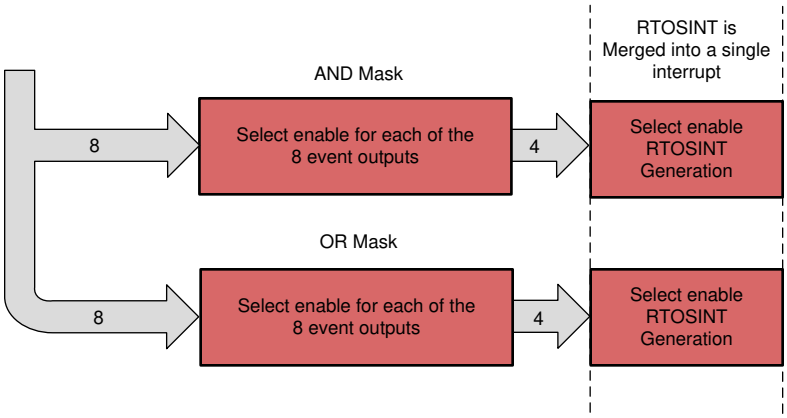


Figure 14-2. EBC Units Event Masking

To use the AND and OR masks:

1. Configure the GLBL_EVENT_AND_MASK and GLBL_OR_EVENT_MASK registers to select the desired EBC unit outputs for any of the available four masks.
2. To enable an RTOSINT for the configured mask, write 1 to the corresponding bit in the GLBL_AND_EVENT_INT_MASK or GLBL_OR_EVNT_INT_MASK register.
3. To use the mask output as an input to the system event counter unit, configure the CTM_INPUT_SEL register with the mux value for the desired mask. The input mux values are listed in Section 14.3.1.4.
4. To use the mask output as a qualifier to the CRC unit, configure the CRC_QUALIFIER register.

For example, to generate a real-time interrupt on MASK1 when EBC units 2, 3, AND 6 events are triggered, write 0x46 to GLBL_EVENT_AND_MASK, and then write 0x1 to GLBL_AND_EVENT_INT_MSK.

14.3 System Event Counter Unit

The SEC units provide system profiling, analysis, and debug capability. The SEC units contain counters that can enhance the debug and profiling process in various types of system scenarios such as:

- Profiling code segments
- Counting duration between specified memory reads and writes
- Counting system events (such as interrupts)
- Counting duration between system events
- System timer
- Measuring the number of wait states in code segments
- Measuring the maximum amount of time spent in between a pair of events, measured over multiple iterations
- Chaining counters to link events or create larger counters

Furthermore, the SEC unit has the capability to:

- Function as a counter capable of counting:
  - Any of the match events generated by the EBC units.
  - Events generated by the EBC units. These events can be used to start and stop the counting.
  - System events including the PIE interrupts, timer interrupts, and CLA task interrupts. These system events can be used to start and stop the counting.
  - More information on the input sources for the SEC units can be found in Section 14.3.1.4.
- Generate an interrupt or a watch point if the count reaches a reference value.
- Perform counter operation in one of the following two modes:
  - Duration mode: The counter counts the CPU cycles as long as the event is active.
  - Event mode: The counter counts only the positive edge of the event signal. This is effectively counting the number of times the event transitions from inactive to active.

14.3.1 System Event Counter Modes

The following are the operating modes of the SEC unit. The counters are initialized to zero when the SEC module receives a reset input signal, and always count up.

- Continuous Count: In this mode, the counter continues to count as specified by the input selector. The counter can count the CPU cycles without any events selected. In this mode, the module can be used as a software-controlled SYSCLK counter. Continuous mode is active when CTM_CNTL.START_STOP_MODE and CTM_REF are both set to 0.
- Timer Mode Count: In this mode, the counter counts up to a set reference value, defined in the CTM_REF register. Upon reaching the reference value, the counter generates an event that can send an interrupt to the CPU or generate a watch point. The RST_ON_MATCH bit in the CTM_CNTL register configures the counter to either continue incrementing or reset when a match event occurs.
- Start-Stop Count: In this mode, two events are configured to act as start and stop indicators to the counter. The counter commences counting only when the defined start event occurs. The counter then continues to count up until the stop event occurs. Once the first start event has occurred, further start events are ignored until the stop event occurs.

In any of the counter modes of operation, there is a possibility that the 32-bit counter value overflows. If an overflow occurs, the counter value resets to zero and continues to count up, and the OVERFLOW bit in the CTM_STATUS register is set high. The OVERFLOW bit remains high until either the counter is reset, or the application writes 1 to the OVERFLOW_CLEAR bit of the CTM_CLEAR register.

14.3.1.1 Counting Active Levels Versus Edges

The SEC units can be configured to either count active levels or edges of the selected inputs.

Each SEC unit has eight inputs from the EBC units and many inputs from other events in the device. Each SEC unit can be configured to count any of the input events or just count up on every cycle. For example, if an input event occurs and is active for 25 cycles, the SEC unit counter increments only by 1 in event mode; whereas in the duration mode, the counter increments by 25.

14.3.1.2 Max Mode

Max mode is also supported by the SEC units. This mode allows the user to detect the maximum count that has occurred during various count iterations in start-stop mode. For example, a user can set up the counter in the start-stop count mode to count the duration of a critical code loop. Every time the stop event occurs and the counter stops, the counter value is checked against the current MAX_COUNT present in the register. If the new value is greater, then the MAX_COUNT register is updated. The counter always resets to zero at the stop event and is ready to start counting on the next start event. Therefore, the MAX_COUNT contains the maximum number of cycles that occurred between the start and stop condition over many iterations.

14.3.1.3 Cumulative Mode

The SEC units can be used to yield the cumulative count over several start and stop events. In this mode the, unlike Max mode, the counter does not reset due to a stop event. Instead it stops counting and resumes counting when a start event occurs. in cumulative count mode, the MAX_COUNT is not valid.

14.3.1.4 Input Signal Selection

The SEC inputs can be selected from various signals from in the system to enable debug and system analysis. Figure 14-3 shows the SEC inputs. Each event selector MUX can select from various signals on in the system. These signals are shown in Table 14-1.

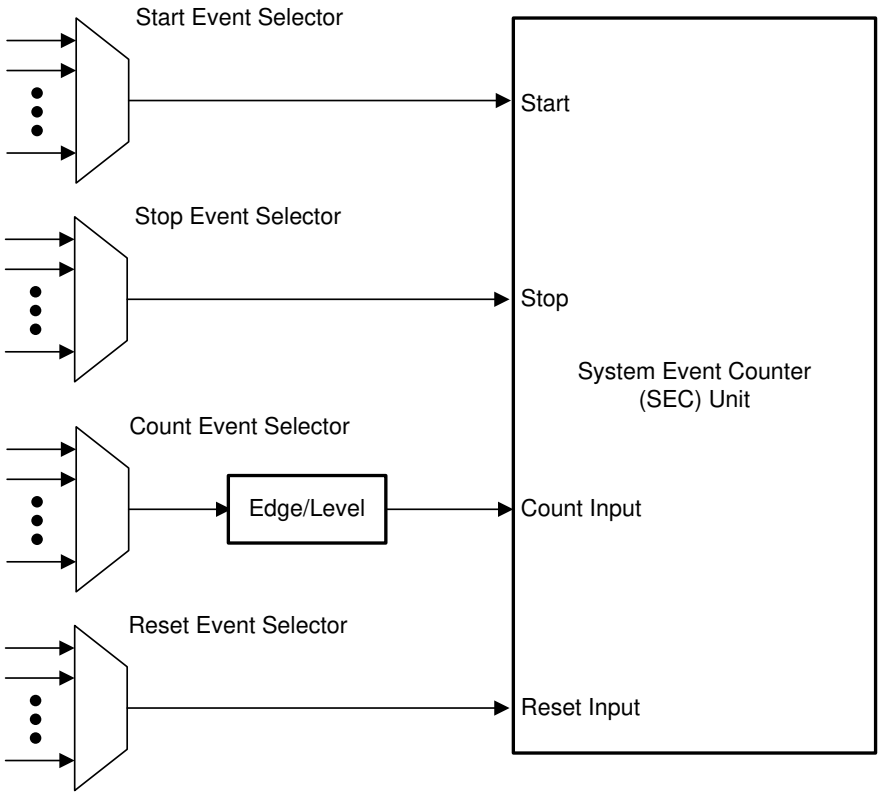


Figure 14-3. System Event Counter Inputs

Table 14-1. Event Selector Mux Signals

CTM/STA/STO/RST_INP_SEL	EVENT_INPUT_SELECTED	Polarity	Synchronization Requirement
0	EBC1	High	Disable
1	EBC2	High	Disable
2	EBC3	High	Disable
3	EBC4	High	Disable
4	EBC5	High	Disable
5	EBC6	High	Disable
6	EBC7	High	Disable
7	EBC8	High	Disable
8	COUNTER1_EVENT	High	Disable
9	COUNTER2_EVENT	High	Disable
10	COUNTER3_EVENT	High	Disable
11	COUNTER4_EVENT	High	Disable
12	ERAD_OR_MASK0	High	Disable
13	ERAD_OR_MASK1	High	Disable
14	ERAD_OR_MASK2	High	Disable
15	ERAD_OR_MASK3	High	Disable
16	ERAD_AND_MASK0	High	Disable
17	ERAD_AND_MASK1	High	Disable
18	ERAD_AND_MASK2	High	Disable
19	ERAD_AND_MASK3	High	Disable
20	PIE_INT1	High	Disable
21	PIE_INT2	High	Disable
22	PIE_INT3	High	Disable
23	PIE_INT4	High	Disable
24	PIE_INT5	High	Disable
25	PIE_INT6	High	Disable
26	PIE_INT7	High	Disable
27	PIE_INT8	High	Disable
28	PIE_INT9	High	Disable
29	PIE_INT10	High	Disable
30	PIE_INT11	High	Disable
31	PIE_INT12	High	Disable
32	CPU1_TINT0	High	Disable
33	CPU1_TINT1	High	Disable
34	CPU1_TINT2	High	Disable
35	CLA_INTERRUPT1	High	Disable
36	CLA_INTERRUPT2	High	Disable
37	CLA_INTERRUPT3	High	Disable
38	CLA_INTERRUPT4	High	Disable
39	CLA_INTERRUPT5	High	Disable
40	CLA_INTERRUPT6	High	Disable
41	CLA_INTERRUPT7	High	Disable
42	CLA_INTERRUPT8	High	Disable
43	ECAT_PDI_SOF	High	Disable

Table 14-1. Event Selector Mux Signals (continued)

CTM/STA/STO/RST_INP_SEL	EVENT_INPUT_SELECTED	Polarity	Synchronization Requirement
44	ECAT_PDI_EOF	High	Disable
45	ECAT_PCI_WD_TRIGGER	High	Disable
46	ECAT_PDI_uc_IRQ	High	Disable
47	ECAT_SYNC_OUT0	High	Disable
48	ECAT_SYNC_OUT1	High	Disable
49	ECAT_DRAM_PARITY_ERROR	High	Disable
50	MCANA_EVT0	High	Disable
51	MCANA_EVT1	High	Disable
52	MCANA_EVT2	High	Disable
53	ADCSOCAO	High	Disable
54	ADCSOCBO	High	Disable
55	CLATASKRUN1	High	Disable
56	CLATASKRUN2	High	Disable
57	CLATASKRUN3	High	Disable
58	CLATASKRUN4	High	Disable
59	CLATASKRUN5	High	Disable
60	CLATASKRUN6	High	Disable
61	CLATASKRUN7	High	Disable
62	CLATASKRUN8	High	Disable
63	EPWMXBAR1	Low	Enable
64	EPWMXBAR2	Low	Enable
65	EPWMXBAR3	Low	Enable
66	EPWMXBAR4	Low	Enable
67	EPWMXBAR5	Low	Enable
68	EPWMXBAR6	Low	Enable
69	EPWMXBAR7	Low	Enable
70	EPWMXBAR8	Low	Enable
71	INPUTXBAR1	High	Enable
72	INPUTXBAR2	High	Enable
73	INPUTXBAR3	High	Enable
74	INPUTXBAR4	High	Enable
75	INPUTXBAR5	High	Enable
76	INPUTXBAR6	High	Enable
77	INPUTXBAR7	High	Enable
78	INPUTXBAR8	High	Enable
79	INPUTXBAR9	High	Enable
80	INPUTXBAR10	High	Enable
81	INPUTXBAR11	High	Enable
82	INPUTXBAR12	High	Enable
83	INPUTXBAR13	High	Enable
84	INPUTXBAR14	High	Enable
85	INPUTXBAR15	High	Enable
86	INPUTXBAR16	High	Enable
87	CPUx_CPUSTAT	High	Disable

Table 14-1. Event Selector Mux Signals (continued)

CTM/STA/STO/RST_INP_SEL	EVENT_INPUT_SELECTED	Polarity	Synchronization Requirement
88	CPUx_DBGACK	High	Disable
89	CPUx_NMI	High	Disable
90	CMPSS1_CTRIPH_OR_CTRIPL	High	Enable
91	CMPSS2_CTRIPH_OR_CTRIPL	High	Enable
92	CMPSS3_CTRIPH_OR_CTRIPL	High	Enable
93	CMPSS4_CTRIPH_OR_CTRIPL	High	Enable
94	CMPSS5_CTRIPH_OR_CTRIPL	High	Reserved
95	CMPSS6_CTRIPH_OR_CTRIPL	High	Disable
96	CMPSS7_CTRIPH_OR_CTRIPL	High	Disable
97	CMPSS8_CTRIPH_OR_CTRIPL	High	Disable
98	SD1FLT1_COMPH_OR_COMPL	High	Disable
99	SD1FLT2_COMPH_OR_COMPL	High	Disable
100	SD1FLT3_COMPH_OR_COMPL	High	Disable
101	SD1FLT4_COMPH_OR_COMPL	High	Disable
102	SD2FLT1_COMPH_OR_COMPL	High	Disable
103	SD2FLT2_COMPH_OR_COMPL	High	Disable
104	SD2FLT3_COMPH_OR_COMPL	High	Disable
105	SD2FLT4_COMPH_OR_COMPL	High	Disable
106	ADCAINT1	High	Disable
107	ADCAINT2	High	Disable
108	ADCAINT3	High	Disable
109	ADCAINT4	High	Disable
110	ADCBINT1	High	Disable
111	ADCBINT2	High	Disable
112	ADCBINT3	High	Disable
113	ADCBINT4	High	Disable
114	ADCCINT1	High	Disable
115	ADCCINT2	High	Disable
116	ADCCINT3	High	Disable
117	ADCCINT4	High	Disable
118	ADCDINT1	Reserved	Enable
119	ADCDINT2	High	Disable
120	ADCDINT3	High	Disable
121	ADCDINT4	Reserved	Reserved
122-127	Reserved	Reserved	Reserved

14.3.2 Reset on Event

Resetting the counters on external events is also possible. Additionally all the counter event outputs are applied back to each of counter input MUX, which selects the event that can be used as a reset input. When enabled, an active high on the reset input causes the counter to reset. This gives a powerful feature that allows setting up threshold monitors. This can be used to flag an interrupt or a watch point, if the distance between two events crosses a certain threshold.

14.3.3 Operation Conditions

The SEC units count accurately only when the CPU is operating in normal conditions. If the counters are running and capturing the CPU cycles while the CPU is controlled through the debugger to single-step through the code, then the result can differ from when the CPU was executing the code in normal conditions.

Note that if the counters are set up to use the value of the raw program counter register (VPC) as the source for the start and stop events, the value of the CPU cycles measured can be off by a few cycles when the CALL instruction is executed.

14.4 ERAD Ownership, Initialization and Reset

Although the features of the ERAD module are typically used by the debugger, user applications can also take advantage of the capabilities to monitor buses and generate interrupts and events. There are three possible ownership scenarios:

1. The user elects to completely hand over the ownership of the ERAD module to the application software or the debugger.
2. Both the application code and the debugger share use of the ERAD module. Only the current owner of the module (application code or debugger) is allowed to use the module at a given time. When ownership is shared between application and debugger, the user application has the responsibility of resolving any ownership conflicts.
3. There is no ERAD owner. In this mode, both application code and debugger can access the module at any given time. It is critical for the software, both on the application side and the debugger side, to resolve any potential conflicts. An example scenario in this mode can be for the debugger to use some of the EBC and SEC units, while the application software uses the remaining units.

The ERAD module initializes the internal states and all registers to their initial/reset states under the following conditions:

- At power-on-reset (POR)
- With DCON and SYSRSN
  - Debug logic disconnected when the debugger owns the module
  - Functional reset when application owns the module

14.5 ERAD Programming Sequence

The ERAD module can be used to set hardware breakpoints and hardware watch points. The programming sequences to set hardware breakpoints, hardware watch points, or to use the timers to profile and analyze the system are described in this section. The same sequences can be used by both the debugger software and user application code.

Refer to the Driverlib example projects in C2000Ware for JavaScript files to configure the ERAD module. Example projects are also available to showcase the usage of these script files. These examples can be found in the driverlib\f2838x\examples\c28x\erad directory under the C2000Ware installation directory.

14.5.1 Hardware Breakpoint and Hardware Watch Point Programming Sequence

The programming sequence is identical when using the EBC units, regardless of whether the debug software or the application is programing the units. A typical programming sequence for a unit is:

- Read and make sure the ownership is set as expected; if not, acquire the ownership before proceeding further if required.
- Make sure the unit is in IDLE mode.
- Set up the address reference, mask, bus select and stop bits.
- Enable the corresponding module bit in the global enable register.
- Once the usage is completed, write 1 to clear the EVENT_FIRED sticky bit. This takes the module back to the enabled state.

The example programming sequences for hardware breakpoints and hardware watch points are:

Set a hardware breakpoint on address 0x201000:

- Read GLBL_OWNER to confirm ownership.
- Read HWBP_STATUS to confirm the module is in IDLE state.
- Write 0x0 to HWBP_MASK.
- Write 0x201000 to HWBP_REF.
- Set HWBP_CNTL.STOP = 1 and HWBP_CNTL.BUS_SEL = 0000 (PAB). If a trace is to be generated instead of a breakpoint, set HWBP_CNTL.STOP = 0.
- Enable the corresponding module bit in the global enable register.

Set a hardware watch point on read of addresses from 0x121010 to 0x12101F:

- Read GLBL_OWNER to confirm ownership.
- Read HWBP_STATUS to confirm the module is in IDLE state.
- Write 0xF to HWBP_MASK.
- Write 0x121010 to HWBP_REF.
- Write HWBP_CNTL.STOP = 1 and HWBP_CNTL.BUS_SEL = 0011 (DRAB). If an RTOSINTn is to be generated instead of a watch point, set HWBP_CNTL.STOP = 0.
- Enable the corresponding module bit in the global enable register.

Set a hardware watch point on write to address 0xFF10101A:

- Read GLBL_OWNER to confirm ownership
- Read HWBP_STATUS to confirm the module is in IDLE state
- Write 0x0 to HWBP_MASK
- Write 0xFF10101A to HWBP_REF
- Write HWBP_CNTL.STOP = 1, HWBP_CNTL.BUS_SEL = 0010 (DWAB). If an RTOSINTn is to be generated instead of a watch point, set HWBP_CNTL.STOP = 0.
- Enable the corresponding module bit in the global enable register.

14.5.2 Timer and Counter Programming Sequence

The programming sequence is identical when using the SEC units, regardless of whether the debug software or the application is programing the units. Typical programming sequence for a unit is:

- Read and make sure the ownership is set as expected. If not, acquire the ownership before proceeding further if required.
- Make sure the unit is in IDLE mode.
- Set up the counter reference, counter registers (clear/reset if a clean start is required) and CTM_CNTL.
- Enable the corresponding module bit in the global enable register.
- Once the usage is completed, write 1 to clear the EVENT_FIRED sticky bit. This takes the module back to the enabled state.

Set up a free running counter:

- Read and make sure the ownership is set as expected. If not, acquire the ownership before proceeding further, if required.
- Read CTM_STATUS to confirm that the module is in IDLE state.
- Write 0x0 to CTM_INPUT_SEL.
- Write 0x0 to CTM_CNTL.
- Enable the module in the GLBL_ENABLE register.

Set up the counter to count the duration spent between addresses 0x1000 and 0x1210:

- Read and make sure the ownership is set as expected. If not, acquire the ownership before proceeding further, if required.
- Read CTM_STATUS to confirm that the module is in IDLE state.
- Set up the EBC unit 1 to generate an event for VPC = 0x1000.
- Set up the EBC unit 2 to generate an event for VPC = 0x1210.
- Set up the start and stop input selects to use comparator event 1 and 2, respectively. This is achieved by writing CTM_INPUT_SEL.STA_INP_SEL = 0x0 and CTM_INPUT_SEL_2.STO_INP_SEL = 0x1.
- Enable the counter in the START_STOP_MODE of operation. This is achieved by writing CTM_CNTL. START_STOP_MODE = 1.
- Enable the module in the GLBL_ENABLE register.

Set up the counter to count the number of times a function at address 0x2010 is called and fire an RTOSINT if this count reaches 0x300:

- Read and make sure the ownership is set as expected. If not, acquire the ownership before proceeding further, if required.
- Read CTM_STATUS to confirm that the module is in IDLE state.
- Set up the EBC unit 1 to generate an event for VPC = 0x2010.
- Setup the counter to select comparator event 1 as the event to count. This is achieved by writing CTM_INPUT_SEL.CNT_INP_SEL= 0 and CTM_CNTL.CNT_INP_SEL_EN = 1.
- Write 0x300 CTM_REF.
- Enable the counter in the EVENT_MODE, and also allow the counter to generate an RTOSINT when the count matches the reference. This is achieved by writing 0x88 to CTM_CNTL (bit 3 = 1 and bit 7 = 1).
- Enable the module in the GLBL_ENABLE register.

14.6 Cyclic Redundancy Check Unit

The cyclic redundancy check (CRC) units monitor CPU buses and compute CRC when the self-test code is executed. This capability aids in achieving simpler, non-intrusive and interruptible self-test mechanisms with Software Test Library (STL). Each CRC unit is used to monitor a different CPU interface. For example, CRC unit 1 is used to monitor the Program Counter, while CRC unit 2 is used to monitor the data read address bus. Table 14-2 identifies the CPU interface monitored by each of the CRC units.

Table 14-2. CPU Interfaces Monitored by CRC Units

CRC Unit	CPU Interface
CRC Unit 1	Program Counter Register
CRC Unit 2	Data Read Address Bus
CRC Unit 3	Data Read Data Bus
CRC Unit 4	Data Write Address Bus
CRC Unit 5	Data Write Data Bus
CRC Unit 6	Instruction Register Value (Unsecured)
CRC Unit 7	Instruction Register Value (Secure-Zone 1)
CRC Unit 8	Instruction Register Value (Secure-Zone 2)

The main purpose of the CRC units is to ensure that the CPU functionally remains intact when it is executing the same software test library over multiple iterations. This is done by comparing the computed CRC values after each iteration, with a pre-computed golden value.

CRC units 7 and 8 are intended to compute the Instruction Register values for secure-zone 1 and secure-zone 2 instruction execution. Computed CRC values for the a given secure-zone is available only for accesses originating from that zone.

14.6.1 CRC Unit Qualifier

By default, all the valid events on a given interface enables a CRC unit for computation. However there are optional qualifiers that can be used to gate the CRC computation when valid events occur. If required, these qualifiers can be generated by masking the EBC units events. The AND/OR masks discussed in Section 14.2.2 are used to generate these qualifiers. Refer to the CRC_QUALIFIER register for a full list of available qualifiers. Figure 14-4 shows the connection between EBC event masking and exporting with the CRC qualifiers.

EBC units have the capability to monitor the Program Counter, data writes and data reads. CRC units can use the EBC units to decide when the check values can be calculated. For example, if required to calculate the check values only when the CPU is executing a specific function, the user can set up an EBC unit to monitor the PC and generate a CRC qualifier when that function is executed. This allows the CRC unit to calculated the check value only when desired.

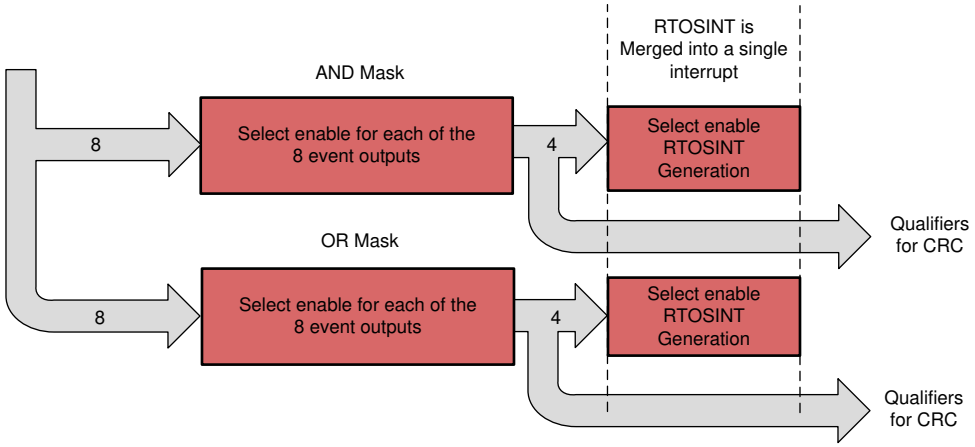


Figure 14-4. Event Masking and Exporting for CRC Qualifiers

14.6.2 CRC Unit Programming Sequence

The following sequence can be used to initialize a CRC unit to calculate the check value for the corresponding CPU interface.

1. Initialize the CRC unit by writing a 1 to the corresponding CRC_INIT field in the CRC_GLOBAL_CTRL register.
2. Configure the seed value (if required) using CRC_SEED register (CRC_EN can be 0 for when modifying the CRC_SEED register).
3. Configure the CRC_QUALIFIER register if additional qualification from EBC units is required; If not additional qualification is not required, set the CRC_QUALIFIER register to 0.
4. Enable the CRC unit by setting the CRC_EN to 1 at the beginning of the software for which the CRC calculation is done.
5. Disable the CRC unit by setting the CRC_EN to 0 at the end of the software for which the CRC calculation is done.
6. Read the CRC_CURRENT register to record the computed CRC. This check value can be compared with previous check values to make sure no changes have occurred.
7. Repeat steps 1-7 periodically as needed.

Note

Sufficient NOP's must be added immediately after the CRC submodule is enabled to make sure the pipeline contains predictable code as soon as the CRC is enabled. Similarly, sufficient NOP's must be added before the CRC submodule is disabled. Disabling the CRC write access takes a few cycles, therefore there must be predictable code in the pipeline stages until the write takes place.

14.7 Program Counter Trace

The Program Counter (PC) Trace module can be used to keep track of PC discontinuities or jumps, as a means to trace an entire program execution sequence over a period of time. When trace is completed or stopped, trace data can be read out using a debugger and analyzed to reconstruct the code execution sequence. The PC Trace module provides multiple modes and controls to govern when to trace and when not to trace. The trace module is tightly coupled with the Enhanced Bus Comparator, the System Event Counter Unit, and certain critical system level event signals.

Trace data is stored in an addressable memory buffer that can be read by software or a debugger. The trace data stored in this buffer includes additional status information on trace validity that can be used to correctly reconstruct the code execution system. For each discontinuity, two PC values are stored: the source of the discontinuity, and the destination. Figure 14-5 illustrates the operation of the PC trace module. The trace buffer is a circular buffer with overflow: when the buffer becomes full, new trace data is written starting from the top of the buffer, and an overflow status bit is set.

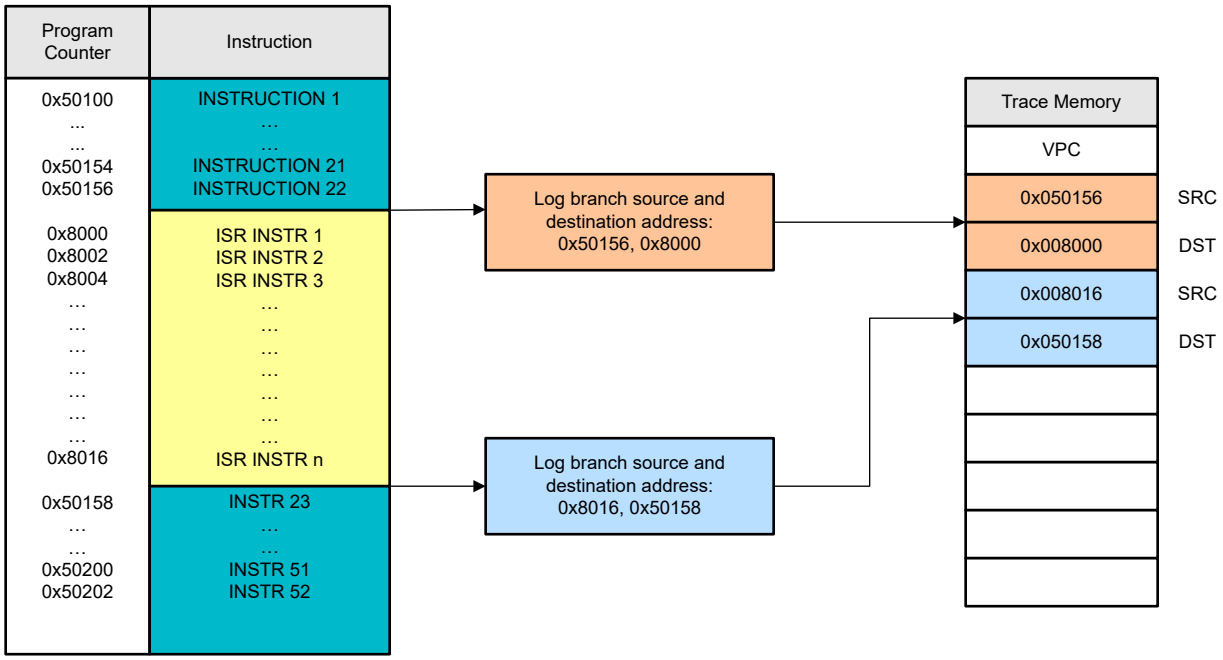


Figure 14-5. PC Trace Operation

14.7.1 Functional Block Diagram

Figure 14-6 describes the device PC trace architecture. The trace module is tightly coupled with the CPU, and receives the current program counter (VPC), program address (PAB), and various qualifying signals from the CPU interface. The Trace core captures these values whenever a PC discontinuity (for example, branch operation) is detected. The DCSM also interfaces with the PC Trace module, providing security zone data to prevent unauthorized trace information from being exposed in trace memory. The PC Trace module interfaces with the Enhanced Bus Comparator Unit and System Event Counter Unit, providing the ability to select events from these units as triggers to start a trace, stop a trace, or determine the bounding conditions for a windowed trace operation.

The Trace Core qualifies trace source and destination addresses, and stores these addresses sequentially in the trace memory buffer. Additionally, the Trace Core can generate hit events every time a new trace is stored in the memory buffer; this event signal is connected to the ERAD counter block so that the number of entries in the buffer can be tracked. This counter value can in turn be used to create a STOP event at a predefined threshold.

Note

Discontinuities that arise due to block repeats (RPTB instruction blocks) are not captured by the PC trace module.

Debugger accesses are always treated as unsecure accesses. Unlike other ERAD components, there is no concept of debug or application ownership in the PC Trace module.

The Trace Core generates a trace hit event for a discontinuity arising from a speculative instruction fetch, even if the fetched instruction does not reach the execution phase in the CPU pipeline. As a result, the BUFFER_FULL signal can be set prematurely due to a speculative prefetch. The user can always safely discard the oldest discontinuity pair present in the memory buffer when a full buffer is detected, to mitigate this scenario.

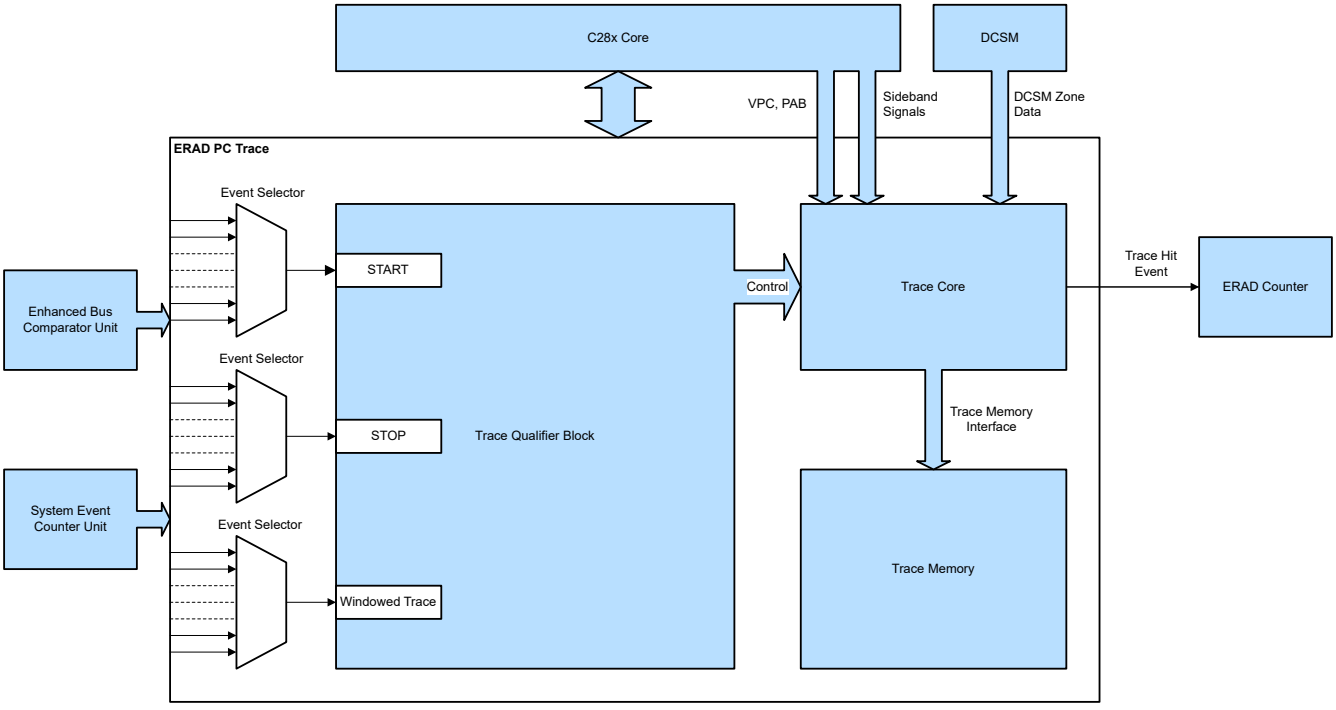


Figure 14-6. PC Trace Block Diagram

14.7.2 Trace Qualification Modes

There are three modes of trace qualification available:

1. Normal mode: In this mode, trace is enabled without any qualifiers. Trace control happens purely through software writing to the PCTRACE_GLOBAL register. When operating PC Trace in normal mode, The PCTRACE_LOGPC_SOFTENABLE captures the program counter value at the point when PCTRACE_GLOBAL.EN is set to 1, and the PCTRACE_LOGPC_SOFTDISABLE register captures the program counter value at the point PCTRACE_GLOBAL.EN is set to 0.
2. Windowed Trace mode: In windowed mode, PC Trace can be activated or stopped based on the value of a signal coming from the Event Bus Comparator, System Event Counter, or other system events. The WINDOWED_INP_SEL field in the PCTRACE_QUAL1 register specifies the input signal used to qualify PC Trace operation in windowed mode. By default trace is active when the input signal is high and inactive when the input signal is low; this behavior can be reversed by setting the PCTRACE_QUAL1.WINDOWED_INP_INV bit high.
3. Start-Stop mode: In this mode, one input signal starts the PC trace operation, and a different input signal stops the trace operation. Event Bus Comparator signals, System Event Counter signals and other system event signals can be used as inputs to start or stop the PC Trace. The START_INP_SEL and STOP_INP_SEL fields in the PCTRACE_QUAL2 register specify the input signals used to qualify PC Trace operation in start-stop mode. Once a start event arrives and PC trace operation begins, the PC Trace module ignores all further start events until a stop event is received. Trace start and stop operations are triggered on the rising edge of the input event; this behavior can be reversed by setting the START_INP_INV and STOP_INP_INV bits in the PCTRACE_QUAL1 register.

To set the PC Trace operation mode, write to the TRACE_MODE bit in the PCTRACE_QUAL1 register.

14.7.3 Trace Memory

PC Trace memory is a 32-bit-wide read-only memory buffer which holds each 22-bit PC value, together with a security BLOCKED status bit. The memory map section of the device data manual specifies the size of the PC Trace memory buffer. Trace memory entries are stored in pairs: the discontinuity source and destination addresses. Table 14-3 describes the bit field structure of each trace memory entry.

Table 14-3. Trace Memory Entry Bit Fields

Bits	Field Name	Description
[21:0]	PROGRAM_COUNTER	Program counter source or destination address value where discontinuity occurred
[22]	BLOCKED	1 = PROGRAM_COUNTER[21:0] is invalid due to security permissions; 0 = PROGRAM_COUNTER[21:0] is valid
[31:23]	RESERVED	Reserved

Note

For addresses that are blocked due to security zone restrictions, the PROGRAM_COUNTER value is set to 0.

Trace memory is intended solely for debug purposes and does not have parity or Error Correction Code (ECC) support.

14.7.4 Trace Input Signal Conditioning

The PC Trace module provides input conditioning options for signals that are used to qualify operation in Start-Stop or Windowed modes. For each of these (START, STOP, WINDOWED), there is a two-stage synchronizer for asynchronous input signals, and an inverter. The PC Trace input conditioning options are controllable via the PCTRACE_QUAL1 register using the following bits:

- WINDOWED_INP_INV: This bit inverts the input signal for Windowed trace mode selected in WINDOWED_INP_SEL. When set to 1, the trace operation starts on the falling edge of the input, and stops the rising edge of the input. When this bit is set to 0, the trace operation starts on the rising edge of the input, and stops on the falling edge of the input.
- WINDOWED_INP_SYNCH: This bit passes the input signal selected in WINDOWED_INP_SEL through a two-stage synchronizer.
- START_INP_INV: This bit inverts the input signal for trace START operation selected in PCTRACE_QUAL2.START_INP_SEL. When set to 1, the trace operation starts on the falling edge of the input. When this bit is set to 0, the trace operation starts on the rising edge of the input.
- START_INP_SYNCH: This bit passes the input signal selected in PCTRACE_QUAL2.START_INP_SEL through a two-stage synchronizer.
- STOP_INP_INV: This bit inverts the input signal for trace STOP operation selected in PCTRACE_QUAL2.STOP_INP_SEL. When set to 1, the trace operation stops on the falling edge of the input. When this bit is set to 0, the trace operation stops on the rising edge of the input.
- STOP_INP_SYNCH: This bit passes the input signal selected in PCTRACE_QUAL2.STOP_INP_SEL through a two-stage synchronizer.

Figure 14-7 describes the signal conditioning circuit available for each input qualifier signal.

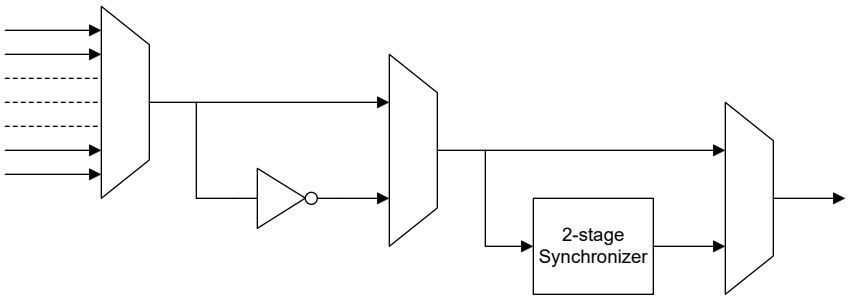


Figure 14-7. Trace Qualifier Input Conditioning Circuit

14.7.5 PC Trace Software Operation

An example software sequence to perform PC Trace operation is as follows:

1. Initialize the PC Trace module by writing 1 to PCTRACE_GLOBAL.INIT. The trace initialize operation resets the buffer pointer and overflow flags, and clears the values of the PCTRACE_LOGPC_SOFTENABLE and PCTRACE_LOGPC_SOFTDISABLE registers.
2. Start the PC Trace operation by writing 1 to PCTRACE|_GLOBAL.EN.
3. Execute the desired code sequence to be profiled.
4. Stop the PC Trace operation by writing 0 to PCTRACE_GLOBAL.EN. This step is optional, as the PC Trace buffer can be read while trace operation is active.
5. To determine if there are valid entries in the trace buffer and how many:
   a. Examine the buffer pointer by reading PCTRACE_BUFFER.PTR. If the pointer value is 0, then no discontinuities have been detected since PC Trace was initialized. A non-zero value indicates the number of locations in the buffer that contain valid entries. For instance, PTR=4 indicates that locations 0, 1, 2 and 3 contain valid entries (SRC, DST, SRC, DST).
   b. Examine the BUFFER_FULL bit in the PCTRACE_BUFFER register. If BUFFER_FULL=1, then the following possibilities apply:
      - If PTR = 0, then the buffer is simply full and contains the maximum number of entries possible.
      - If PTR > 0, then a buffer overflow has occurred. The value of PTR indicates by how many entries the buffer has overflowed: this is a circular buffer.
6. Read PCTRACE_LOGPC_SOFTENABLE (and optionally PCTRACE_LOGPC_SOFTDISABLE) if needed to determine the bounding addresses of the trace operation, in case the code sequence being traced does not begin or end at a discontinuity boundary (for example, partial function trace).
7. Trace discontinuities are recorded in the trace buffer in pairs (SRC, DST). Use this data to reconstruct the code execution sequence. While reading the trace buffer, be sure to examine the BLOCKED status bit to confirm the validity of each trace entry.

14.7.6 Trace Operation in Debug Mode

PC Trace is designed to capture data while the CPU is executing code. Debugger operations such as halt, step, and manual PC modification can compromise the reliability of PC trace data collection. Only use or interpret PC Trace data in the context of a continuous CPU run without debugger interruption or intervention.

CPU execution does not preclude debugger operation of the PC trace module. While the CPU is running and executing code, the debugger can read and write PC Trace registers, read the trace buffer, and enable or disable PC trace operation. The PCTRACE_LOGPC_SOFTENABLE and PCTRACE_LOGPC_SOFTDISABLE registers record the start and stop PC addresses to provide accurate trace window information during manual trace starts or stops triggered by writing to the PCTRACE_GLOBAL.EN bit.

14.8 Software

14.8.1 ERAD Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/erad

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

14.8.1.1 ERAD Profiling Interrupts

FILE: erad_ex1_profileinterrupts.c

This example configures CPU Timer0, 1, and 2 to be profiled using the ERAD module. Included is a JavaScript file, profile_interrupts.js, which is used with the scripting console to program ERAD registers and view profiling data.

To properly use the provided ERAD script, the following variables must be set in the scripting environment prior to launching the ERAD script:

- var PROJ_NAME = "erad_debugger_ex1_profileinterrupts"
- var PROJ_WKSPC_LOC = "<proj_workspace_path>"
- var PROJ_CONFIG = "<name of active configuration [CPU1_FLASH|CPU1_RAM]>"

To run the ERAD script, use the following command in the scripting console:

- loadJSFile("<proj_workspace_path>\\erad_debugger_ex1_profileinterrupts\\erad_ex1_profile_interrupts.js", 0);

The included JavaScript file, erad_ex1_profile_interrupts.js, uses Debug Server Scripting (DSS) features. For information on using the DSS, please visit: *http://software-dl.ti.com/ccs/esd/documents/users_guide/sdto_dss_handbook.html*

Note that the script must be run after loading and running the .out on the C28x core. Only CPU timer 2 ISR is profiled in this example.

This example uses 2 HW breakpoints and 4 counters:
- HWBP_1 : PC = start address of cpuTimer2ISR
- HWBP_2 : PC = end address of cpuTimer2ISR
- CTM_1 : Used to count the cpuTimer2ISR execution cycles. Configured in start-stop mode with start event as HWBP_1 and stop event as HWBP_2
- CTM_2 : Used to count the number of times the system event TIMER2_TINT2 has occurred. Configured in rising-edge count mode with counting input as system event TIMER2_TINT2 (INP_SEL[25])
- CTM_3 : Used to count the number of times cputTimer2ISR executes. Configured in rising-edge count mode with counting input as HWBP_1 (INP_SEL[0])
- CTM_4 : Used to count the latency from the system event TIMER2_TINT2 to cpuTimer2ISR entry. Configured in start-stop mode with start event as TIMER2_TINT2 and stop event as HWBP_1

External Connections
- None

Watch Variables
- cpuTimer0IntCount
- cpuTimer1IntCount
- cpuTimer2IntCount

Profiling Script Output
- Current ISR cycle count (CTM_1)
- Max ISR cycle count (maximum value of CTM_1)
- Interrupt occurrence count (CTM_2)
- ISR execution count (CTM_3)
- ISR entry delay cycle count (maximum value of CTM_4)

Note that the large difference between Interrupt occurrence count (CTM_2) and ISR execution count (CTM_3) is because the ISR takes more number of cycles than the actual interrupt period. ISR entry delay cycle count will also be higher due to the same reason.

14.8.1.2 ERAD Profile Function

FILE: erad_ex1_profile_function.c

This example uses BUSCOMP1, BUSCOMP2 and COUNTER1 of the ERAD module to profile a function (delayFunction). It calculates the CPU cycles taken between the the start address of the function to the end address of the function

Two dummy variable are written to inside the function - startCount and endCount. BUSCOMP3, BUSCOMP4 and COUNTER2 are used to profile the time taken between the access to startCount variable till the access to endCount variable.

Both the counters are setup to operate in START-STOP mode and count the number of CPU cycles spend between the respective bus comparator events.

Watch Variables

- cycles_Functio - the maximum number of cycles between the start of function to the end of function
- cycles_Data - the maximum number of cycles taken between accessing startCount variable to endCount variable

External Connections
None

14.8.1.3 ERAD Profile Function

FILE: erad_ex1_profile_function_syscfg.c

This example uses BUSCOMP1, BUSCOMP2 and COUNTER1 of the ERAD module to profile a function (delayFunction). It calculates the CPU cycles taken between the the start address of the function to the end address of the function

Two dummy variable are written to inside the function - startCount and endCount. BUSCOMP3, BUSCOMP4 and COUNTER2 are used to profile the time taken between the access to startCount variable till the access to endCount variable.

Both the counters are setup to operate in START-STOP mode and count the number of CPU cycles spend between the respective bus comparator events.

Watch Variables

- cycles_Function - the maximum number of cycles between the start of function to the end of function
- cycles_Data - the maximum number of cycles taken between accessing startCount variable to endCount variable

External Connections
None

14.8.1.4 ERAD HWBP Monitor Program Counter

FILE: erad_ex2_bus_monitor.c

In this example, the function delayFunction is called multiple times. The function does read and writes to the global variables startCount and endCount.

The BUSCOMP1 and COUNTER1 is used to count the number of times the function delayFunction was invoked. BUSCOMP2 is used to generate an interrupt when there is read access to the startCount variable and BUSCOMP3 is used to generate an interrupt when there is a write access to the endCount variable

Watch Variables

- funcCount - number of times the function delayFunction was invoked
- isrCount - number of times the ISR was invoked

External Connections

- None

14.8.1.5 ERAD HWBP Monitor Program Counter

FILE: erad_ex2_bus_monitor_syscfg.c

In this example, the function delayFunction is called multiple times. The function does read and writes to the global variables startCount and endCount.

The BUSCOMP1 and COUNTER1 is used to count the number of times the function delayFunction was invoked. BUSCOMP2 is used to generate an interrupt when there is read access to the startCount variable and BUSCOMP3 is used to generate an interrupt when there is a write access to the endCount variable

Watch Variables

- funcCount - number of times the function delayFunction was invoked
- isrCount - number of times the ISR was invoked

External Connections

- None

14.8.1.6 ERAD Profile Function

FILE: erad_ex2_profilefunction.c

This example contains a basic FIR calculation and sorting algorithm to help demonstrate the function profiling capability of the ERAD peripheral. A number of FIR sums are calculated within a loop and are then sorted using the insertion sort algorithm. Cycle counts of both the FIR calculations and the sorting algorithm are output to the screen through the scripting console. In this example, it can be seen that sorting the data takes up a majority of the CPU cycles executed in this program.

To properly use the provided ERAD script, the following variables must be set in the scripting environment prior to launching the ERAD script:

- var PROJ_NAME = "erad_debugger_ex2_profilefunction"
- var PROJ_WKSPC_LOC = "<proj_workspace_path>"
- var PROJ_CONFIG = "<name of active configuration [CPU1_FLASH|CPU1_RAM]>"

To run the ERAD script, use the following command in the scripting console:

- loadJSFile("<proj_workspace_path>\\erad_debugger_ex2_profilefunction\\erad_ex2_profile_function.js", 0);

Note that the script must be run after loading and running the .out on the C28x core.

The included JavaScript file, erad_ex2_profile_function.js, uses Debug Server Scripting (DSS) features. For information on using the DSS, please visit: *http://software-dl.ti.com/ccs/esd/documents/users_guide/sdto_dss_handbook.html*

This example uses 4 HW breakpoints and 2 counters:

- HWBP_1 : PC = start address of performFIR
- HWBP_2 : PC = end address of performFIR
- HWBP_3 : PC = start address of sortMax
- HWBP_4 : PC = end address of sortMax
- CTM_1 : Used to count the performFIR execution cycles. Configured in start-stop mode with start event as HWBP_1 and stop event as HWBP_2
- CTM_2 : Used to count the sortMax execution cycles. Configured in start-stop mode with start event as HWBP_3 and stop event as HWBP_4

External Connections

- None.

Watch Variables

- FIR_iterationCounter - A counter for the number of times FIR calculation and sorting was performed

Profiling Script Output

- Current FIR cycle count (CTM_1)
- Max FIR cycle count (maximum value of CTM_1)
- Current sorting function cycle count (CTM_2)
- Max sorting function cycle count (maximum value of CTM_2)

Note that the the counters are reset after the stop event. The counter value remains 0 till the next start event occurs. The javascript continuously reads the counter value in a while(1) and hence the current counter may return 0.

14.8.1.7 ERAD Stack Overflow

FILE: erad_ex3_stackoverflow.c

This example shows the basic setup of CAN in order to transmit and receive messages on the CAN bus. The CAN peripheral is configured to transmit messages with a specific CAN ID. A message is then transmitted once per second, using a simple delay loop for timing. The message that is sent is a 2 byte message that contains an incrementing pattern.

This example sets up the CAN controller in External Loopback test mode. Data transmitted is visible on the CANTXA pin and is received internally back to the CAN Core.

A buffer is created to store message history up to 50 messages for the duration of the program. A logic error is intentionally made to allow the buffer to overflow, eventually causing a stack overflow. The included JavaScript file, stack_overflow.js, programs ERAD registers in order to detect the stack overflow and halt the CPU once the illegal write is made. The illegal write is made after 507 messages are received.

To properly use the provided ERAD script, the following variables must be set in the scripting environment prior to launching the ERAD script:

- var PROJ_NAME = "erad_debugger_ex3_stackoverflow"
- var PROJ_WKSPC_LOC = <proj_workspace_path>

To run the ERAD script, use the following command in the scripting console:

- loadJSFile("<proj_workspace_path>\\erad_debugger_ex3_stackoverflow\\erad_ex3_stack_overflow.js", 0);

Note that the script must be run after loading and running the .out on the C28x core.

The included JavaScript file, erad_ex3_stack_overflow.js, uses Debug Server Scripting (DSS) features. For information on using the DSS, please visit: *http://software-dl.ti.com/ccs/esd/documents/users_guide/sdto_dss_handbook.html*

This example uses 1 HW watchpoint :

- HWBP_1 : Data Write Address Bus = Stack end address + 1

External Connections

- None.

Watch Variables

- msgCount - A counter for the number of successful messages received
- txMsgData - An array with the data being sent
- rxMsgData - An array with the data that was received
- msgHistoryBuff - An array meant to store the last 50 messages received

Profiling Script Output

- "STACK OVERFLOW detected. Halting CPU." will be printed in the scripting console when a stack overflow occurs (that is, when the watchpoint is hit)

14.8.1.8 ERAD HWBP Stack Overflow Detection

FILE: erad_ex3_stack_overflow_detect.c

This example uses BUSCOMP1 to monitor the stack. The Bus comparator is set to monitor the data write access bus and generate an RTOS interrupt CPU when a write is detected to end of the STACK within a threshold.

Watch Variables
- functionCallCount - the number of times the recursive function overflowing the STACK is called.
- x indicates that the ISR has been entered

External Connections
None

14.8.1.9 ERAD HWBP Stack Overflow Detection

FILE: erad_ex3_stack_overflow_detect_syscfg.c

This example uses BUSCOMP1 to monitor the stack. The Bus comparator is set to monitor the data write access bus and generate an RTOS interrupt CPU when a write is detected to end of the STACK within a threshold.

Watch Variables
- functionCallCount - the number of times the recursive function overflowing the STACK is called.
- x indicates that the ISR has been entered

External Connections
None

14.8.1.10 ERAD Profile Interrupts CLA

FILE: erad_ex4_profileinterrupts_cla.c

This example configures EPWM1A to run at 1 KHz (period = 1 ms) to trigger a start-of-conversion on ADC channel A0. This channel will, in turn, sample EPWM4A which is set to run at 100Hz. At the end-of-conversion the ADC interrupt is fired. The interrupt signal will be used to trigger a CLA task that runs an FIR filter. The filter is designed to be low pass with a cutoff frequency of 100Hz; it will remove the odd harmonics in the input signal smoothing the square wave to a sinusoidal shape. The CLA background task will continuously buffer the filtered output in a circular buffer.

This example also utilizes the ERAD peripheral to profile the Interrupt Service Routine (ISR) cla1ISR1 (on the C28x core). The ISR contains a loop that simulates storing a random amount of data to a location in order to introduce variability into the cycle measurements. The ERAD peripheral is also configured to count the number of times the system event CLA_INTERRUPT1 occurs.

To properly use the provided ERAD script, the following variables must be set in the scripting environment prior to launching the ERAD script:

- var PROJ_NAME = "erad_debugger_ex4_profileinterrupts_cla"
- var PROJ_WKSPC_LOC = "<proj_workspace_path>"
- var PROJ_CONFIG = "<name of active configuration [CPU1_FLASH|CPU1_RAM]>"

To run the ERAD script, use the following command in the scripting console:

- loadJSFile("<proj_workspace_path>\\erad_debugger_ex4_profileinterrupts_cla\ \erad_ex4_profile_interrupts_cla.js", 0);

Note that the script must be run after loading and running the .out on the C28x core.

The included JavaScript file, erad_ex4_profile_interrupts_cla.js, uses Debug Server Scripting (DSS) features. For information on using the DSS, please visit: *http://software-dl.ti.com/ccs/esd/documents/users_guide/sdto_dss_handbook.html*

This example uses 4 HW breakpoints and 2 counters:
- HWBP_1 : PC = start address of cla1Isr1
- HWBP_2 : PC = end address of cla1Isr1
- CTM_1 : Used to count the cla1Isr1 execution cycles. Configured in start-stop mode with start event as HWBP_1 and stop event as HWBP_2

- CTM_2 : Used to count the number of times the system event CLA_INTERRUPT1 event has occurred. Configured in rising-edge count mode with counting input as system event CLA_INTERRUPT1 (INP_SEL[26])

External Connections

- connect A0 to EPWM4A

Watch Variables

- ISR_count - A counter that signifies how many times cla1ISR1 executes

Profiling Script Output

- Current ISR cycle count (CTM_1)
- Max ISR cycle count (maximum value of CTM_1)
- Interrupt occurrence count (CTM_2)

14.8.1.11 ERAD Profiling Interrupts

FILE: erad_ex4_profile_interrupt.c

This example shows how an ISR can be profiled by ERAD. The CPU timer generates interrupts periodically. We set up the counters to count the CPU cycles elapsed while executing the ISR, to count the number of interrupts, the number of ISR executions and the CPU cycles elapsed between the interrupt and the execution of the ISR.

This example uses 2 bus comparators and 4 counters:

- BUSCOMP_1 : PC = start address of cpuTimer1ISR
- BUSCOMP_2 : PC = address of cpuTimer1IntCount variable access. This specifies the end address of the code of interest.
- COUNTER_1 : Used to count the cpuTimer1ISR execution cycles. Configured in start-stop mode with start event as BUSCOMP_1 and stop event as BUSCOMP_2
- COUNTER_2 : Used to count the number of times the system event TIMER1_TINT1 has occurred. Configured in rising-edge count mode with counting input as system event TIMER1_TINT1
- COUNTER_3 : Used to count the number of times cputTimer2ISR executes. Configured in rising-edge count mode with counting input as BUSCOMP_1
- COUNTER_4 : Used to count the latency from the system event TIMER1_TINT1 to cpuTimer1ISR entry. Configured in start-stop mode with start event as TIMER1_TINT1 and stop event as BUSCOMP_1

We configure the COUNTER1 to generate an interrupt once it reaches a threshold value.

External Connections

- None

Profiling Output

- Current ISR cycle count (COUNTER_1)
- Interrupt occurrence count (COUNTER_2)
- ISR execution count (COUNTER_3)
- ISR entry delay cycle count (maximum value of COUNTER_4)
- x - To show that the ISR executed

14.8.1.12 ERAD Profiling Interrupts

FILE: erad_ex4_profile_interrupt_syscfg.c

This example shows how an ISR can be profiled by ERAD. The CPU timer generates interrupts periodically. We set up the counters to count the CPU cycles elapsed while executing the ISR, to count the number of interrupts, the number of ISR executions and the CPU cycles elapsed between the interrupt and the execution of the ISR.

This example uses 2 bus comparators and 4 counters:

- BUSCOMP_1 : PC = start address of cpuTimer1ISR
- BUSCOMP_2 : PC = address of cpuTimer1IntCount variable access. This specifies the end address of the code of interest.

- COUNTER_1 : Used to count the cpuTimer1ISR execution cycles. Configured in start-stop mode with start event as BUSCOMP_1 and stop event as BUSCOMP_2
- COUNTER_2 : Used to count the number of times the system event TIMER1_TINT1 has occurred. Configured in rising-edge count mode with counting input as system event TIMER1_TINT1
- COUNTER_3 : Used to count the number of times cputTimer2ISR executes. Configured in rising-edge count mode with counting input as BUSCOMP_1
- COUNTER_4 : Used to count the latency from the system event TIMER1_TINT1 to cpuTimer1ISR entry. Configured in start-stop mode with start event as TIMER1_TINT1 and stop event as BUSCOMP_1

We configure the COUNTER1 to generate an interrupt once it reaches a threshold value.

External Connections

- None

Profiling Output

- Current ISR cycle count (COUNTER_1)
- Interrupt occurrence count (COUNTER_2)
- ISR execution count (COUNTER_3)
- ISR entry delay cycle count (maximum value of COUNTER_4)
- x - To show that the ISR executed

14.8.1.13 ERAD MEMORY ACCESS RESTRICT

FILE: erad_ex5_restricted_write_detect.c

This example uses BUSCOMP1 to monitor the Data Write Address Bus. It monitors the bus and generates an RTOS interrupt if a certain region of memory is accessed by the PC. The user may disable the Bus Comparator to access that region.

Use the COM port (Baud=9600) to try to write to the restricted area.

Watch Variables

- x : stores the number of times the region of memory is accessed

External Connections

- None

14.8.1.14 ERAD INTERRUPT ORDER

FILE: erad_ex6_interrupt_order.c

This example uses a COUNTER to monitor the sequence of ISRs executed. An interrupt is generated if the ISRs executed are not in the expected order. The expected order is CPUTimer0 ,then CPUTimer1 and then CPUTimer2

The counter is configured in Start-Stop Mode to count the number of times CPUTimer interrupt occurs between the CPUTimer1 interrupt and CPUTimer2 ISRs. Ideally, this count should be zero if the interrupts are occurring in the expected order. we configure a threshold value of 1 to genarete an RTOS interrupt. This indicates that the CPUTimer2 interrupt has come out of order.

For demonstaration puproses, this example disables CPUTimer1 to simulate this error.

Watch Variables

- cpuTimer0IntCount: Number of executions of ISR0
- cpuTimer1IntCount: Number of executions of ISR1
- cpuTimer2IntCount: Number of executions of ISR2

External Connections

- None

14.8.1.15 ERAD AND CLB

FILE: erad_ex7_reg_write_clb.c

This example uses 4 BUS COMPARATORS of ERAD along with the CLB. One bus comparator monitors a write to x, another one monitors a write to y. The other two monitor a write of 0x1 and 0x0. By using the LUTs in the CLB1 tile, we can monitor a write of 0x1 to x or 0x0 to x. These are used to change the state of FSM2 in the CLB1 tile. If y is accessed before writing a 0x1 to x, an interrupt is generated and y is changed to 0x0 again. The LED2 indicates when access to y is allowed(it is off at this point) The LED1 indicates if an invalid access is attempted. A COUNTER in ERAD is used to count the number of access attempts to y.

Watch Variables

- y
- x
- a - counts the number of access attempts to y

External Connections

None

14.8.1.16 ERAD PWM PROTECTION

FILE: erad_ex8_pwm_protection.c

This example uses a BUS COMPARATOR and the CLB to detect the event when the delay between the interrupt and the ISR execution is longer than expected. The PWM output is also tripped in this case.

Watch Variables

- adcAResults stores the results of the conversions from the ADC

External Connections

- Monitor the PWM output (GPIO0)

14.9 ERAD Registers

This section describes the Embedded Real-Time Analysis and Diagnostic Registers.

14.9.1 ERAD Base Address Table (C28)

Table 14-4. ERAD Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
EradGlobalRegs	ERAD_GLOBAL_REGS	ERAD_GLOBAL_BASE	0x0005_E800	YES	YES	-	-	YES
EradHWBP1Regs	ERAD_HWBP_REGS	ERAD_HWBP1_BASE	0x0005_E900	YES	YES	-	-	YES
EradHWBP2Regs	ERAD_HWBP_REGS	ERAD_HWBP2_BASE	0x0005_E908	YES	YES	-	-	YES
EradHWBP3Regs	ERAD_HWBP_REGS	ERAD_HWBP3_BASE	0x0005_E910	YES	YES	-	-	YES
EradHWBP4Regs	ERAD_HWBP_REGS	ERAD_HWBP4_BASE	0x0005_E918	YES	YES	-	-	YES
EradHWBP5Regs	ERAD_HWBP_REGS	ERAD_HWBP5_BASE	0x0005_E920	YES	YES	-	-	YES
EradHWBP6Regs	ERAD_HWBP_REGS	ERAD_HWBP6_BASE	0x0005_E928	YES	YES	-	-	YES
EradHWBP7Regs	ERAD_HWBP_REGS	ERAD_HWBP7_BASE	0x0005_E930	YES	YES	-	-	YES
EradHWBP8Regs	ERAD_HWBP_REGS	ERAD_HWBP8_BASE	0x0005_E938	YES	YES	-	-	YES
EradCounter1Regs	ERAD_COUNTER_REGS	ERAD_COUNTER1_BASE	0x0005_E980	YES	YES	-	-	YES
EradCounter2Regs	ERAD_COUNTER_REGS	ERAD_COUNTER2_BASE	0x0005_E990	YES	YES	-	-	YES
EradCounter3Regs	ERAD_COUNTER_REGS	ERAD_COUNTER3_BASE	0x0005_E9A0	YES	YES	-	-	YES
EradCounter4Regs	ERAD_COUNTER_REGS	ERAD_COUNTER4_BASE	0x0005_E9B0	YES	YES	-	-	YES
EradCRCGlobalRegs	ERAD_CRC_GLOBAL_REGS	ERAD_CRC_GLOBAL_BASE	0x0005_EA00	YES	YES	-	-	YES
EradCRC1Regs	ERAD_CRC_REGS	ERAD_CRC1_BASE	0x0005_EA10	YES	YES	-	-	YES
EradCRC2Regs	ERAD_CRC_REGS	ERAD_CRC2_BASE	0x0005_EA20	YES	YES	-	-	YES
EradCRC3Regs	ERAD_CRC_REGS	ERAD_CRC3_BASE	0x0005_EA30	YES	YES	-	-	YES
EradCRC4Regs	ERAD_CRC_REGS	ERAD_CRC4_BASE	0x0005_EA40	YES	YES	-	-	YES

Table 14-4. ERAD Base Address Table (C28) (continued)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
EradCRC5Regs	ERAD_CRC_REGS	ERAD_CRC5_BASE	0x0005_EA50	YES	YES	-	-	YES
EradCRC6Regs	ERAD_CRC_REGS	ERAD_CRC6_BASE	0x0005_EA60	YES	YES	-	-	YES
EradCRC7Regs	ERAD_CRC_REGS	ERAD_CRC7_BASE	0x0005_EA70	YES	YES	-	-	YES
EradCRC8Regs	ERAD_CRC_REGS	ERAD_CRC8_BASE	0x0005_EA80	YES	YES	-	-	YES

14.9.2 ERAD_GLOBAL_REGS Registers

Table 14-5 lists the memory-mapped registers for the ERAD_GLOBAL_REGS registers. All register offset addresses not listed in Table 14-5 should be considered as reserved locations and the register contents should not be modified.

Table 14-5. ERAD_GLOBAL_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	GLBL_EVENT_STAT	Global Event Status Register		Go
2h	GLBL_HALT_STAT	Global Halt Status Register		Go
4h	GLBL_ENABLE	Global Enable Register	EALLOW	Go
6h	GLBL_CTM_RESET	Global Counter Reset	EALLOW	Go
8h	GLBL_NMI_CTL	Global Debug NMI control	EALLOW	Go
Ah	GLBL_OWNER	Global Ownership	EALLOW	Go
Ch	GLBL_EVENT_AND_MASK	Global Bus Comparator Event AND Mask Register	EALLOW	Go
Eh	GLBL_EVENT_OR_MASK	Global Bus Comparator Event OR Mask Register	EALLOW	Go
10h	GLBL_AND_EVENT_INT_MASK	Global AND Event Interrupt Mask Register	EALLOW	Go
12h	GLBL_OR_EVENT_INT_MASK	Global OR Event Interrupt Mask Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 14-6 shows the codes that are used for access types in this section.

Table 14-6. ERAD_GLOBAL_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

14.9.2.1 GLBL_EVENT_STAT Register (Offset = 0h) [Reset = 0h]

GLBL_EVENT_STAT is shown in Figure 14-8 and described in Table 14-7.

Return to the Summary Table.

This register contains one bit for each of the bus comparator modules and the counter modules that are present in a device. Each bit directly reflects the state of the EVENT_FIRED bit of the corresponding module. This facilitates software to just read one register and find out if any of the debug modules had fired.

Figure 14-8. GLBL_EVENT_STAT Register

15	14	13	12	11	10	9	8
RESERVED				CTM4	CTM3	CTM2	CTM1
R-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
HWBP8	HWBP7	HWBP6	HWBP5	HWBP4	HWBP3	HWBP2	HWBP1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 14-7. GLBL_EVENT_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	CTM4	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Counter unit 4. 0 No Event 1 Event Fired Reset type: ERAD_RESET
10	CTM3	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Counter unit 3. 0 No Event 1 Event Fired Reset type: ERAD_RESET
9	CTM2	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Counter unit 2. 0 No Event 1 Event Fired Reset type: ERAD_RESET
8	CTM1	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Counter unit 1. 0 No Event 1 Event Fired Reset type: ERAD_RESET
7	HWBP8	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Enhanced Bus Comparator (EBC) unit 8. 0 No Event 1 Event Fired Reset type: ERAD_RESET
6	HWBP7	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Enhanced Bus Comparator (EBC) unit 7. 0 No Event 1 Event Fired Reset type: ERAD_RESET
5	HWBP6	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Enhanced Bus Comparator (EBC) unit 6. 0 No Event 1 Event Fired Reset type: ERAD_RESET

Table 14-7. GLBL_EVENT_STAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	HWBP5	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Enhanced Bus Comparator (EBC) unit 5. 0 No Event 1 Event Fired Reset type: ERAD_RESET
3	HWBP4	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Enhanced Bus Comparator (EBC) unit 4. 0 No Event 1 Event Fired Reset type: ERAD_RESET
2	HWBP3	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Enhanced Bus Comparator (EBC) unit 3. 0 No Event 1 Event Fired Reset type: ERAD_RESET
1	HWBP2	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Enhanced Bus Comparator (EBC) unit 2. 0 No Event 1 Event Fired Reset type: ERAD_RESET
0	HWBP1	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Enhanced Bus Comparator (EBC) unit 1. 0 No Event 1 Event Fired Reset type: ERAD_RESET

14.9.2.2 GLBL_HALT_STAT Register (Offset = 2h) [Reset = 0h]

GLBL_HALT_STAT is shown in Figure 14-9 and described in Table 14-8.

Return to the Summary Table.

This register contains one bit for each of the bus comparator modules and the counter modules that are present in a device. Each bit directly reflects the state of the EVENT_FIRED status bit. This facilitates software to just read one register and find out if any of the debug modules have fired.

Figure 14-9. GLBL_HALT_STAT Register

15	14	13	12	11	10	9	8
RESERVED				CTM4	CTM3	CTM2	CTM1
R-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
HWBP8	HWBP7	HWBP6	HWBP5	HWBP4	HWBP3	HWBP2	HWBP1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 14-8. GLBL_HALT_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	CTM4	R	0h	This bit directly reflects the state of the completed bit of the Counter unit 4. 0 No Event 1 Event Fired Reset type: ERAD_RESET
10	CTM3	R	0h	This bit directly reflects the state of the completed bit of the Counter unit 3. 0 No Event 1 Event Fired Reset type: ERAD_RESET
9	CTM2	R	0h	This bit directly reflects the state of the completed bit of the Counter unit 2. 0 No Event 1 Event Fired Reset type: ERAD_RESET
8	CTM1	R	0h	This bit directly reflects the state of the completed bit of the Counter unit 1. 0 No Event 1 Event Fired Reset type: ERAD_RESET
7	HWBP8	R	0h	This bit directly reflects the state of the completed bit of the Enhanced Bus Comparator (EBC) unit 8. 0 Not Completed 1 Completed Reset type: ERAD_RESET
6	HWBP7	R	0h	This bit directly reflects the state of the completed bit of the Enhanced Bus Comparator (EBC) unit 7. 0 Not Completed 1 Completed Reset type: ERAD_RESET
5	HWBP6	R	0h	This bit directly reflects the state of the completed bit of the Enhanced Bus Comparator (EBC) unit 6. 0 Not Completed 1 Completed Reset type: ERAD_RESET

Table 14-8. GLBL_HALT_STAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	HWBP5	R	0h	This bit directly reflects the state of the completed bit of the Enhanced Bus Comparator (EBC) unit 5. 0 Not Completed 1 Completed Reset type: ERAD_RESET
3	HWBP4	R	0h	This bit directly reflects the state of the completed bit of the Enhanced Bus Comparator (EBC) unit 4. 0 Not Completed 1 Completed Reset type: ERAD_RESET
2	HWBP3	R	0h	This bit directly reflects the state of the completed bit of the Enhanced Bus Comparator (EBC) unit 3. 0 Not Completed 1 Completed Reset type: ERAD_RESET
1	HWBP2	R	0h	This bit directly reflects the state of the completed bit of the Enhanced Bus Comparator (EBC) unit 2. 0 Not Completed 1 Completed Reset type: ERAD_RESET
0	HWBP1	R	0h	This bit directly reflects the state of the completed bit of the Enhanced Bus Comparator (EBC) unit 1. 0 Not Completed 1 Completed Reset type: ERAD_RESET

14.9.2.3 GLBL_ENABLE Register (Offset = 4h) [Reset = 0h]

GLBL_ENABLE is shown in Figure 14-10 and described in Table 14-9.

Return to the Summary Table.

This register contains one bit for each of the bus comparator modules and the counter modules that are present in a device. Each bit directly acts as a global enable for the corresponding module. This bit has to be set to 1 for the module to be functional.

Figure 14-10. GLBL_ENABLE Register

15	14	13	12	11	10	9	8
RESERVED				CTM4	CTM3	CTM2	CTM1
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
HWBP8	HWBP7	HWBP6	HWBP5	HWBP4	HWBP3	HWBP2	HWBP1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 14-9. GLBL_ENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	CTM4	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Counter unit 4. 0 Disabled 1 Enabled Reset type: ERAD_RESET
10	CTM3	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Counter unit 3. 0 Disabled 1 Enabled Reset type: ERAD_RESET
9	CTM2	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Counter unit 2. 0 Disabled 1 Enabled Reset type: ERAD_RESET
8	CTM1	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Counter unit 1. 0 Disabled 1 Enabled Reset type: ERAD_RESET
7	HWBP8	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Enhanced Bus Comparator (EBC) unit 8. 0 Disabled 1 Enabled Reset type: ERAD_RESET
6	HWBP7	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Enhanced Bus Comparator (EBC) unit 7. 0 Disabled 1 Enabled Reset type: ERAD_RESET
5	HWBP6	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Enhanced Bus Comparator (EBC) unit 6. 0 Disabled 1 Enabled Reset type: ERAD_RESET

Table 14-9. GLBL_ENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	HWBP5	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Enhanced Bus Comparator (EBC) unit 5. 0 Disabled 1 Enabled Reset type: ERAD_RESET
3	HWBP4	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Enhanced Bus Comparator (EBC) unit 4. 0 Disabled 1 Enabled Reset type: ERAD_RESET
2	HWBP3	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Enhanced Bus Comparator (EBC) unit 3. 0 Disabled 1 Enabled Reset type: ERAD_RESET
1	HWBP2	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Enhanced Bus Comparator (EBC) unit 2. 0 Disabled 1 Enabled Reset type: ERAD_RESET
0	HWBP1	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Enhanced Bus Comparator (EBC) unit 1. 0 Disabled 1 Enabled Reset type: ERAD_RESET

14.9.2.4 GLBL_CTM_RESET Register (Offset = 6h) [Reset = 0h]

GLBL_CTM_RESET is shown in Figure 14-11 and described in Table 14-10.

Return to the Summary Table.

This register contains one bit for each of the counter modules that are present in a device. Each bit directly acts as a reset for the counters for the corresponding module.(It does not affect anything else except resetting the counter.
Example: If the counter was previously incrementing before reset, then on a reset event the counter gets reset and continues to increment again).

Figure 14-11. GLBL_CTM_RESET Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CTM4	CTM3	CTM2	CTM1
R-0h				R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h

Table 14-10. GLBL_CTM_RESET Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	CTM4	R-0/W	0h	This bit directly resets the state the Counter unit 4. 0 No Effect 1 Reset Reset type: ERAD_RESET
2	CTM3	R-0/W	0h	This bit directly resets the state the Counter unit 3. 0 No Effect 1 Reset Reset type: ERAD_RESET
1	CTM2	R-0/W	0h	This bit directly resets the state the Counter unit 2. 0 No Effect 1 Reset Reset type: ERAD_RESET
0	CTM1	R-0/W	0h	This bit directly resets the state the Counter unit 1. 0 No Effect 1 Reset Reset type: ERAD_RESET

14.9.2.5 GLBL_NMI_CTL Register (Offset = 8h) [Reset = 0h]

GLBL_NMI_CTL is shown in Figure 14-12 and described in Table 14-11.

Return to the Summary Table.

This register contains one bit for each of the bus comparator modules anf the counter modules that can cause an NMI to the CPU. When the corresponding bit of a unit is set to 1, then if an event occurs from that module, then an NMI will be generated.

Figure 14-12. GLBL_NMI_CTL Register

15	14	13	12	11	10	9	8
RESERVED				CTM4	CTM3	CTM2	CTM1
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
HWBP8	HWBP7	HWBP6	HWBP5	HWBP4	HWBP3	HWBP2	HWBP1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 14-11. GLBL_NMI_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	CTM4	R/W	0h	This bit enables the genration of an NMI to the CPU by Counter unit 4 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
10	CTM3	R/W	0h	This bit enables the genration of an NMI to the CPU by Counter unit 3. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
9	CTM2	R/W	0h	This bit enables the genration of an NMI to the CPU by Counter unit 2. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
8	CTM1	R/W	0h	This bit enables the genration of an NMI to the CPU by Counter unit 1. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
7	HWBP8	R/W	0h	This bit enables the genration of an NMI to the CPU by Enhanced Bus Comparator (EBC) unit 8. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
6	HWBP7	R/W	0h	This bit enables the genration of an NMI to the CPU by Enhanced Bus Comparator (EBC) unit 7. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
5	HWBP6	R/W	0h	This bit enables the genration of an NMI to the CPU by Enhanced Bus Comparator (EBC) unit 6. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET

Table 14-11. GLBL_NMI_CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	HWBP5	R/W	0h	This bit enables the genration of an NMI to the CPU by Enhanced Bus Comparator (EBC) unit 5. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
3	HWBP4	R/W	0h	This bit enables the genration of an NMI to the CPU by Enhanced Bus Comparator (EBC) unit 4. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
2	HWBP3	R/W	0h	This bit enables the genration of an NMI to the CPU by Enhanced Bus Comparator (EBC) unit 3. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
1	HWBP2	R/W	0h	This bit enables the genration of an NMI to the CPU by Enhanced Bus Comparator (EBC) unit 2. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
0	HWBP1	R/W	0h	This bit enables the genration of an NMI to the CPU by Enhanced Bus Comparator (EBC) unit 1. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET

14.9.2.6 GLBL_OWNER Register (Offset = Ah) [Reset = 0h]

GLBL_OWNER is shown in Figure 14-13 and described in Table 14-12.

Return to the Summary Table.

Global Ownership

Figure 14-13. GLBL_OWNER Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						OWNER	
R-0h						R/W-0h	

Table 14-12. GLBL_OWNER Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R	0h	Reserved
1-0	OWNER	R/W	0h	This register determines whether Application Code or Debugger owns this module or it's kept in No Owner state where debugger or application can access the module. 00 No Owner 01 Application owned 10 Debugger owned 11 Reserved Reset type: ERAD_RESET

14.9.2.7 GLBL_EVENT_AND_MASK Register (Offset = Ch) [Reset = FFFFFFFFh]

GLBL_EVENT_AND_MASK is shown in Figure 14-14 and described in Table 14-13.

Return to the Summary Table.

Global Bus Comparator Event AND Mask Register

Figure 14-14. GLBL_EVENT_AND_MASK Register

31	30	29	28	27	26	25	24
MASK4_HWBP8	MASK4_HWBP7	MASK4_HWBP6	MASK4_HWBP5	MASK4_HWBP4	MASK4_HWBP3	MASK4_HWBP2	MASK4_HWBP1
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
MASK3_HWBP8	MASK3_HWBP7	MASK3_HWBP6	MASK3_HWBP5	MASK3_HWBP4	MASK3_HWBP3	MASK3_HWBP2	MASK3_HWBP1
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
MASK2_HWBP8	MASK2_HWBP7	MASK2_HWBP6	MASK2_HWBP5	MASK2_HWBP4	MASK2_HWBP3	MASK2_HWBP2	MASK2_HWBP1
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
MASK1_HWBP8	MASK1_HWBP7	MASK1_HWBP6	MASK1_HWBP5	MASK1_HWBP4	MASK1_HWBP3	MASK1_HWBP2	MASK1_HWBP1
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 14-13. GLBL_EVENT_AND_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MASK4_HWBP8	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 8: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND4 output Reset type: ERAD_RESET
30	MASK4_HWBP7	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 7: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND4 output Reset type: ERAD_RESET
29	MASK4_HWBP6	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 6: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND4 output Reset type: ERAD_RESET
28	MASK4_HWBP5	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 5: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND4 output Reset type: ERAD_RESET
27	MASK4_HWBP4	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 4: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND4 output Reset type: ERAD_RESET

Table 14-13. GLBL_EVENT_AND_MASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MASK4_HWBP3	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 3: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND4 output Reset type: ERAD_RESET
25	MASK4_HWBP2	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 2: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND4 output Reset type: ERAD_RESET
24	MASK4_HWBP1	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 1: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND4 output Reset type: ERAD_RESET
23	MASK3_HWBP8	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 8: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND3 output Reset type: ERAD_RESET
22	MASK3_HWBP7	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 7: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND3 output Reset type: ERAD_RESET
21	MASK3_HWBP6	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 6: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND3 output Reset type: ERAD_RESET
20	MASK3_HWBP5	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 5: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND3 output Reset type: ERAD_RESET
19	MASK3_HWBP4	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 4: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND3 output Reset type: ERAD_RESET
18	MASK3_HWBP3	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 3: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND3 output Reset type: ERAD_RESET
17	MASK3_HWBP2	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 2: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND3 output Reset type: ERAD_RESET

Table 14-13. GLBL_EVENT_AND_MASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	MASK3_HWBP1	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 1: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND3 output Reset type: ERAD_RESET
15	MASK2_HWBP8	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 8: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND2 output Reset type: ERAD_RESET
14	MASK2_HWBP7	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 7: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND2 output Reset type: ERAD_RESET
13	MASK2_HWBP6	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 6: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND2 output Reset type: ERAD_RESET
12	MASK2_HWBP5	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 5: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND2 output Reset type: ERAD_RESET
11	MASK2_HWBP4	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 4: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND2 output Reset type: ERAD_RESET
10	MASK2_HWBP3	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 3: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND2 output Reset type: ERAD_RESET
9	MASK2_HWBP2	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 2: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND2 output Reset type: ERAD_RESET
8	MASK2_HWBP1	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 1: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND2 output Reset type: ERAD_RESET
7	MASK1_HWBP8	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 8: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND1 output Reset type: ERAD_RESET

Table 14-13. GLBL_EVENT_AND_MASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	MASK1_HWBP7	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 7: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND1 output Reset type: ERAD_RESET
5	MASK1_HWBP6	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 6: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND1 output Reset type: ERAD_RESET
4	MASK1_HWBP5	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 5: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND1 output Reset type: ERAD_RESET
3	MASK1_HWBP4	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 4: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND1 output Reset type: ERAD_RESET
2	MASK1_HWBP3	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 3: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND1 output Reset type: ERAD_RESET
1	MASK1_HWBP2	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 2: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND1 output Reset type: ERAD_RESET
0	MASK1_HWBP1	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 1: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND1 output Reset type: ERAD_RESET

14.9.2.8 GLBL_EVENT_OR_MASK Register (Offset = Eh) [Reset = FFFFFFFFh]

GLBL_EVENT_OR_MASK is shown in Figure 14-15 and described in Table 14-14.

Return to the Summary Table.

Global Bus Comparator Event OR Mask Register

Figure 14-15. GLBL_EVENT_OR_MASK Register

31	30	29	28	27	26	25	24
MASK4_HWBP8	MASK4_HWBP7	MASK4_HWBP6	MASK4_HWBP5	MASK4_HWBP4	MASK4_HWBP3	MASK4_HWBP2	MASK4_HWBP1
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
MASK3_HWBP8	MASK3_HWBP7	MASK3_HWBP6	MASK3_HWBP5	MASK3_HWBP4	MASK3_HWBP3	MASK3_HWBP2	MASK3_HWBP1
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
MASK2_HWBP8	MASK2_HWBP7	MASK2_HWBP6	MASK2_HWBP5	MASK2_HWBP4	MASK2_HWBP3	MASK2_HWBP2	MASK2_HWBP1
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
MASK1_HWBP8	MASK1_HWBP7	MASK1_HWBP6	MASK1_HWBP5	MASK1_HWBP4	MASK1_HWBP3	MASK1_HWBP2	MASK1_HWBP1
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 14-14. GLBL_EVENT_OR_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MASK4_HWBP8	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 8: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR4 output Reset type: ERAD_RESET
30	MASK4_HWBP7	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 7: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR4 output Reset type: ERAD_RESET
29	MASK4_HWBP6	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 6: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR4 output Reset type: ERAD_RESET
28	MASK4_HWBP5	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 5: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR4 output Reset type: ERAD_RESET
27	MASK4_HWBP4	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 4: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR4 output Reset type: ERAD_RESET

Table 14-14. GLBL_EVENT_OR_MASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MASK4_HWBP3	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 3: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR4 output Reset type: ERAD_RESET
25	MASK4_HWBP2	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 2: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR4 output Reset type: ERAD_RESET
24	MASK4_HWBP1	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 1: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR4 output Reset type: ERAD_RESET
23	MASK3_HWBP8	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 8: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR3 output Reset type: ERAD_RESET
22	MASK3_HWBP7	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 7: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR3 output Reset type: ERAD_RESET
21	MASK3_HWBP6	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 6: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR3 output Reset type: ERAD_RESET
20	MASK3_HWBP5	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 5: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR3 output Reset type: ERAD_RESET
19	MASK3_HWBP4	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 4: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR3 output Reset type: ERAD_RESET
18	MASK3_HWBP3	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 3: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR3 output Reset type: ERAD_RESET
17	MASK3_HWBP2	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 2: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR3 output Reset type: ERAD_RESET

Table 14-14. GLBL_EVENT_OR_MASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	MASK3_HWBP1	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 1: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR3 output Reset type: ERAD_RESET
15	MASK2_HWBP8	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 8: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR2 output Reset type: ERAD_RESET
14	MASK2_HWBP7	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 7: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR2 output Reset type: ERAD_RESET
13	MASK2_HWBP6	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 6: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR2 output Reset type: ERAD_RESET
12	MASK2_HWBP5	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 5: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR2 output Reset type: ERAD_RESET
11	MASK2_HWBP4	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 4: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR2 output Reset type: ERAD_RESET
10	MASK2_HWBP3	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 3: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR2 output Reset type: ERAD_RESET
9	MASK2_HWBP2	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 2: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR2 output Reset type: ERAD_RESET
8	MASK2_HWBP1	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 1: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR2 output Reset type: ERAD_RESET
7	MASK1_HWBP8	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 8: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR1 output Reset type: ERAD_RESET

Table 14-14. GLBL_EVENT_OR_MASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	MASK1_HWBP7	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 7: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR1 output Reset type: ERAD_RESET
5	MASK1_HWBP6	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 6: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR1 output Reset type: ERAD_RESET
4	MASK1_HWBP5	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 5: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR1 output Reset type: ERAD_RESET
3	MASK1_HWBP4	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 4: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR1 output Reset type: ERAD_RESET
2	MASK1_HWBP3	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 3: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR1 output Reset type: ERAD_RESET
1	MASK1_HWBP2	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 2: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR1 output Reset type: ERAD_RESET
0	MASK1_HWBP1	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 1: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR1 output Reset type: ERAD_RESET

14.9.2.9 GLBL_AND_EVENT_INT_MASK Register (Offset = 10h) [Reset = 0h]

GLBL_AND_EVENT_INT_MASK is shown in Figure 14-16 and described in Table 14-15.

Return to the Summary Table.

Global AND Event Interrupt Mask Register

Figure 14-16. GLBL_AND_EVENT_INT_MASK Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RTOSINT_MASK4	RTOSINT_MASK3	RTOSINT_MASK2	RTOSINT_MASK1
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 14-15. GLBL_AND_EVENT_INT_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	RTOSINT_MASK4	R/W	0h	RTOSINT generation mask for global AND events MASK4: 1 Corresponding GLOBAL_EVENT_AND is enabled for RTOSINT generation 0 Corresponding GLOBAL_EVENT_AND is disabled for RTOSINT generation Reset type: ERAD_RESET
2	RTOSINT_MASK3	R/W	0h	RTOSINT generation mask for global AND events MASK3: 1 Corresponding GLOBAL_EVENT_AND is enabled for RTOSINT generation 0 Corresponding GLOBAL_EVENT_AND is disabled for RTOSINT generation Reset type: ERAD_RESET
1	RTOSINT_MASK2	R/W	0h	RTOSINT generation mask for global AND events MASK2: 1 Corresponding GLOBAL_EVENT_AND is enabled for RTOSINT generation 0 Corresponding GLOBAL_EVENT_AND is disabled for RTOSINT generation Reset type: ERAD_RESET
0	RTOSINT_MASK1	R/W	0h	RTOSINT generation mask for global AND events MASK1: 1 Corresponding GLOBAL_EVENT_AND is enabled for RTOSINT generation 0 Corresponding GLOBAL_EVENT_AND is disabled for RTOSINT generation Reset type: ERAD_RESET

14.9.2.10 GLBL_OR_EVENT_INT_MASK Register (Offset = 12h) [Reset = 0h]

GLBL_OR_EVENT_INT_MASK is shown in Figure 14-17 and described in Table 14-16.

Return to the Summary Table.

Global OR Event Interrupt Mask Register

Figure 14-17. GLBL_OR_EVENT_INT_MASK Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RTOSINT_MASK4	RTOSINT_MASK3	RTOSINT_MASK2	RTOSINT_MASK1
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 14-16. GLBL_OR_EVENT_INT_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	RTOSINT_MASK4	R/W	0h	RTOSINT generation mask for global OR events MASK3: 1 Corresponding GLOBAL_EVENT_OR is enabled for RTOSINT generation 0 Corresponding GLOBAL_EVENT_OR is disabled for RTOSINT generation Reset type: ERAD_RESET
2	RTOSINT_MASK3	R/W	0h	RTOSINT generation mask for global OR events MASK2: 1 Corresponding GLOBAL_EVENT_OR is enabled for RTOSINT generation 0 Corresponding GLOBAL_EVENT_OR is disabled for RTOSINT generation Reset type: ERAD_RESET
1	RTOSINT_MASK2	R/W	0h	RTOSINT generation mask for global OR events MASK2: 1 Corresponding GLOBAL_EVENT_OR is enabled for RTOSINT generation 0 Corresponding GLOBAL_EVENT_OR is disabled for RTOSINT generation Reset type: ERAD_RESET
0	RTOSINT_MASK1	R/W	0h	RTOSINT generation mask for global OR events MASK1: 1 Corresponding GLOBAL_EVENT_OR is enabled for RTOSINT generation 0 Corresponding GLOBAL_EVENT_OR is disabled for RTOSINT generation Reset type: ERAD_RESET

14.9.3 ERAD_HWBP_REGS Registers

Table 14-17 lists the memory-mapped registers for the ERAD_HWBP_REGS registers. All register offset addresses not listed in Table 14-17 should be considered as reserved locations and the register contents should not be modified.

Table 14-17. ERAD_HWBP_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	HWBP_MASK	HWBP (EBC) Mask Register	EALLOW	Go
2h	HWBP_REF	HWBP (EBC) Reference Register	EALLOW	Go
4h	HWBP_CLEAR	HWBP (EBC) Clear Register	EALLOW	Go
6h	HWBP_CNTL	HWBP (EBC) Control Register	EALLOW	Go
7h	HWBP_STATUS	HWBP (EBC) Status Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 14-18 shows the codes that are used for access types in this section.

Table 14-18. ERAD_HWBP_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

14.9.3.1 HWBP_MASK Register (Offset = 0h) [Reset = 0h]

HWBP_MASK is shown in Figure 14-18 and described in Table 14-19.

Return to the Summary Table.

HWBP (EBC) Mask Register

Figure 14-18. HWBP_MASK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MASK																															
R/W-0h																															

Table 14-19. HWBP_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MASK	R/W	0h	This register contains address mask for comparison. The contents of this register are used along with the reference register to determine the address match. The equation used to determine a match is as follows. Match is true if, (address | mask) == (ref | mask) This register is writable by CPU only if application owns the unit and if EALLOW is set. Otherwise, the writes are ignored. The register is writable by the debugger only if the debugger owns this unit. Otherwise, the writes are ignored. Reset type: ERAD_RESET

14.9.3.2 HWBP_REF Register (Offset = 2h) [Reset = 0h]

HWBP_REF is shown in Figure 14-19 and described in Table 14-20.

Return to the Summary Table.

HWBP (EBC) Reference Register

Figure 14-19. HWBP_REF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REF																															
R/W-0h																															

Table 14-20. HWBP_REF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REF	R/W	0h	This register contains the reference address for comparison. The contents of this register are used along with the mask register to determine the address match. The equation used to determine a match is as follows. Match is true if, (address | mask) == (ref | mask) This register is writable by CPU only if application owns the unit and if EALLOW is set. Otherwise, the writes are ignored. The register is writable by the debugger only if the debugger owns this unit. Otherwise, the writes are ignored. Reset type: ERAD_RESET

14.9.3.3 HWBP_CLEAR Register (Offset = 4h) [Reset = 0h]

HWBP_CLEAR is shown in Figure 14-20 and described in Table 14-21.

Return to the Summary Table.

HWBP (EBC) Clear Register

Figure 14-20. HWBP_CLEAR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EVENT_CLR
R-0h							R-0/W-0h

Table 14-21. HWBP_CLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	EVENT_CLR	R-0/W	0h	Event Clear register: 0 No action. 1 A write with this bit set to 1 will clear the sticky EVENT_FIRED bit in the HWBP_STATUS register and bring the Breakpoint Module statemachine status back to IDLE. Reads of this bit position will always return a 0. Reset type: ERAD_RESET

14.9.3.4 HWBP_CNTL Register (Offset = 6h) [Reset = 0h]

HWBP_CNTL is shown in Figure 14-21 and described in Table 14-22.

Return to the Summary Table.

HWBP (EBC) Control Register

Figure 14-21. HWBP_CNTL Register

15	14	13	12	11	10	9	8
RESERVED				RESERVED	RESERVED	COMP_MODE	
R-0h				R/W-0h	R-0h	R/W-0h	
7	6	5	4	3	2	1	0
COMP_MODE	RTOSINT	STOP	BUS_SEL				RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h				R-0h

Table 14-22. HWBP_CNTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R	0h	Reserved
9-7	COMP_MODE	R/W	0h	Enhanced Bus Comparator (EBC) compare modes: 000 Regular masked compare HWBP_MSK will be ignored for the following modes: 100 Bus value GT HWBP_REF 101 Bus value GE HWBP_REF 110 Bus value LT HWBP_REF 111 Bus value LE HWBP_REF GT means Greater Than GE means Greater or Equal LT means Less Than LE means Lesser or Equal Reset type: ERAD_RESET
6	RTOSINT	R/W	0h	This bit decides whether the Enhanced Bus Comparator (EBC) unit will generate RTOSINTn interrupt when event matches occur. Note that the event outputs will always be generated regardless of the state of this bit. 0 The Enhanced Bus Comparator (EBC) unit will not cause any action towards the CPU. 1 The Enhanced Bus Comparator (EBC) unit will assert RTOSINTn for matching data accesses and trace tags for matching program fetches. Reset type: ERAD_RESET
5	STOP	R/W	0h	This bit decides whether the Enhanced Bus Comparator (EBC) unit will generate CPU halting signals when event matches occur. Note that the event outputs will always be generated regardless of the state of this bit. 0 The Enhanced Bus Comparator (EBC) unit will not cause any action towards halting the CPU. 1 The Enhanced Bus Comparator (EBC) unit will assert ANASTOP for matching data accesses and break tags for matching program fetches. These can cause the CPU to HALT Reset type: ERAD_RESET

Table 14-22. HWBP_CNTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-1	BUS_SEL	R/W	0h	These bits are used to select which CPU buses will be used for comparison to generate the match events. For each bus selected, the corresponding strobes will automatically be selected to determine valid accesses. 0000 PAB for instruction fetches 0001 VPC 0010 DWAB for data write accesses 0011 DRAB for data read accesses 0100 DWDB for write data match 0101 DRDB for read data match 0110 VPC Instruction aligned match 0111 VPC R1 aligned match 1000 VPC R2 aligned match 1001 VPC W aligned match All other combinations are RESERVED. Reset type: ERAD_RESET
0	RESERVED	R	0h	Reserved

14.9.3.5 HWBP_STATUS Register (Offset = 7h) [Reset = 400h]

HWBP_STATUS is shown in Figure 14-22 and described in Table 14-23.

Return to the Summary Table.

HWBP (EBC) Status Register

Figure 14-22. HWBP_STATUS Register

15	14	13	12	11	10	9	8
STATUS		MODULE_ID					
R-0h		R-4h					
7	6	5	4	3	2	1	0
RESERVED							EVENT_FIRED
R-0h							R-0h

Table 14-23. HWBP_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	STATUS	R	0h	Bus comaparator status: 00 Idle 10 Enabled 11 Completed Reset type: ERAD_RESET
13-8	MODULE_ID	R	4h	These bits are always a constant representing a unique identification for the Enhanced Bus Comparator (EBC) unit. Reset type: ERAD_RESET
7-1	RESERVED	R	0h	Reserved
0	EVENT_FIRED	R	0h	This is a sticky bit which gets set every time the HWBP (EBC) unit generates a match event. This will be used by software to figure out whether this HWBP module fired an event or not. This bit will get cleared by writing a '1' to bit 0 of the HWBP_CLEAR register. Reset type: ERAD_RESET

14.9.4 ERAD_COUNTER_REGS Registers

Table 14-24 lists the memory-mapped registers for the ERAD_COUNTER_REGS registers. All register offset addresses not listed in Table 14-24 should be considered as reserved locations and the register contents should not be modified.

Table 14-24. ERAD_COUNTER_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CTM_CNTL	Counter Control Register	EALLOW	Go
1h	CTM_STATUS	Counter Status Register	EALLOW	Go
2h	CTM_REF	Counter Reference Register	EALLOW	Go
4h	CTM_COUNT	Counter Current Value Register	EALLOW	Go
6h	CTM_MAX_COUNT	Counter Max Count Value Register	EALLOW	Go
8h	CTM_INPUT_SEL	Counter Input Select Register	EALLOW	Go
9h	CTM_CLEAR	Counter Clear Register	EALLOW	Go
Ah	CTM_INPUT_SEL_2	Counter Input Select Extension Register	EALLOW	Go
Bh	CTM_INPUT_COND	Counter Input Conditioning Register		Go

Complex bit access types are encoded to fit into small table cells. Table 14-25 shows the codes that are used for access types in this section.

Table 14-25. ERAD_COUNTER_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

14.9.4.1 CTM_CNTL Register (Offset = 0h) [Reset = 0h]

CTM_CNTL is shown in Figure 14-23 and described in Table 14-26.

Return to the Summary Table.

Counter Control Register

Figure 14-23. CTM_CNTL Register

15	14	13	12	11	10	9	8
RESERVED				CNT_INP_SEL_EN	RST_EN	RESERVED	START_STOP_CUMULATIVE
R-0h				R/W-0h	R/W-0h	R-0h	R/W-0h
7	6	5	4	3	2	1	0
RTOSINT	STOP	RESERVED	RST_ON_MATCH	EVENT_MODE	START_STOP_MODE	RESERVED	
R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	

Table 14-26. CTM_CNTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	CNT_INP_SEL_EN	R/W	0h	0 = Disable using the input_select register for the count input. The counter will always count CPU cycles. 1 = Enable using the input_select register for the count input. The counter will count the event selected by the count input register. Reset type: ERAD_RESET
10	RST_EN	R/W	0h	This bit decides if the reset input is enabled or not. Setting this to 1 will cause the counter to reset to zero whenever the selected reset input goes active high. No event will be generated when the counter is reset. Setting this bit to 0 will cause the counter to ignore the reset inputs. Reset type: ERAD_RESET
9	RESERVED	R	0h	Reserved
8	START_STOP_CUMULATIVE	R/W	0h	This bit decides whether the counter counts to give the cumulative cycle count for 'n' number of successive start stop events or clears the counter on very stop event to record the MAX_COUNT across successive start stop sequences. 0 When in START_STOP mode counter gets cleared on every stop event and MAX_COUNT records the max value 1 When in START_STOP mode counter keeps counting between successive start stop events to generate a cumulative count w/o clearing the counter on any stop events. MAX_COUNT register is invalid when this bit is set. Reset type: ERAD_RESET
7	RTOSINT	R/W	0h	This bit decides whether the counter module will generate RTOSINTn interrupt when count value matches the reference. Note that the event outputs will always be generated regardless of the state of this bit. 0 The counter unit will not cause any action towards the CPU. 1 The counter unit will assert RTOSINTn when the count value matches the reference value. Reset type: ERAD_RESET
6	STOP	R/W	0h	This bit decides whether the counter module will generate a watchpoint to the CPU when the count value matches the reference. Note that the event outputs will always be generated regardless of the state of this bit. 0 The counter unit will not generate a watchpoint. 1 The counter unit will assert ANASTOP when the count value matches the reference. Reset type: ERAD_RESET

Table 14-26. CTM_CNTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	RESERVED	R	0h	Reserved
4	RST_ON_MATCH	R/W	0h	This bit is used to decide whether the counter will reset to zero once it reaches the reference value. 0 Counter will stay at the reference value and the counter will go to COMPLETED state and further counting will be stopped. 1 The counter will reset to zero once it reaches the match value and will stay enabled. Reset type: ERAD_RESET
3	EVENT_MODE	R/W	0h	This bit is used to decide whether the counter will count the level of the event or the edge of the event. 0 Counter will increment the count as long as the count input is active high. 1 The counter will count only on the rising edge of the count input. Reset type: ERAD_RESET
2	START_STOP_MODE	R/W	0h	This bit is used to decide whether the counter will count in the START_STOP mode or not. 0 Normal count mode. The counter will not depend on the START and STOP events 1 This is the START-STOP mode of the counter. The counter will start counting only after the START input has been asserted. It will continue to count the selected event till the STOP event is seen. Reset type: ERAD_RESET
1-0	RESERVED	R	0h	Reserved

14.9.4.2 CTM_STATUS Register (Offset = 1h) [Reset = 10h]

CTM_STATUS is shown in Figure 14-24 and described in Table 14-27.

Return to the Summary Table.

Counter Status Register

Figure 14-24. CTM_STATUS Register

15	14	13	12	11	10	9	8
STATUS				MODULE_ID			
R-0h				R-4h			
7	6	5	4	3	2	1	0
MODULE_ID						OVERFLOW	EVENT_FIRED
R-4h						R-0h	R-0h

Table 14-27. CTM_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	STATUS	R	0h	Counter unit status, 00 Idle 10 Enabled 11 Completed Reset type: ERAD_RESET
11-2	MODULE_ID	R	4h	These bits are always a constant representing a unique identification for the trigger unit. Reset type: ERAD_RESET
1	OVERFLOW	R	0h	This is a sticky bit which gets set every time the counter overflows and wraps around after reaching 0xffffffff.This bit will get cleared by writing a '1' to bit 9 of the CTM_CNTL register. Reset type: ERAD_RESET
0	EVENT_FIRED	R	0h	This is a sticky bit which gets set every time the CTM unit generates a match event. This will be used by software to figure out whether this CTM module fired an event or not. This bit will get cleared by writing a '1' to bit 9 of the CTM_CNTL register. Reset type: ERAD_RESET

14.9.4.3 CTM_REF Register (Offset = 2h) [Reset = 0h]

CTM_REF is shown in Figure 14-25 and described in Table 14-28.

Return to the Summary Table.

Counter Reference Register

Figure 14-25. CTM_REF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REF																															
R/W-0h																															

Table 14-28. CTM_REF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REF	R/W	0h	This register contains the counter reference value for comparison. The counter will generate an event if the count value matches the reference register (considering both upper and lower half of the register). This register is writable by CPU only if application owns the unit and if EALLOW is set. Otherwise, the writes are ignored. The register is writable by the debugger only if the debugger owns this unit. Otherwise, the writes are ignored. Refrence match is enabled only when a non zero value is programmed on one of the REF register. Reset type: ERAD_RESET

14.9.4.4 CTM_COUNT Register (Offset = 4h) [Reset = 0h]

CTM_COUNT is shown in Figure 14-26 and described in Table 14-29.

Return to the Summary Table.

Counter Current Value Register

Figure 14-26. CTM_COUNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT																															
R/W-0h																															

Table 14-29. CTM_COUNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNT	R/W	0h	This register contains the current count value. The counter will generate an event if the count value matches the reference register (considering both upper and lower half of the register). This register is writable by CPU only if application owns the unit and if EALLOW is set. Otherwise, the writes are ignored. The register is writable by the debugger only if the debugger owns this unit. Otherwise, the writes are ignored. Reset type: ERAD_RESET

14.9.4.5 CTM_MAX_COUNT Register (Offset = 6h) [Reset = 0h]

CTM_MAX_COUNT is shown in Figure 14-27 and described in Table 14-30.

Return to the Summary Table.

Counter Max Count Value Register

Figure 14-27. CTM_MAX_COUNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAX_COUNT																															
R/W-0h																															

Table 14-30. CTM_MAX_COUNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MAX_COUNT	R/W	0h	This register contains the maximum recorded counter value. This is relevant only in the Start Stop mode of operation. This register is writable by CPU only if application owns the unit and if EALLOW is set. Otherwise, the writes are ignored. The register is writable by the debugger only if the debugger owns this unit. Otherwise, the writes are ignored. Reset type: ERAD_RESET

14.9.4.6 CTM_INPUT_SEL Register (Offset = 8h) [Reset = 0h]

CTM_INPUT_SEL is shown in Figure 14-28 and described in Table 14-31.

Return to the Summary Table.

Counter Input Select Register

Figure 14-28. CTM_INPUT_SEL Register

15	14	13	12	11	10	9	8
RESERVED	STA_INP_SEL						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
RESERVED	CNT_INP_SEL						
R-0h	R/W-0h						

Table 14-31. CTM_INPUT_SEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14-8	STA_INP_SEL	R/W	0h	These bits decide which of the inputs will be selected as the START event for the counter. These inputs will be hooked up to the event outputs from the breakpoint module, counter module and to other system events. The usage of these bits are relevant only in the START_STOP mode of counting. Reset type: ERAD_RESET
7	RESERVED	R	0h	Reserved
6-0	CNT_INP_SEL	R/W	0h	These bits decide which of the inputs will be selected to enable counting. These inputs will be hooked up to the event outputs from the breakpoint module, counter module and to other system events Reset type: ERAD_RESET

14.9.4.7 CTM_CLEAR Register (Offset = 9h) [Reset = 0h]

CTM_CLEAR is shown in Figure 14-29 and described in Table 14-32.

Return to the Summary Table.

Counter Clear Register

Figure 14-29. CTM_CLEAR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						OVERFLOW_CLEAR	EVENT_CLEAR
R-0h						R-0/W-0h	R-0/W-0h

Table 14-32. CTM_CLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R	0h	Reserved
1	OVERFLOW_CLEAR	R-0/W	0h	Clear OVERFLOW: 0 No action. 1 A write with this bit set to 1 will clear the sticky OVERFLOW bit in the CTM_STATUS register. Reads of this bit position will always return a 0. Reset type: ERAD_RESET
0	EVENT_CLEAR	R-0/W	0h	Clear EVENT_FIRED: 0 No action. 1 A write with this bit set to 1 will clear the sticky EVENT_FIRED bit in the CTM_STATUS register and bring the Breakpoint Module statemachine status back to IDLE. Reads of this bit position will always return a 0. Reset type: ERAD_RESET

14.9.4.8 CTM_INPUT_SEL_2 Register (Offset = Ah) [Reset = 0h]

CTM_INPUT_SEL_2 is shown in Figure 14-30 and described in Table 14-33.

Return to the Summary Table.

Counter Input Select Extension Register

Figure 14-30. CTM_INPUT_SEL_2 Register

15	14	13	12	11	10	9	8
RESERVED	RST_INP_SEL						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
RESERVED	STO_INP_SEL						
R-0h	R/W-0h						

Table 14-33. CTM_INPUT_SEL_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14-8	RST_INP_SEL	R/W	0h	These bits decide are used to select the event input that will be used as the reset input. These bits matter only if the Enable Reset bit is set to 1. Reset type: ERAD_RESET
7	RESERVED	R	0h	Reserved
6-0	STO_INP_SEL	R/W	0h	These bits decide which of the inputs will be selected as the STOP event for the counter. These inputs will be hooked up to the event outputs from the breakpoint module, counter module and to other system events. The usage of these bits are relevant only in the START_STOP mode of counting. Reset type: ERAD_RESET

14.9.4.9 CTM_INPUT_COND Register (Offset = Bh) [Reset = 0h]

CTM_INPUT_COND is shown in Figure 14-31 and described in Table 14-34.

Return to the Summary Table.

Counter Input Conditioning Register

Figure 14-31. CTM_INPUT_COND Register

15	14	13	12	11	10	9	8
RESERVED		RST_INP_SYNCH	RST_INP_INV	RESERVED		STO_INP_SYNCH	STO_INP_INV
R-0h		R/W-0h	R/W-0h	R-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED		STA_INP_SYNCH	STA_INP_INV	RESERVED		CTM_INP_SYNCH	CTM_INP_INV
R-0h		R/W-0h	R/W-0h	R-0h		R/W-0h	R/W-0h

Table 14-34. CTM_INPUT_COND Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	RESERVED	R	0h	Reserved
13	RST_INP_SYNCH	R/W	0h	Enable the 2-stage synchronizer for the selected Reset input Reset type: ERAD_RESET
12	RST_INP_INV	R/W	0h	Invert the Selected Reset input Reset type: ERAD_RESET
11-10	RESERVED	R	0h	Reserved
9	STO_INP_SYNCH	R/W	0h	Enable the 2-stage synchronizer for the selected Stop input Reset type: ERAD_RESET
8	STO_INP_INV	R/W	0h	Invert the Selected Stop input Reset type: ERAD_RESET
7-6	RESERVED	R	0h	Reserved
5	STA_INP_SYNCH	R/W	0h	Enable the 2-stage synchronizer for the selected Start input Reset type: ERAD_RESET
4	STA_INP_INV	R/W	0h	Invert the Selected Start input Reset type: ERAD_RESET
3-2	RESERVED	R	0h	Reserved
1	CTM_INP_SYNCH	R/W	0h	Enable the 2-stage synchronizer for the selected Counter input Reset type: ERAD_RESET
0	CTM_INP_INV	R/W	0h	Invert the Selected Counter input Reset type: ERAD_RESET

14.9.5 ERAD_CRC_GLOBAL_REGS Registers

Table 14-35 lists the memory-mapped registers for the ERAD_CRC_GLOBAL_REGS registers. All register offset addresses not listed in Table 14-35 should be considered as reserved locations and the register contents should not be modified.

Table 14-35. ERAD_CRC_GLOBAL_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CRC_GLOBAL_CTRL	CRC_GLOBAL_CRTL		Go

Complex bit access types are encoded to fit into small table cells. Table 14-36 shows the codes that are used for access types in this section.

Table 14-36. ERAD_CRC_GLOBAL_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

14.9.5.1 CRC_GLOBAL_CTRL Register (Offset = 0h) [Reset = 0h]

CRC_GLOBAL_CTRL is shown in Figure 14-32 and described in Table 14-37.

Return to the Summary Table.

CRC Global Control Register

Figure 14-32. CRC_GLOBAL_CTRL Register

15	14	13	12	11	10	9	8
CRC8_EN	CRC7_EN	CRC6_EN	CRC5_EN	CRC4_EN	CRC3_EN	CRC2_EN	CRC1_EN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CRC8_INIT	CRC7_INIT	CRC6_INIT	CRC5_INIT	CRC4_INIT	CRC3_INIT	CRC2_INIT	CRC1_INIT
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h

Table 14-37. CRC_GLOBAL_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	CRC8_EN	R/W	0h	0 = No action. 1 = Corresponding CRC Module is enabled and a valid event on this interface starts CRC computation Reset type: SYSRSn
14	CRC7_EN	R/W	0h	0 = No action. 1 = Corresponding CRC Module is enabled and a valid event on this interface starts CRC computation Reset type: SYSRSn
13	CRC6_EN	R/W	0h	0 = No action. 1 = Corresponding CRC Module is enabled and a valid event on this interface starts CRC computation Reset type: SYSRSn
12	CRC5_EN	R/W	0h	0 = No action. 1 = Corresponding CRC Module is enabled and a valid event on this interface starts CRC computation Reset type: SYSRSn
11	CRC4_EN	R/W	0h	0 = No action. 1 = Corresponding CRC Module is enabled and a valid event on this interface starts CRC computation Reset type: SYSRSn
10	CRC3_EN	R/W	0h	0 = No action. 1 = Corresponding CRC Module is enabled and a valid event on this interface starts CRC computation Reset type: SYSRSn
9	CRC2_EN	R/W	0h	0 = No action. 1 = Corresponding CRC Module is enabled and a valid event on this interface starts CRC computation Reset type: SYSRSn
8	CRC1_EN	R/W	0h	0 = No action. 1 = Corresponding CRC Module is enabled and a valid event on this interface starts CRC computation Reset type: SYSRSn
7	CRC8_INIT	R-0/W	0h	0 = No action. 1 = Initialize the CRC Module (Module registers/statemachine gets cleared for a fresh start) Reads of this bit position always returns zero Reset type: SYSRSn
6	CRC7_INIT	R-0/W	0h	0 = No action. 1 = Initialize the CRC Module (Module registers/statemachine gets cleared for a fresh start) Reads of this bit position always returns zero Reset type: SYSRSn

Table 14-37. CRC_GLOBAL_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	CRC6_INIT	R-0/W	0h	0 = No action. 1 = Initialize the CRC Module (Module registers/statemachine gets cleared for a fresh start) Reads of this bit position always returns zero Reset type: SYSRSn
4	CRC5_INIT	R-0/W	0h	0 = No action. 1 = Initialize the CRC Module (Module registers/statemachine gets cleared for a fresh start) Reads of this bit position always returns zero Reset type: SYSRSn
3	CRC4_INIT	R-0/W	0h	0 = No action. 1 = Initialize the CRC Module (Module registers/statemachine gets cleared for a fresh start) Reads of this bit position always returns zero Reset type: SYSRSn
2	CRC3_INIT	R-0/W	0h	0 = No action. 1 = Initialize the CRC Module (Module registers/statemachine gets cleared for a fresh start) Reads of this bit position always returns zero Reset type: SYSRSn
1	CRC2_INIT	R-0/W	0h	0 = No action. 1 = Initialize the CRC Module (Module registers/statemachine gets cleared for a fresh start) Reads of this bit position always returns zero Reset type: SYSRSn
0	CRC1_INIT	R-0/W	0h	0 = No action. 1 = Initialize the CRC Module (Module registers/statemachine gets cleared for a fresh start) Reads of this bit position always returns zero Reset type: SYSRSn

14.9.6 ERAD_CRC_REGS Registers

Table 14-38 lists the memory-mapped registers for the ERAD_CRC_REGS registers. All register offset addresses not listed in Table 14-38 should be considered as reserved locations and the register contents should not be modified.

Table 14-38. ERAD_CRC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CRC_CURRENT	CRC_CURRENT		Go
2h	CRC_SEED	CRC SEED value		Go
4h	CRC_QUALIFIER	CRC_QUALIFIER		Go

Complex bit access types are encoded to fit into small table cells. Table 14-39 shows the codes that are used for access types in this section.

Table 14-39. ERAD_CRC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

14.9.6.1 CRC_CURRENT Register (Offset = 0h) [Reset = 0h]

CRC_CURRENT is shown in Figure 14-33 and described in Table 14-40.

Return to the Summary Table.

Current computed CRC value

Figure 14-33. CRC_CURRENT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC_CURRENT																															
R-0h																															

Table 14-40. CRC_CURRENT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CRC_CURRENT	R	0h	Reads the current CRC value Reset type: SYSRSn

14.9.6.2 CRC_SEED Register (Offset = 2h) [Reset = 0h]

CRC_SEED is shown in Figure 14-34 and described in Table 14-41.

Return to the Summary Table.

CRC SEED value

Figure 14-34. CRC_SEED Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC_SEED																															
R/W-0h																															

Table 14-41. CRC_SEED Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CRC_SEED	R/W	0h	CRC Seed Register Reset type: SYSRSn

14.9.6.3 CRC_QUALIFIER Register (Offset = 4h) [Reset = 0h]

CRC_QUALIFIER is shown in Figure 14-35 and described in Table 14-42.

Return to the Summary Table.

CRC compute enable register

Figure 14-35. CRC_QUALIFIER Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			CRC_QUALIFIER				
R-0h			R/W-0h				

Table 14-42. CRC_QUALIFIER Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4-0	CRC_QUALIFIER	R/W	0h	0000 = No Qualifier, every valid event is qualified for CRC computation 00001 = CRC Compute Qualified by HWBP_EVENT1 00010 = CRC Compute Qualified by HWBP_EVENT2 00011 = CRC Compute Qualified by HWBP_EVENT3 00100 = CRC Compute Qualified by HWBP_EVENT4 00101 = CRC Compute Qualified by HWBP_EVENT5 00110 = CRC Compute Qualified by HWBP_EVENT6 00111 = CRC Compute Qualified by HWBP_EVENT7 01000 = CRC Compute Qualified by HWBP_EVENT8 01001 = CRC Compute Qualified by HWBP_EVENT_OR1 01010 = CRC Compute Qualified by HWBP_EVENT_OR2 01011 = CRC Compute Qualified by HWBP_EVENT_OR3 01100 = CRC Compute Qualified by HWBP_EVENT_OR4 01101 = CRC Compute Qualified by HWBP_EVENT_AND1 01110 = CRC Compute Qualified by HWBP_EVENT_AND2 01111 = CRC Compute Qualified by HWBP_EVENT_AND3 10000 = CRC Compute Qualified by HWBP_EVENT_AND4 Others = No Qualifier, every valid event is qualified for CRC computation Reset type: SYSRSn

14.9.7 ERAD Registers to Driverlib Functions

Table 14-43. ERAD Registers to Driverlib Functions

File	Driverlib Function
GLBL_EVENT_STAT	
erad.h	ERAD_getEventStatus
GLBL_HALT_STAT	
erad.h	ERAD_getHaltStatus
GLBL_ENABLE	
erad.h	ERAD_enableModules
erad.h	ERAD_disableModules
GLBL_CTM_RESET	
erad.h	ERAD_resetCounter
GLBL_NMI_CTL	
erad.h	ERAD_enableNMI
erad.h	ERAD_disableNMI

Table 14-43. ERAD Registers to Driverlib Functions (continued)

File	Driverlib Function
GLBL_OWNER	
erad.h	ERAD_getOwnership
erad.h	ERAD_setOwnership
GLBL_EVENT_AND_MASK	
erad.c	ERAD_configMask
GLBL_EVENT_OR_MASK	
erad.c	ERAD_configMask
GLBL_AND_EVENT_INT_MASK	
erad.c	ERAD_configMask
GLBL_OR_EVENT_INT_MASK	
erad.c	ERAD_configMask
HWBP_MASK	
erad.c	ERAD_configBusComp
HWBP_REF	
erad.c	ERAD_configBusComp
HWBP_CLEAR	
erad.h	ERAD_clearBusCompEvent
HWBP_CNTL	
erad.c	ERAD_configBusComp
HWBP_STATUS	
erad.h	ERAD_getBusCompStatus
CTM_CNTL	
erad.c	ERAD_configCounterInCountingMode
erad.c	ERAD_configCounterInStartStopMode
erad.c	ERAD_configCounterInCumulativeMode
erad.h	ERAD_enableCounterResetInput
erad.h	ERAD_disableCounterResetInput
CTM_STATUS	
erad.h	ERAD_getCounterStatus
CTM_REF	
erad.c	ERAD_configCounterInCountingMode
erad.c	ERAD_configCounterInStartStopMode
erad.c	ERAD_configCounterInCumulativeMode
CTM_COUNT	
erad.h	ERAD_getCurrentCount
erad.h	ERAD_setCurrentCount
CTM_MAX_COUNT	
erad.h	ERAD_getMaxCount
erad.h	ERAD_setMaxCount
CTM_INPUT_SEL	
erad.c	ERAD_configCounterInCountingMode
erad.c	ERAD_configCounterInStartStopMode
erad.c	ERAD_configCounterInCumulativeMode
erad.h	ERAD_enableCounterResetInput
CTM_CLEAR	

Table 14-43. ERAD Registers to Driverlib Functions (continued)

File	Driverlib Function
erad.h	ERAD_clearCounterEvent
erad.h	ERAD_clearCounterOverflow
CTM_INPUT_SEL_2	
erad.c	ERAD_configCounterInStartStopMode
erad.c	ERAD_configCounterInCumulativeMode
erad.h	ERAD_enableCounterResetInput
CTM_INPUT_COND	
erad.h	ERAD_setCounterInputConditioning
CRC_GLOBAL_CTRL	
erad.h	ERAD_initCRC
erad.h	ERAD_enableCRC
erad.h	ERAD_disableCRC
erad.h	ERAD_setSeed
erad.h	ERAD_setCRCQualifier
CRC_CURRENT	
erad.h	ERAD_getCurrentCRC
CRC_SEED	
erad.h	ERAD_setSeed
CRC_QUALIFIER	
erad.h	ERAD_setCRCQualifier

Chapter 15

General-Purpose Input/Output (GPIO)

The GPIO module controls the device's digital multiplexing, which uses shared pins to maximize application flexibility. The pins are named by their general-purpose I/O name (for example, GPIO0, GPIO25, GPIO58). These pins can be individually selected to operate as digital I/O (also called GPIO mode), or connected to one of several peripheral I/O signals. The input signals can be qualified to remove unwanted noise.

15.1 Introduction

Up to twelve independent peripheral signals are multiplexed on a single GPIO-enabled pin in addition to the CPU-controlled I/O capability. Each pin output can be controlled by either a peripheral or one of the four CPU masters.

- CPU1
- CPU1.CLA
- CPU2
- CPU2.CLA

There are up to 8 possible I/O ports:

- Port A consists of GPIO0-GPIO31
- Port B consists of GPIO32-GPIO63
- Port C consists of GPIO64-GPIO95
- Port D consists of GPIO96-GPIO127
- Port E consists of GPIO128-GPIO159
- Port F consists of GPIO160-GPIO191
- Port G consists of GPIO192-GPIO223
- Port H consists of GPIO224-GPIO255

Note

Some GPIO and I/O ports can be unavailable on particular devices. See the *GPIO Registers* section for available GPIO and I/O ports.

Figure 15-1 shows the GPIO logic for a single pin.

Note

High-speed SPI and AUXCLKIN use a different signal path that do not support inversion or qualification. For more details on high-speed SPI pins, see Section 15.6.

The USB PHY pin muxing is not shown in Figure 15-1. For more details on USB pins, see Section 15.5.

There are two key features to note in Figure 15-1. The first is that the input and output paths are entirely separate, connecting only at the pin. The second is that peripheral muxing takes place far from the pin. As a result, it is always possible for both CPUs and CLAs to read the physical state of the pin independent of CPU mastering and peripheral muxing. Likewise, external interrupts can be generated from peripheral activity. All pin options such as input qualification and open-drain output are valid for all masters and peripherals. However, the peripheral muxing, CPU muxing, and pin options can only be configured by CPU1.Table 15-1 provides details of GPIO registers accessible by different masters.

A separate configuration is required for the USB signals. See Section 15.5 for details.

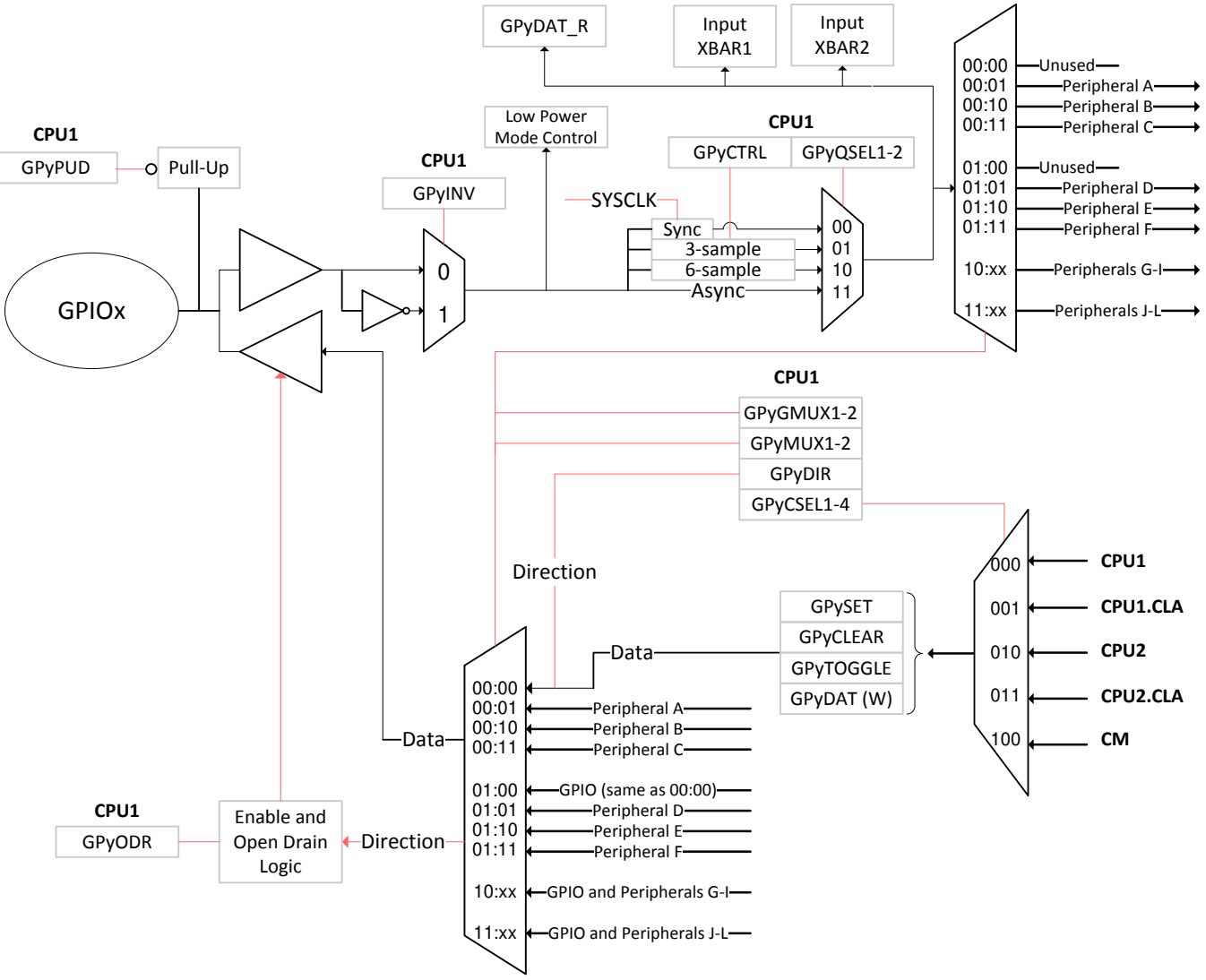


Figure 15-1. GPIO Logic for a Single Pin

Table 15-1. GPIO access by different controllers

Register Type	Function	CPU	CLA	DMA	HIC	Comments
GPIO_CTRL	Peripheral muxing, Pull Control ,etc.	Yes	NO	NO	NO	-
GPIO_DATA	GPIODAT, SET, CLEAR, TOGGLE, and pin status, etc.	Yes	Yes	NO	No	Based on GPxCSEL configuration.
GPIO_DATA_READ	Read back of GPIODAT register	Yes	Yes	NO	Yes	-

15.1.1 GPIO Related Collateral

Foundational Materials

- C2000 Academy - GPIO

Getting Started Materials

- *How to Maximize GPIO Usage in C2000 Devices Application Report*
- [FAQ] C2000 GPIO FAQ

15.2 Configuration Overview

I/O pin configuration consists of several steps:

1. **Plan the device pin-out**
   Make a list of all required peripherals for the application. Using the peripheral mux information in the device data manual, choose which GPIOs to use for the peripheral signals. Decide which of the remaining GPIOs to use as inputs and outputs for each CPU and CLA.

   Once the peripheral muxing has been chosen, implement the mux by writing the appropriate values to the GPyMUX1/2 and GPyGMUX1/2 registers. When changing the GPyGMUX value for a pin, always set the corresponding GPyMUX bits to zero first to avoid glitching in the muxes. By default, all pins are general-purpose I/Os, not peripheral signals.
2. **(Optional) Enable internal pullup resistors**
   To enable or disable the pullup resistors, write to the appropriate bits in the GPIO pullup disable registers (GPyPUD). All pullups are disabled by default. Pullups can be used to keep input pins in a known state when there is no external signal driving them.
3. **Select input qualification**
   If the pin is used as an input, specify the required input qualification, if any. The input qualification sampling period is selected in the GPyCTRL registers, while the type of qualification is selected in the GPyQSEL1 and GPyQSEL2 registers. By default, all qualification is synchronous with a sampling period equal to PLLSYSCLK. For an explanation of input qualification, see Section 15.4.
4. **Select the direction of any general-purpose I/O pins**
   For each pin configured as a GPIO, specify the direction of the pin as either input or output using the GPyDIR registers. By default, all GPIO pins are inputs. Before changing a pin to an output, load the output latch with the value to be driven by writing that value to the GPySET, GPyCLEAR, or GPyDAT registers. Once the latch is loaded, write to GPyDIR to change the pin direction. By default, all output latches are zero.

   The GPyDAT_R register can be used to read what value was written to the GPyDAT register.
5. **Select low-power mode wake-up sources**
   GPIOs 0-63 can be used to wake the system up from low power modes. To select one or more GPIOs for wake-up, write to the appropriate bits in the GPIOLPMSEL0 and GPIOLPMSEL1 registers. These registers are part of the CPU system register space. For more information on low-power modes and GPIO wake-up, see the Low-Power Modes section in the *System Control and Interrupts* chapter.
6. **Select external interrupt sources**
   Configuring external interrupts is a two-step process. First, the interrupts themselves must be enabled and their polarity must be configured using the XINTnCR registers. Second, the XINT1-5 GPIO pins must be set by selecting the sources for Input X-BAR signals 4, 5, 6, 13, and 14, respectively. For more information on the Input X-BAR architecture, see the *Crossbar (X-BAR)* chapter.

15.3 Digital General-Purpose I/O Control

The values on the pins that are configured as GPIO can be changed by using the following registers.

- **GPyDAT Registers**

  Each I/O port has one data register. Each bit in the data register corresponds to one GPIO pin. No matter how the pin is configured (GPIO or peripheral function), the corresponding bit in the data register reflects the current state of the pin after qualification. Writing to the GPyDAT register clears or sets the corresponding output latch and if the pin is enabled as a general-purpose output (GPIO output), the pin is also driven either low or high. If the pin is not configured as a GPIO output, then the value is latched but the pin is not driven. Only if the pin is later configured as a GPIO output is the latched value driven onto the pin.

  When using the GPyDAT register to change the level of an output pin, be cautious to not accidentally change the level of another pin. For example, to change the output latch level of GPIOA1 by writing to the GPADAT register bit 0 using a read-modify-write instruction, a problem can occur if another I/O port A signal changes level between the read and the write stage of the instruction. Following is an analysis of why this happens:

  The GPyDAT registers reflect the state of the pin, not the latch. This means the register reflects the actual pin value. However, there is a lag between when the register is written to when the new pin value is reflected back in the register. This can pose a problem when this register is used in subsequent program statements to alter the state of GPIO pins. An example is shown below where two program statements attempt to drive two different GPIO pins that are currently low to a high state.

  If Read-Modify-Write operations are used on the GPyDAT registers, because of the delay between the output and the input of the first instruction (I1), the second instruction (I2) reads the old value and writes the value back.

```
GpioDataRegs.GPADAT.bit.GPIO1 = 1;  //I1 performs read-modify-write of GPADAT
GpioDataRegs.GPADAT.bit.GPIO2 = 1;  //I2 also a read-modify-write of GPADAT
                                    //GPADAT gets the old value of GPIO1 due to the delay
```

  The second instruction waits for the first to finish the write due to the write-followed-by-read protection on this peripheral frame. There is some lag, however, between the write of (I1) and the GPyDAT bit reflecting the new value (1) on the pin. During this lag, the second instruction reads the old value of GPIO1 (0) and writes the value back along with the new value of GPIO2 (1). Therefore, GPIO1 pin stays low.

  One answer is to put some NOPs between instructions. A better answer is to use the GPySET/GPyCLEAR/GPyTOGGLE registers instead of the GPyDAT registers. These registers always read back a 0 and writes of 0 have no effect. Only bits that need to be changed can be specified without disturbing any other bits that are currently in the process of changing.

- **GPyDAT_R Registers**

  The GPyDAT_R registers are read only registers that return the value written to the GPyDAT registers instead of pin status. Writes to these registers have no effect.

- **GPySET Registers**

  The set registers are used to drive specified GPIO pins high without disturbing other pins. Each I/O port has one set register and each bit corresponds to one GPIO pin. The set registers always read back 0. If the corresponding pin is configured as an output, then writing a 1 to that bit in the set register sets the output latch high and the corresponding pin is driven high. If the pin is not configured as a GPIO output, then the value is latched but the pin is not driven. Only if the pin is later configured as a GPIO output is the latched value driven onto the pin. Writing a 0 to any bit in the set registers has no effect.

- **GPyCLEAR Registers**

  The clear registers are used to drive specified GPIO pins low without disturbing other pins. Each I/O port has one clear register. The clear registers always read back 0. If the corresponding pin is configured as a general-purpose output, then writing a 1 to the corresponding bit in the clear register clears the output latch and the pin is driven low. If the pin is not configured as a GPIO output, then the value is latched but the pin is not driven. Only if the pin is later configured as a GPIO output is the latched value driven onto the pin. Writing a 0 to any bit in the clear registers has no effect.
- **GPyTOGGLE Registers**

  The toggle registers are used to drive specified GPIO pins to the opposite level without disturbing other pins. Each I/O port has one toggle register. The toggle registers always read back 0. If the corresponding pin is configured as an output, then writing a 1 to that bit in the toggle register flips the output latch and pulls the corresponding pin in the opposite direction. That is, if the output pin is driven low, then writing a 1 to the corresponding bit in the toggle register pulls the pin high. Likewise, if the output pin is high, then writing a 1 to the corresponding bit in the toggle register pulls the pin low. If the pin is not configured as a GPIO output, then the value is latched but the pin is not driven. Only if the pin is later configured as a GPIO output is the latched value driven onto the pin. Writing a 0 to any bit in the toggle registers has no effect.

15.4 Input Qualification

The input qualification scheme has been designed to be very flexible. Select the type of input qualification for each GPIO pin by configuring the GPyQSEL1 and GPyQSEL2 registers. In the case of a GPIO input pin, the qualification can be specified as only synchronized to SYSCLKOUT or qualification by a sampling window. For pins that are configured as peripheral inputs, the input can also be asynchronous in addition to synchronized to SYSCLKOUT or qualified by a sampling window. The remainder of this section describes the options available.

15.4.1 No Synchronization (Asynchronous Input)

This mode is used for peripherals where input synchronization is not required or the peripheral itself performs the synchronization. Examples include communication ports McBSP, SCI, SPI, and I^2C. In addition, the ePWM trip zone ($\overline{TZn}$) signals can function independent of the presence of SYSCLKOUT.

Note

Using input synchronization when the peripheral itself performs the synchronization can cause unexpected results. The user must make sure that the GPIO pin is configured for asynchronous in this case.

15.4.2 Synchronization to SYSCLKOUT Only

This is the default qualification mode of all the pins at reset. In this mode, the input signal is only synchronized to the system clock (SYSCLKOUT). Because the incoming signal is asynchronous, a SYSCLKOUT period of delay is needed for the input to the device to be changed. No further qualification is performed on the signal.

15.4.3 Qualification Using a Sampling Window

In this mode, the signal is first synchronized to the system clock (SYSCLKOUT) and then qualified by a specified number of cycles before the input is allowed to change. Figure 15-2 and Figure 15-3 show how the input qualification is performed to eliminate unwanted noise. Two parameters are specified by the user for this type of qualification: 1) the sampling period, or how often the signal is sampled, and 2) the number of samples to be taken.

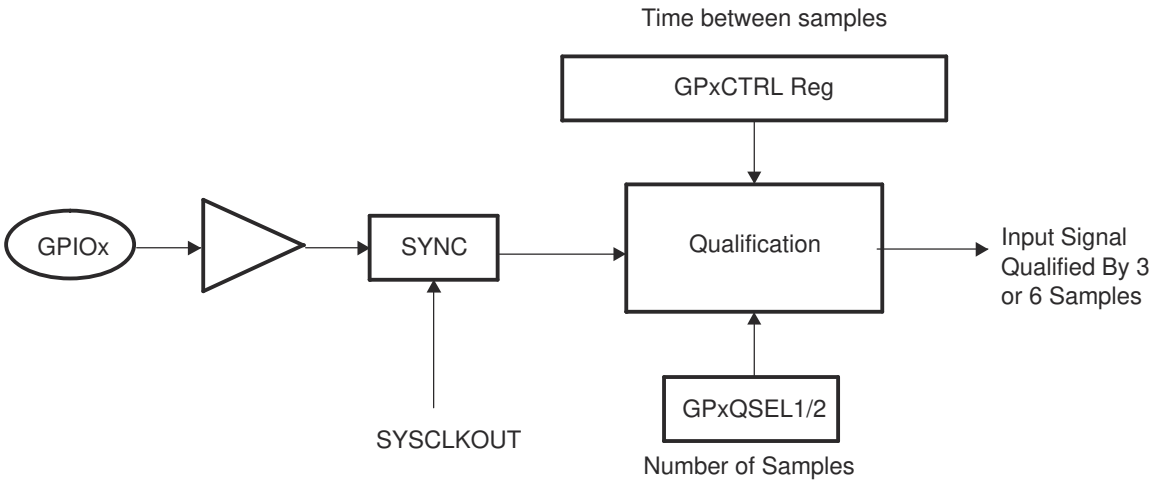


Figure 15-2. Input Qualification Using a Sampling Window

To qualify the signal, the input signal is sampled at a regular period. The sampling period is specified by the user and determines the time duration between samples, or how often the signal is sampled, relative to the CPU clock (SYSCLKOUT).

The sampling period is specified by the qualification period (QUALPRDn) bits in the GPxCTRL register. The sampling period is configurable in groups of 8 input signals. For example, GPIO0 to GPIO7 use GPACTRL[QUALPRD0] setting and GPIO8 to GPIO15 use GPACTRL[QUALPRD1]. Table 15-2 and Table 15-3 show the relationship between the sampling period or sampling frequency and the GPxCTRL[QUALPRDn] setting.

Table 15-2. Sampling Period

	Sampling Period
If GPxCTRL[QUALPRDn] = 0	$1 \times T_{SYSCLKOUT}$
If GPxCTRL[QUALPRDn] ≠ 0	$2 \times$ GPxCTRL[QUALPRDn] $\times T_{SYSCLKOUT}$
	Where $T_{SYSCLKOUT}$ is the period in time of SYSCLKOUT

Table 15-3. Sampling Frequency

	Sampling Frequency
If GPxCTRL[QUALPRDn] = 0	$f_{SYSCLKOUT}$
If GPxCTRL[QUALPRDn] ≠ 0	$f_{SYSCLKOUT} \times 1 \div (2 \times$ GPxCTRL[QUALPRDn])
	Where $f_{SYSCLKOUT}$ is the frequency of SYSCLKOUT

From these equations, the minimum and maximum time between samples can be calculated for a given SYSCLKOUT frequency:

Example: Maximum Sampling Frequency:

If GPxCTRL[QUALPRDn] = 0

then the sampling frequency is $f_{SYSCLKOUT}$

If, for example, $f_{SYSCLKOUT}$ = 60 MHz

then the signal is sampled at 60 MHz or one sample every 16.67 ns.

Example: Minimum Sampling Frequency:

If GPxCTRL[QUALPRDn] = 0xFF (255)

then the sampling frequency is $f_{SYSCLKOUT} \times 1 \div (2 \times$ GPxCTRL[QUALPRDn])

If, for example, $f_{SYSCLKOUT}$ = 60 MHz

then the signal is sampled at 60 MHz × 1 ÷ (2 × 255) (117.647 kHz) or one sample every 8.5 µs.

Number of samples:

The number of times the signal is sampled is either three samples or six samples as specified in the qualification selection (GPAQSEL1, GPAQSEL2, GPBQSEL1, and GPBQSEL2) registers. When three or six consecutive cycles are the same, then the input change is passed through to the device.

Total Sampling Window Width:

The sampling window is the time during which the input signal is sampled as shown in Figure 15-3. By using the equation for the sampling period, along with the number of samples to be taken, the total width of the window can be determined.

For the input qualifier to detect a change in the input, the level of the signal must be stable for the duration of the sampling window width or longer.

The number of sampling periods within the window is always one less than the number of samples taken. For a three-sample window, the sampling window width is two sampling periods wide where the sampling period is defined in Table 15-2. Likewise, for a six-sample window, the sampling window width is five sampling periods wide. Table 15-4 and Case 2: Six-Sample Sampling Window Width show the calculations used to determine the total sampling window width based on GPxCTRL[QUALPRDn] and the number of samples taken.

Table 15-4. Case 1: Three-Sample Sampling Window Width

	Total Sampling Window Width
If GPxCTRL[QUALPRDn] = 0	$2 \times T_{SYSCLKOUT}$
If GPxCTRL[QUALPRDn] ≠ 0	$2 \times 2 \times$ GPxCTRL[QUALPRDn] $\times T_{SYSCLKOUT}$
	Where $T_{SYSCLKOUT}$ is the period in time of SYSCLKOUT

Table 15-5. Case 2: Six-Sample Sampling Window Width

	Total Sampling Window Width
If GPxCTRL[QUALPRDn] = 0	$5 \times T_{SYSCLKOUT}$
If GPxCTRL[QUALPRDn] ≠ 0	$5 \times 2 \times$ GPxCTRL[QUALPRDn] $\times T_{SYSCLKOUT}$
	Where $T_{SYSCLKOUT}$ is the period in time of SYSCLKOUT

Note

The external signal change is asynchronous with respect to both the sampling period and SYSCLKOUT. Due to the asynchronous nature of the external signal, the input must be held stable for a time greater than the sampling window width to make sure the logic detects a change in the signal. The extra time required can be up to an additional sampling period + $T_{SYSCLKOUT}$.

The required duration for an input signal to be stable for the qualification logic to detect a change is described in the data sheet.

For the example shown in Figure 15-3, the input qualification has been configured as follows:

- GPxQSEL1/2 = 1,0. This indicates a six-sample qualification.
- GPxCTRL[QUALPRDn] = 1. The sampling period is $t_w(SP) = 2 \times GPxCTRL[QUALPRDn] \times T_{SYSCLKOUT} = 2 \times T_{SYSCLKOUT}$.

This configuration results in the following:

- The width of the sampling window is:

  $t_w(IQSW) = 5 \times t_w(SP) = 5 \times 2 \times GPxCTRL[QUALPRDn] \times T_{SYSCLKOUT} = 5 \times 2 \times T_{SYSCLKOUT}$

- If, for example, $T_{SYSCLKOUT}$ = 16.67 ns, then the duration of the sampling window is:

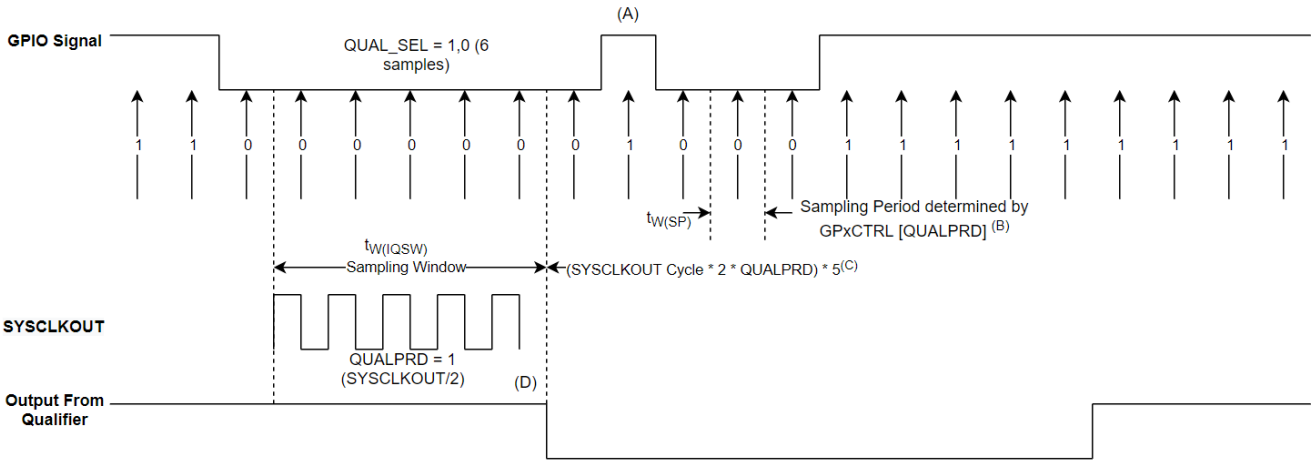
  Sampling period, $t_w(SP) = 2 \times T_{SYSCLKOUT} = 2 \times 16.67$ ns = 33.3 ns

  Sampling window, $t_w(IQSW) = 5 \times t_w(SP) = 5 \times 33.3$ ns = 166.7 ns

- To account for the asynchronous nature of the input relative to the sampling period and SYSCLKOUT, up to a single additional sampling period and SYSCLK period is required to detect a change in the input signal. For this example:

  $t_w(IQSW) + t_w(SP) + T_{SYSCLKOUT} = 166.7$ ns + 33.3 ns + 16.67 ns = 216.7 ns

- In Figure 15-3, the glitch (A) is shorter then the qualification window and is ignored by the input qualifier.



A. This glitch will be ignored by the input qualifier. The QUALPRD bit field specifies the qualification sampling period. It can vary from 00 to 0xFF. If QUALPRD = 00, then the sampling period is 1 SYSCLKOUT cycle. For any other value "n", the qualification sampling period in 2n SYSCLKOUT cycles (i.e., at every 2n SYSCLKOUT cycles, the GPIO pin will be sampled).
B. The qualification period selected via the GPxCTRL register applies to groups of 8 GPIO pins.
C. The qualification block can take either three or six samples. The QUAL_SEL Register selects which sample mode is used.
D. In the example shown, for the qualifier to detect the change, the input should be stable for 10 SYSCLKOUT cycles or greater. In other words, the inputs should be stable for (5 x QUALPRD x 2) SYSCLKOUT cycles. That would ensure 5 sampling periods for detection to occur. Since external signals are driven asynchronously, an 13-SYSCLKOUT-wide pulse ensures reliable recognition.

Figure 15-3. Input Qualifier Clock Cycles

15.5 USB Signals

The USB module on this device has an internal physical layer transceiver (PHY). The I/O signals are not normal digital signals, and as a result, the signals do not connect to the pins through the normal GPIO mux path. Instead, a special analog mux is used. To connect the USB signals to the device pins, set the GPyAMSEL bits appropriately as shown in Table 15-6. Do not enable pullups or any other special pin option when using the USB signals.

Table 15-6. USB I/O Signal Muxing

Signal	GPIO	AMSEL
USBDM	42	GPBAMSEL[10]
USBDP	43	GPBAMSEL[11]

15.6 SPI Signals

The SPI module on this device has a high-speed mode that enables 40 Mbps communication. To achieve the highest possible speed, a special GPIO configuration is used on a single GPIO mux option for each SPI. These GPIOs can also be used by the SPI when not in high-speed mode (HS_MODE = 0). Table 15-7 shows which GPIOs have the special mux option to allow SPI high-speed mode.

To select these mux options, configure the GPyGMUX and GPyMUX registers as shown in Table 15-7.

Table 15-7. GPIO Configuration for High-Speed SPI

GPIO	SPI Signal	Mux Configuration	
GPIO58	SPISIMOA	GPBGMUX2[21:20]=11	GPBMUX2[21:20]=11
GPIO59	SPISOMIA	GPBGMUX2[23:22]=11	GPBMUX2[23:22]=11
GPIO60	SPICLKA	GPBGMUX2[25:24]=11	GPBMUX2[25:24]=11
GPIO61	$\overline{\text{SPISTEA}}$	GPBGMUX2[27:26]=11	GPBMUX2[27:26]=11
GPIO63	SPISIMOB	GPBGMUX2[31:30]=11	GPBMUX2[31:30]=11
GPIO64	SPISOMIB	GPCGMUX1[1:0]=11	GPCMUX1[1:0]=11
GPIO65	SPICLKB	GPCGMUX1[3:2]=11	GPCMUX1[3:2]=11
GPIO66	$\overline{\text{SPISTEB}}$	GPCGMUX1[5:4]=11	GPCMUX1[5:4]=11
GPIO69	SPISIMOC	GPCGMUX1[11:10]=11	GPCMUX1[11:10]=11
GPIO70	SPISOMIC	GPCGMUX1[13:12]=11	GPCMUX1[13:12]=11
GPIO71	SPICLKC	GPCGMUX1[15:14]=11	GPCMUX1[15:14]=11
GPIO72	$\overline{\text{SPISTEC}}$	GPCGMUX1[17:16]=11	GPCMUX1[17:16]=11
GPIO91	SPISIMOD	GPCGMUX2[23:22]=11	GPCMUX2[23:22]=11
GPIO92	SPISOMID	GPCGMUX2[25:24]=11	GPCMUX2[25:24]=11
GPIO93	SPICLKD	GPCGMUX2[27:26]=11	GPCMUX2[27:26]=11
GPIO94	$\overline{\text{SPISTED}}$	GPCGMUX2[29:28]=11	GPCMUX2[29:28]=11

15.7 GPIO and Peripheral Muxing

15.7.1 GPIO Muxing

Up to twelve different peripheral functions are multiplexed to each pin along with a general-purpose input/output (GPIO) function. This allows you to choose the peripheral mix and pinout that works best for your particular application. Refer to Table 15-8 for muxing combinations and definitions.

Table 15-8. GPIO Muxed Pins

0, 4, 8, 12	1	2	3	5	6	7	9	10	11	13	14	15	ALT
GPIO0	EPWM1A				I2CA_SDA		CM-I2CA_SDA	ESC_GPI0		FSITXA_D0			
GPIO1	EPWM1B		MFSRB		I2CA_SCL		CM-I2CA_SCL	ESC_GPI1		FSITXA_D1			
GPIO2	EPWM2A			OUTPUTXBAR1	I2CB_SDA			ESC_GPI2		FSITXA_CLK			
GPIO3	EPWM2B	OUTPUTXBAR2	MCLKRB	OUTPUTXBAR2	I2CB_SCL			ESC_GPI3		FSIRXA_D0			
GPIO4	EPWM3A			OUTPUTXBAR3	CANA_TX		MCAN_TX	ESC_GPI4		FSIRXA_D1			
GPIO5	EPWM3B	MFSRA	OUTPUTXBAR3		CANA_RX		MCAN_RX	ESC_GPI5		FSIRXA_CLK			
GPIO6	EPWM4A	OUTPUTXBAR4	EXTSYNCOUT	EQEP3_A	CANB_TX			ESC_GPI6		FSITXB_D0			
GPIO7	EPWM4B	MCLKRA	OUTPUTXBAR5	EQEP3_B	CANB_RX			ESC_GPI7		FSITXB_D1			
GPIO8	EPWM5A	CANB_TX	ADCSOCAO	EQEP3_ STROBE	SCIA_TX		MCAN_TX	ESC_GPO0		FSITXB_CLK	FSITXA_D1	FSIRXA_D0	
GPIO9	EPWM5B	SCIB_TX	OUTPUTXBAR6	EQEP3_INDEX	SCIA_RX			ESC_GPO1		FSIRXB_D0	FSITXA_D0	FSIRXA_CLK	
GPIO10	EPWM6A	CANB_RX	ADCSOCBO	EQEP1_A	SCIB_TX		MCAN_RX	ESC_GPO2		FSIRXB_D1	FSITXA_CLK	FSIRXA_D1	
GPIO11	EPWM6B	SCIB_RX	OUTPUTXBAR7	EQEP1_B	SCIB_RX			ESC_GPO3		FSIRXB_CLK	FSIRXA_D1		
GPIO12	EPWM7A	CANB_TX	MDXB	EQEP1_ STROBE	SCIC_TX			ESC_GPO4		FSIRXC_D0	FSIRXA_D0		
GPIO13	EPWM7B	CANB_RX	MDRB	EQEP1_INDEX	SCIC_RX			ESC_GPO5		FSIRXC_D1	FSIRXA_CLK		
GPIO14	EPWM8A	SCIB_TX	MCLKXB		OUTPUTXBAR3			ESC_GPO6		FSIRXC_CLK			
GPIO15	EPWM8B	SCIB_RX	MFSXB		OUTPUTXBAR4			ESC_GPO7		FSIRXD_D0			
GPIO16	SPIA_SIMO	CANB_TX	OUTPUTXBAR7	EPWM9A		SD1_D1			SSIA_TX	FSIRXD_D1			
GPIO17	SPIA_SOMI	CANB_RX	OUTPUTXBAR8	EPWM9B		SD1_C1			SSIA_RX	FSIRXD_CLK			
GPIO18	SPIA_CLK	SCIB_TX	CANA_RX	EPWM10A		SD1_D2	MCAN_RX	EMIF1_CS2n	SSIA_CLK	FSIRXE_D0			
GPIO19	SPIA_STEn	SCIB_RX	CANA_TX	EPWM10B		SD1_C2	MCAN_TX	EMIF1_CS3n	SSIA_FSS	FSIRXE_D1			
GPIO20	EQEP1_A	MDXA	CANB_TX	EPWM11A		SD1_D3		EMIF1_BA0	TRACE_DATA0	FSIRXE_CLK	SPIC_SIMO		
GPIO21	EQEP1_B	MDRA	CANB_RX	EPWM11B		SD1_C3		EMIF1_BA1	TRACE_DATA1	FSIRXF_D0	SPIC_SOMI		
GPIO22	EQEP1_ STROBE	MCLKXA	SCIB_TX	EPWM12A	SPIB_CLK	SD1_D4	MCAN_TX	EMIF1_RAS	TRACE_DATA2	FSIRXF_D1	SPIC_CLK		
GPIO23	EQEP1_INDEX	MFSXA	SCIB_RX	EPWM12B	SPIB_STEn	SD1_C4	MCAN_RX	EMIF1_CAS	TRACE_DATA3	FSIRXF_CLK	SPIC_STEn		
GPIO24	OUTPUTXBAR1	EQEP2_A	MDXB		SPIB_SIMO	SD2_D1	PMBUSA_SCL	EMIF1_DQM0	TRACE_CLK	EPWM13A		FSIRXG_D0	
GPIO25	OUTPUTXBAR2	EQEP2_B	MDRB		SPIB_SOMI	SD2_C1	PMBUSA_SDA	EMIF1_DQM1	TRACE_SWO	EPWM13B	FSITXA_D1	FSIRXG_D1	
GPIO26	OUTPUTXBAR3	EQEP2_INDEX	MCLKXB	OUTPUTXBAR3	SPIB_CLK	SD2_D2	PMBUSA_ALERT	EMIF1_DQM2	ESC_MDIO_ CLK	EPWM14A	FSITXA_D0	FSIRXG_CLK	
GPIO27	OUTPUTXBAR4	EQEP2_ STROBE	MFSXB	OUTPUTXBAR4	SPIB_STEn	SD2_C2	PMBUSA_CTL	EMIF1_DQM3	ESC_MDIO_ DATA	EPWM14B	FSITXA_CLK	FSIRXH_D0	
GPIO28	SCIA_RX	EMIF1_CS4n		OUTPUTXBAR5	EQEP3_A	SD2_D3	EMIF1_CS2n			EPWM15A		FSIRXH_D1	
GPIO29	SCIA_TX	EMIF1_SDCKE		OUTPUTXBAR6	EQEP3_B	SD2_C3	EMIF1_CS3n	ESC_LATCH0	ESC_I2C_SDA	EPWM15B	ESC_SYNC0	FSIRXH_CLK	

Table 15-8. GPIO Muxed Pins (continued)

0, 4, 8, 12	1	2	3	5	6	7	9	10	11	13	14	15	ALT
GPIO30	CANA_RX	EMIF1_CLK	MCAN_RX	OUTPUTXBAR7	EQEP3_STROBE	SD2_D4	EMIF1_CS4n	ESC_LATCH1	ESC_I2C_SCL	EPWM16A	ESC_SYNC1	SPID_SIMO	
GPIO31	CANA_TX	EMIF1_WEn	MCAN_TX	OUTPUTXBAR8	EQEP3_INDEX	SD2_C4	EMIF1_RNW	I2CA_SDA	CM-I2CA_SDA	EPWM16B		SPID_SOMI	
GPIO32	I2CA_SDA	EMIF1_CS0n	SPIA_SIMO			CLB_OUTPUTXBAR1	EMIF1_OEn	I2CA_SCL	CM-I2CA_SCL			SPID_CLK	
GPIO33	I2CA_SCL	EMIF1_RNW	SPIA_SOMI			CLB_OUTPUTXBAR2	EMIF1_BA0					SPID_STEn	
GPIO34	OUTPUTXBAR1	EMIF1_CS2n	SPIA_CLK		I2CB_SDA	CLB_OUTPUTXBAR3	EMIF1_BA1	ESC_LATCH0	ENET_MII_CRS	SCIA_TX	ESC_SYNC0		
GPIO35	SCIA_RX	EMIF1_CS3n	SPIA_STEn		I2CB_SCL	CLB_OUTPUTXBAR4	EMIF1_A0	ESC_LATCH1	ENET_MII_COL		ESC_SYNC1		
GPIO36	SCIA_TX	EMIF1_WAIT			CANA_RX	CLB_OUTPUTXBAR5	EMIF1_A1	MCAN_RX		SD1_D1			
GPIO37	OUTPUTXBAR2	EMIF1_OEn			CANA_TX	CLB_OUTPUTXBAR6	EMIF1_A2	MCAN_TX		SD1_D2			
GPIO38		EMIF1_A0		SCIC_TX	CANB_TX	CLB_OUTPUTXBAR7	EMIF1_A3	ENET_MII_RX_DV	ENET_MII_CRS	SD1_D3			
GPIO39		EMIF1_A1		SCIC_RX	CANB_RX	CLB_OUTPUTXBAR8	EMIF1_A4	ENET_MII_RX_ERR	ENET_MII_COL	SD1_D4			
GPIO40		EMIF1_A2			I2CB_SDA				ENET_MII_CRS		ESC_I2C_SDA		
GPIO41		EMIF1_A3			I2CB_SCL			ENET_REVMII_MDIO_RST	ENET_MII_COL		ESC_I2C_SCL		
GPIO42					I2CA_SDA			ENET_MDIO_CLK	UARTA_TX			SCIA_TX	USB0DM
GPIO43					I2CA_SCL			ENET_MDIO_DATA	UARTA_RX			SCIA_RX	USB0DP
GPIO44		EMIF1_A4							ENET_MII_TX_CLK		ESC_TX1_CLK		
GPIO45		EMIF1_A5							ENET_MII_TX_EN		ESC_TX1_ENA		
GPIO46		EMIF1_A6			SCID_RX				ENET_MII_TX_ERR		ESC_MDIO_CLK		
GPIO47		EMIF1_A7			SCID_TX				ENET_PPS0		ESC_MDIO_DATA		
GPIO48	OUTPUTXBAR3	EMIF1_A8			SCIA_TX	SD1_D1			ENET_PPS1		ESC_PHY_CLK		
GPIO49	OUTPUTXBAR4	EMIF1_A9			SCIA_RX	SD1_C1	EMIF1_A5		ENET_MII_RX_CLK	SD2_D1	FSITXA_D0		
GPIO50	EQEP1_A	EMIF1_A10			SPIC_SIMO	SD1_D2	EMIF1_A6		ENET_MII_RX_DV	SD2_D2	FSITXA_D1		
GPIO51	EQEP1_B	EMIF1_A11			SPIC_SOMI	SD1_C2	EMIF1_A7		ENET_MII_RX_ERR	SD2_D3	FSITXA_CLK		
GPIO52	EQEP1_STROBE	EMIF1_A12			SPIC_CLK	SD1_D3	EMIF1_A8		ENET_MII_RX_DATA0	SD2_D4	FSIRXA_D0		
GPIO53	EQEP1_INDEX	EMIF1_D31	EMIF2_D15		SPIC_STEn	SD1_C3	EMIF1_A9		ENET_MII_RX_DATA1	SD1_C1	FSIRXA_D1		
GPIO54	SPIA_SIMO	EMIF1_D30	EMIF2_D14	EQEP2_A	SCIB_TX	SD1_D4	EMIF1_A10		ENET_MII_RX_DATA2	SD1_C2	FSIRXA_CLK	SSIA_TX	

Table 15-8. GPIO Muxed Pins (continued)

0, 4, 8, 12	1	2	3	5	6	7	9	10	11	13	14	15	ALT
GPIO55	SPIA_SOMI	EMIF1_D29	EMIF2_D13	EQEP2_B	SCIB_RX	SD1_C4	EMIF1_D0		ENET_MII_RX_DATA3	SD1_C3	FSITXB_D0	SSIA_RX	
GPIO56	SPIA_CLK	EMIF1_D28	EMIF2_D12	EQEP2_STROBE	SCIC_TX	SD2_D1	EMIF1_D1	I2CA_SDA	ENET_MII_TX_EN	SD1_C4	FSITXB_CLK	SSIA_CLK	
GPIO57	SPIA_STEn	EMIF1_D27	EMIF2_D11	EQEP2_INDEX	SCIC_RX	SD2_C1	EMIF1_D2	I2CA_SCL	ENET_MII_TX_ERR		FSITXB_D1	SSIA_FSS	
GPIO58	MCLKRA	EMIF1_D26	EMIF2_D10	OUTPUTXBAR1	SPIB_CLK	SD2_D2	EMIF1_D3	ESC_LED_LINK0_ACTIVE	ENET_MII_TX_CLK	SD2_C2	FSIRXB_D0	SPIA_SIMO	
GPIO59	MFSRA	EMIF1_D25	EMIF2_D9	OUTPUTXBAR2	SPIB_STEn	SD2_C2	EMIF1_D4	ESC_LED_LINK1_ACTIVE	ENET_MII_TX_DATA0	SD2_C3	FSIRXB_D1	SPIA_SOMI	
GPIO60	MCLKRB	EMIF1_D24	EMIF2_D8	OUTPUTXBAR3	SPIB_SIMO	SD2_D3	EMIF1_D5	ESC_LED_ERR	ENET_MII_TX_DATA1	SD2_C4	FSIRXB_CLK	SPIA_CLK	
GPIO61	MFSRB	EMIF1_D23	EMIF2_D7	OUTPUTXBAR4	SPIB_SOMI	SD2_C3	EMIF1_D6	ESC_LED_RUN	ENET_MII_TX_DATA2		CANA_RX	SPIA_STEn	
GPIO62	SCIC_RX	EMIF1_D22	EMIF2_D6	EQEP3_A	CANA_RX	SD2_D4	EMIF1_D7	ESC_LED_STATE_RUN	ENET_MII_TX_DATA3		CANA_TX		
GPIO63	SCIC_TX	EMIF1_D21	EMIF2_D5	EQEP3_B	CANA_TX	SD2_C4	SSIA_TX		ENET_MII_RX_DATA0	SD1_D1	ESC_RX1_DATA0	SPIB_SIMO	
GPIO64		EMIF1_D20	EMIF2_D4	EQEP3_STROBE	SCIA_RX		SSIA_RX	ENET_MII_RX_DV	ENET_MII_RX_DATA1	SD1_C1	ESC_RX1_DATA1	SPIB_SOMI	
GPIO65		EMIF1_D19	EMIF2_D3	EQEP3_INDEX	SCIA_TX		SSIA_CLK	ENET_MII_RX_ERR	ENET_MII_RX_DATA2	SD1_D2	ESC_RX1_DATA2	SPIB_CLK	
GPIO66		EMIF1_D18	EMIF2_D2		I2CB_SDA		SSIA_FSS	ENET_MII_RX_DATA0	ENET_MII_RX_DATA3	SD1_C2	ESC_RX1_DATA3	SPIB_STEn	
GPIO67		EMIF1_D17	EMIF2_D1					ENET_MII_RX_CLK	ENET_REVMII_MDIO_RST	SD1_D3			
GPIO68		EMIF1_D16	EMIF2_D0						ENET_MII_INTR	SD1_C3	ESC_PHY1_LINKSTATUS		
GPIO69		EMIF1_D15			I2CB_SCL			ENET_MII_TX_EN	ENET_MII_RX_CLK	SD1_D4	ESC_RX1_CLK	SPIC_SIMO	
GPIO70		EMIF1_D14		CANA_RX	SCIB_TX		MCAN_RX		ENET_MII_RX_DV	SD1_C4	ESC_RX1_DV	SPIC_SOMI	
GPIO71		EMIF1_D13		CANA_TX	SCIB_RX		MCAN_TX	ENET_MII_RX_DATA0	ENET_MII_RX_ERR		ESC_RX1_ERR	SPIC_CLK	
GPIO72		EMIF1_D12		CANB_TX	SCIC_TX			ENET_MII_RX_DATA1	ENET_MII_TX_DATA3		ESC_TX1_DATA3	SPIC_STEn	
GPIO73		EMIF1_D11	XCLKOUT	CANB_RX	SCIC_RX			ENET_RMII_CLK	ENET_MII_TX_DATA2	SD2_D2	ESC_TX1_DATA2		
GPIO74		EMIF1_D10					MCAN_TX		ENET_MII_TX_DATA1	SD2_C2	ESC_TX1_DATA1		
GPIO75		EMIF1_D9					MCAN_RX		ENET_MII_TX_DATA0	SD2_D3	ESC_TX1_DATA0		
GPIO76		EMIF1_D8			SCID_TX			ENET_MII_RX_ERR		SD2_C3	ESC_PHY_RESETn		
GPIO77		EMIF1_D7			SCID_RX					SD2_D4	ESC_RX0_CLK		
GPIO78		EMIF1_D6			EQEP2_A					SD2_C4	ESC_RX0_DV		
GPIO79		EMIF1_D5			EQEP2_B					SD2_D1	ESC_RX0_ERR		

Table 15-8. GPIO Muxed Pins (continued)

0, 4, 8, 12	1	2	3	5	6	7	9	10	11	13	14	15	ALT
GPIO80		EMIF1_D4			EQEP2_ STROBE					SD2_C1	ESC_RX0_ DATA0		
GPIO81		EMIF1_D3			EQEP2_INDEX						ESC_RX0_ DATA1		
GPIO82		EMIF1_D2									ESC_RX0_ DATA2		
GPIO83		EMIF1_D1									ESC_RX0_ DATA3		
GPIO84				SCIA_TX	MDXB				UARTA_TX		ESC_TX0_ENA	MDXA	
GPIO85		EMIF1_D0		SCIA_RX	MDRB				UARTA_RX		ESC_TX0_CLK	MDRA	
GPIO86		EMIF1_A13	EMIF1_CAS	SCIB_TX	MCLKXB						ESC_PHY0_ LINKSTATUS	MCLKXA	
GPIO87		EMIF1_A14	EMIF1_RAS	SCIB_RX	MFSXB		EMIF1_DQM3				ESC_TX0_ DATA0	MFSXA	
GPIO88		EMIF1_A15	EMIF1_DQM0				EMIF1_DQM1				ESC_TX0_ DATA1		
GPIO89		EMIF1_A16	EMIF1_DQM1		SCIC_TX		EMIF1_CAS				ESC_TX0_ DATA2		
GPIO90		EMIF1_A17	EMIF1_DQM2		SCIC_RX		EMIF1_RAS				ESC_TX0_ DATA3		
GPIO91		EMIF1_A18	EMIF1_DQM3		I2CA_SDA		EMIF1_DQM2	PMBUSA_SCL	SSIA_TX	FSIRXF_D0	CLB_OUTPUT XBAR1	SPID_SIMO	
GPIO92		EMIF1_A19	EMIF1_BA1		I2CA_SCL		EMIF1_DQM0	PMBUSA_SDA	SSIA_RX	FSIRXF_D1	CLB_OUTPUT XBAR2	SPID_SOMI	
GPIO93			EMIF1_BA0		SCID_TX			PMBUSA_ ALERT	SSIA_CLK	FSIRXF_CLK	CLB_OUTPUT XBAR3	SPID_CLK	
GPIO94					SCID_RX		EMIF1_BA1	PMBUSA_CTL	SSIA_FSS	FSIRXG_D0	CLB_OUTPUT XBAR4	SPID_STEn	
GPIO95			EMIF2_A12							FSIRXG_D1	CLB_OUTPUT XBAR5		
GPIO96			EMIF2_DQM1	EQEP1_A						FSIRXG_CLK	CLB_OUTPUT XBAR6		
GPIO97			EMIF2_DQM0	EQEP1_B						FSIRXH_D0	CLB_OUTPUT XBAR7		
GPIO98			EMIF2_A0	EQEP1_ STROBE						FSIRXH_D1	CLB_OUTPUT XBAR8		
GPIO99			EMIF2_A1	EQEP1_INDEX						FSIRXH_CLK			
GPIO100			EMIF2_A2	EQEP2_A	SPIC_SIMO			ESC_GPI0		FSITXA_D0			
GPIO101			EMIF2_A3	EQEP2_B	SPIC_SOMI			ESC_GPI1		FSITXA_D1			
GPIO102			EMIF2_A4	EQEP2_ STROBE	SPIC_CLK			ESC_GPI2		FSITXA_CLK			
GPIO103			EMIF2_A5	EQEP2_INDEX	SPIC_STEn			ESC_GPI3		FSIRXA_D0			
GPIO104	I2CA_SDA		EMIF2_A6	EQEP3_A	SCID_TX			ESC_GPI4	CM-I2CA_SDA	FSIRXA_D1			
GPIO105	I2CA_SCL		EMIF2_A7	EQEP3_B	SCID_RX			ESC_GPI5	CM-I2CA_SCL	FSIRXA_CLK	ENET_MDIO_ CLK		
GPIO106			EMIF2_A8	EQEP3_ STROBE	SCIC_TX			ESC_GPI6		FSITXB_D0	ENET_MDIO_ DATA		

Table 15-8. GPIO Muxed Pins (continued)

0, 4, 8, 12	1	2	3	5	6	7	9	10	11	13	14	15	ALT
GPIO107			EMIF2_A9	EQEP3_INDEX	SCIC_RX			ESC_GPI7		FSITXB_D1	ENET_REVMII_MDIO_RST		
GPIO108			EMIF2_A10					ESC_GPI8		FSITXB_CLK	ENET_MII_INTR		
GPIO109			EMIF2_A11					ESC_GPI9			ENET_MII_CRS		
GPIO110			EMIF2_WAIT					ESC_GPI10		FSIRXB_D0	ENET_MII_COL		
GPIO111			EMIF2_BA0					ESC_GPI11		FSIRXB_D1	ENET_MII_RX_CLK		
GPIO112			EMIF2_BA1					ESC_GPI12		FSIRXB_CLK	ENET_MII_RX_DV		
GPIO113			EMIF2_CAS					ESC_GPI13			ENET_MII_RX_ERR		
GPIO114			EMIF2_RAS					ESC_GPI14			ENET_MII_RX_DATA0		
GPIO115			EMIF2_CS0n	OUTPUTXBAR5				ESC_GPI15		FSIRXC_D0	ENET_MII_RX_DATA1		
GPIO116			EMIF2_CS2n	OUTPUTXBAR6				ESC_GPI16		FSIRXC_D1	ENET_MII_RX_DATA2		
GPIO117			EMIF2_SDCKE					ESC_GPI17		FSIRXC_CLK	ENET_MII_RX_DATA3		
GPIO118			EMIF2_CLK					ESC_GPI18		FSIRXD_D0	ENET_MII_TX_EN		
GPIO119			EMIF2_RNW					ESC_GPI19		FSIRXD_D1	ENET_MII_TX_ERR		
GPIO120			EMIF2_WEn					ESC_GPI20		FSIRXD_CLK	ENET_MII_TX_CLK		
GPIO121			EMIF2_OEn					ESC_GPI21		FSIRXE_D0	ENET_MII_TX_DATA0		
GPIO122			EMIF2_D15		SPIC_SIMO	SD1_D1		ESC_GPI22			ENET_MII_TX_DATA1		
GPIO123			EMIF2_D14		SPIC_SOMI	SD1_C1		ESC_GPI23			ENET_MII_TX_DATA2		
GPIO124			EMIF2_D13		SPIC_CLK	SD1_D2		ESC_GPI24			ENET_MII_TX_DATA3		
GPIO125			EMIF2_D12		SPIC_STEn	SD1_C2		ESC_GPI25		FSIRXE_D1	ESC_LATCH0		
GPIO126			EMIF2_D11			SD1_D3		ESC_GPI26		FSIRXE_CLK	ESC_LATCH1		
GPIO127			EMIF2_D10			SD1_C3		ESC_GPI27			ESC_SYNC0		
GPIO128			EMIF2_D9			SD1_D4		ESC_GPI28			ESC_SYNC1		
GPIO129			EMIF2_D8			SD1_C4		ESC_GPI29			ESC_TX1_ENA		
GPIO130			EMIF2_D7			SD2_D1		ESC_GPI30			ESC_TX1_CLK		
GPIO131			EMIF2_D6			SD2_C1		ESC_GPI31			ESC_TX1_DATA0		
GPIO132			EMIF2_D5			SD2_D2		ESC_GPO0			ESC_TX1_DATA1		
GPIO133						SD2_C2							AUXCLKIN
GPIO134			EMIF2_D4			SD2_D3		ESC_GPO1			ESC_TX1_DATA2		

Table 15-8. GPIO Muxed Pins (continued)

0, 4, 8, 12	1	2	3	5	6	7	9	10	11	13	14	15	ALT
GPIO135			EMIF2_D3		SCIA_TX	SD2_C3		ESC_GPO2			ESC_TX1_DATA3		
GPIO136			EMIF2_D2		SCIA_RX	SD2_D4		ESC_GPO3			ESC_RX1_DV		
GPIO137	EPWM13A		EMIF2_D1		SCIB_TX	SD2_C4		ESC_GPO4			ESC_RX1_CLK		
GPIO138	EPWM13B		EMIF2_D0		SCIB_RX			ESC_GPO5			ESC_RX1_ERR		
GPIO139	EPWM14A				SCIC_RX			ESC_GPO6			ESC_RX1_DATA0		
GPIO140	EPWM14B				SCIC_TX			ESC_GPO7			ESC_RX1_DATA1		
GPIO141	EPWM15A				SCID_RX			ESC_GPO8			ESC_RX1_DATA2		
GPIO142	EPWM15B				SCID_TX			ESC_GPO9			ESC_RX1_DATA3		
GPIO143	EPWM16A							ESC_GPO10			ESC_LED_LINK0_ACTIVE		
GPIO144	EPWM16B							ESC_GPO11			ESC_LED_LINK1_ACTIVE		
GPIO145	EPWM1A							ESC_GPO12			ESC_LED_ERR		
GPIO146	EPWM1B							ESC_GPO13			ESC_LED_RUN		
GPIO147	EPWM2A							ESC_GPO14			ESC_LED_STATE_RUN		
GPIO148	EPWM2B							ESC_GPO15			ESC_PHY0_LINKSTATUS		
GPIO149	EPWM3A							ESC_GPO16			ESC_PHY1_LINKSTATUS		
GPIO150	EPWM3B							ESC_GPO17			ESC_I2C_SDA		
GPIO151	EPWM4A							ESC_GPO18			ESC_I2C_SCL		
GPIO152	EPWM4B							ESC_GPO19			ESC_MDIO_CLK		
GPIO153	EPWM5A							ESC_GPO20			ESC_MDIO_DATA		
GPIO154	EPWM5B							ESC_GPO21			ESC_PHY_CLK		
GPIO155	EPWM6A							ESC_GPO22			ESC_PHY_RESETn		
GPIO156	EPWM6B							ESC_GPO23			ESC_TX0_ENA		
GPIO157	EPWM7A							ESC_GPO24			ESC_TX0_CLK		
GPIO158	EPWM7B							ESC_GPO25			ESC_TX0_DATA0		
GPIO159	EPWM8A							ESC_GPO26			ESC_TX0_DATA1		
GPIO160	EPWM8B							ESC_GPO27			ESC_TX0_DATA2		
GPIO161	EPWM9A							ESC_GPO28			ESC_TX0_DATA3		
GPIO162	EPWM9B							ESC_GPO29			ESC_RX0_DV		
GPIO163	EPWM10A							ESC_GPO30			ESC_RX0_CLK		

Table 15-8. GPIO Muxed Pins (continued)

0, 4, 8, 12	1	2	3	5	6	7	9	10	11	13	14	15	ALT
GPIO164	EPWM10B							ESC_GPO31			ESC_RX0_ERR		
GPIO165	EPWM11A							MDXA			ESC_RX0_ DATA0		
GPIO166	EPWM11B							MDRA			ESC_RX0_ DATA1		
GPIO167	EPWM12A							MCLKXA			ESC_RX0_ DATA2		
GPIO168	EPWM12B							MFSXA			ESC_RX0_ DATA3		

15.7.2 Peripheral Muxing

For example, multiplexing for the GPIO6 pin is controlled by writing to GPAGMUX[13:12] and GPAMUX[13:12]. By writing to these bits, GPIO6 is configured as either a general-purpose digital I/O or one of several different peripheral functions. An example of GPyGMUX and GPyMUX selection and options for a single GPIO are shown in Table 15-9.

Note

The following table is for example only. GPIO6 may not be available on this device. If GPIO6 is available, the functions mentioned in the table may not match the actual functions available. See Section 15.7.1 for correct list of GPIOs and corresponding mux options for this device.

Table 15-9. GPIO and Peripheral Muxing

GPAGMUX1[13:12]	GPAMUX1[13:12]	Pin Functionality
00	00	GPIO6
00	01	Peripheral 1
00	10	Peripheral 2
00	11	Peripheral 3
01	00	GPIO6
01	01	Peripheral 4
01	10	Peripheral 5
01	11	
10	00	GPIO6
10	01	
10	10	Peripheral 6
10	11	Peripheral 7
11	00	GPIO6
11	01	Peripheral 8
11	10	Peripheral 9
11	11	Peripheral 10

The devices have different multiplexing schemes. If a peripheral is not available on a particular device, that mux selection is reserved on that device and must not be used.

CAUTION

If a reserved GPIO mux configuration that is not mapped to either a peripheral or GPIO mode is selected, the state of the pin is undefined and the pin is driven. Unimplemented configurations are for future expansion and must not be selected. In the device mux table (see the data sheet), these options are indicated as Reserved or left blank.

Some peripherals can be assigned to more than one pin by way of the mux registers. For example, OUTPUTXBAR1 can be assigned to GPIOs p, q, or r (where p, q, and r are example GPIO numbers), depending on individual system requirements. An example of this is shown in Table 15-10.

Note

The following table is for example only. Bit ranges cannot correspond to OUTPUTXBAR1 on this device. See Section 15.7.1 for correct list of GPIOs and corresponding mux options for this device.

If none or more then one of the GPIO pins is configured as peripheral input pins, then that GPIO is set to a hard-wired default value.

Table 15-10. Peripheral Muxing (Multiple Pins Assigned)

GMUX Configuration	MUX Configuration	
Choice 1: GPIOp	GPyGMUX1[5:4]=01	GPyMUX1[5:4]=01
or Choice 2: GPIOq	GPyGMUX2[17:16]=00	GPyMUX2[17:16]=01
or Choice 3: GPIOr	GPyGMUX1[7:6]=01	GPyMUX1[7:6]=01

15.8 Internal Pullup Configuration Requirements

On reset, GPIOs are in input mode and have the internal pullups disabled. An un-driven input can float to a mid-rail voltage and cause wasted shoot-through current on the input buffer. The user must always put each GPIO in one of these configurations:

- Input mode and driven on the board by another component to a level above V_{ih} or below V_{il}
- Input mode with GPIO internal pullup enabled
- Output mode

On devices with lesser pin count packages, pull-ups on unbonded GPIOs are by default enabled to prevent floating inputs. The user must take care to avoid disabling these pullups in their application code.

On devices with larger pin count packages, the pullups for any internally unbonded GPIO must be enabled to prevent floating inputs. TI has provided functions in controlSUITE/C2000Ware that users can call to enable the pullup on any unbonded GPIO for the package they are using. This function, GPIO_EnabledUnbondedIOPullups(), resides in the (Device)_Sysctrl.c file and is called by default from InitSysCtrl(). The user must take care to avoid disabling these pullups in their application code.

15.9 Software

15.9.1 GPIO Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/gpio

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

15.9.1.1 Device GPIO Setup

FILE: gpio_ex1_setup.c

Configures the device GPIO into two different configurations This code is verbose to illustrate how the GPIO could be setup. In a real application, lines of code can be combined for improved code size and efficiency.

This example only sets-up the GPIO. Nothing is actually done with the pins after setup.

In general:

- All pullup resistors are enabled. For ePWMs this may not be desired.
- Input qual for communication ports (CAN, SPI, SCI, I2C) is asynchronous
- Input qual for Trip pins (TZ) is asynchronous
- Input qual for eCAP and eQEP signals is synch to SYSCLKOUT
- Input qual for some I/O's and __interrupts may have a sampling window

15.9.1.2 Device GPIO Toggle

FILE: gpio_ex2_toggle.c

Configures the device GPIO through the sysconfig file. The GPIO pin is toggled in the infinite loop.

15.9.1.3 Device GPIO Interrupt

FILE: gpio_ex3_interrupt.c

Configures the device GPIOs through the sysconfig file. One GPIO output pin, and one GPIO input pin is configured. The example then configures the GPIO input pin to be the source of an external interrupt which toggles the GPIO output pin.

15.9.2 LED Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/led

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

15.9.2.1 LED Blinky Example (CM) - C28X_CM

FILE: led_ex1_c28x_cm_blinky_cm.c

This example demonstrates how to blink a LED using CM.

External Connections

- None.

Watch Variables

- None.

15.9.2.2 LED Blinky Example - C28X_DUAL

FILE: led_ex1_c28x_dual_blinky_cpu1.c

This example demonstrates how to blink a LED using CPU1 and blink another LED using CPU2 (led_ex1_blinky_cpu2.c).

External Connections

- None.

Watch Variables

- None.

15.9.2.3 LED Blinky Example - C28X_CM

FILE: led_ex1_c28x_cm_blinky_cpu1.c

This example demonstrates how to blink a LED using CPU1 and blink another LED using CM (led_ex1_blinky_cm.c).

External Connections

- None.

Watch Variables

- None.

15.9.2.4 LED Blinky Example with DCSM

FILE: led_ex2_blinky_dcsm.c

This example demonstrates how to blink a LED and program the DCSM OTP. Please refer f2838x_dcsm_z1otp.asm and f2838x_dcsm_z1otp.asm files for details on DCSM OTP programming.

External Connections

- None.

Watch Variables

- None.

15.9.2.5 LED Blinky Example - C28X_DUAL

FILE: led_ex2_blinky_sysconfig_cpu1.c

This example demonstrates how to blink a LED using CPU1 and blink another LED using CPU2 (led_ex1_blinky_cpu2.c).

External Connections

- None.

Watch Variables

- None.

15.10 GPIO Registers

This section describes the General-Purpose Input/Output Registers.

15.10.1 GPIO Base Address Table (C28)

Table 15-11. GPIO Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
GpioCtrlRegs	GPIO_CTRL_REGS	GPIOCTRL_BASE	0x0000_7C00	YES	-	-	-	YES
GpioDataRegs	GPIO_DATA_REGS	GPIODATA_BASE	0x0000_7F00	YES	YES	-	YES	YES
GpioDataReadRegs	GPIO_DATA_READ_REGS	GPIODATAREAD_BASE	0x0000_7F80	YES	YES	-	YES	YES

15.10.2 CM GPIO Base Address Table (CM)

Table 15-12. CM GPIO Base Address Table (CM)

DriverLib Name	Base Address	µDMA Access	Ethernet DMA Access
GPIODATA_BASE	0x4008_3000	-	-

Table 15-12. CM GPIO Base Address Table (CM) (continued)

DriverLib Name	Base Address	µDMA Access	Ethernet DMA Access
GPIODATAREAD_BASE	0x4008_3100	-	-

15.10.3 GPIO_CTRL_REGS Registers

Table 15-13 lists the memory-mapped registers for the GPIO_CTRL_REGS registers. All register offset addresses not listed in Table 15-13 should be considered as reserved locations and the register contents should not be modified.

Table 15-13. GPIO_CTRL_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	GPACTRL	GPIO A Qualification Sampling Period Control (GPIO0 to 31)	EALLOW	Go
2h	GPAQSEL1	GPIO A Qualifier Select 1 Register (GPIO0 to 15)	EALLOW	Go
4h	GPAQSEL2	GPIO A Qualifier Select 2 Register (GPIO16 to 31)	EALLOW	Go
6h	GPAMUX1	GPIO A Mux 1 Register (GPIO0 to 15)	EALLOW	Go
8h	GPAMUX2	GPIO A Mux 2 Register (GPIO16 to 31)	EALLOW	Go
Ah	GPADIR	GPIO A Direction Register (GPIO0 to 31)	EALLOW	Go
Ch	GPAPUD	GPIO A Pull Up Disable Register (GPIO0 to 31)	EALLOW	Go
10h	GPAINV	GPIO A Input Polarity Invert Registers (GPIO0 to 31)	EALLOW	Go
12h	GPAODR	GPIO A Open Drain Output Register (GPIO0 to GPIO31)	EALLOW	Go
20h	GPAGMUX1	GPIO A Peripheral Group Mux (GPIO0 to 15)	EALLOW	Go
22h	GPAGMUX2	GPIO A Peripheral Group Mux (GPIO16 to 31)	EALLOW	Go
28h	GPACSEL1	GPIO A Core Select Register (GPIO0 to 7)	EALLOW	Go
2Ah	GPACSEL2	GPIO A Core Select Register (GPIO8 to 15)	EALLOW	Go
2Ch	GPACSEL3	GPIO A Core Select Register (GPIO16 to 23)	EALLOW	Go
2Eh	GPACSEL4	GPIO A Core Select Register (GPIO24 to 31)	EALLOW	Go
3Ch	GPALOCK	GPIO A Lock Configuration Register (GPIO0 to 31)	EALLOW	Go
3Eh	GPACR	GPIO A Lock Commit Register (GPIO0 to 31)	EALLOW	Go
40h	GPBCTRL	GPIO B Qualification Sampling Period Control (GPIO32 to 63)	EALLOW	Go
42h	GPBQSEL1	GPIO B Qualifier Select 1 Register (GPIO32 to 47)	EALLOW	Go
44h	GPBQSEL2	GPIO B Qualifier Select 2 Register (GPIO48 to 63)	EALLOW	Go
46h	GPBMUX1	GPIO B Mux 1 Register (GPIO32 to 47)	EALLOW	Go
48h	GPBMUX2	GPIO B Mux 2 Register (GPIO48 to 63)	EALLOW	Go
4Ah	GPBDIR	GPIO B Direction Register (GPIO32 to 63)	EALLOW	Go
4Ch	GPBPUD	GPIO B Pull Up Disable Register (GPIO32 to 63)	EALLOW	Go
50h	GPBINV	GPIO B Input Polarity Invert Registers (GPIO32 to 63)	EALLOW	Go
52h	GPBODR	GPIO B Open Drain Output Register (GPIO32 to GPIO63)	EALLOW	Go
54h	GPBAMSEL	GPIO B Analog Mode Select register (GPIO32 to GPIO63)	EALLOW	Go
60h	GPBGMUX1	GPIO B Peripheral Group Mux (GPIO32 to 47)	EALLOW	Go
62h	GPBGMUX2	GPIO B Peripheral Group Mux (GPIO48 to 63)	EALLOW	Go
68h	GPBCSEL1	GPIO B Core Select Register (GPIO32 to 39)	EALLOW	Go
6Ah	GPBCSEL2	GPIO B Core Select Register (GPIO40 to 47)	EALLOW	Go
6Ch	GPBCSEL3	GPIO B Core Select Register (GPIO48 to 55)	EALLOW	Go
6Eh	GPBCSEL4	GPIO B Core Select Register (GPIO56 to 63)	EALLOW	Go

Table 15-13. GPIO_CTRL_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
7Ch	GPBLOCK	GPIO B Lock Configuration Register (GPIO32 to 63)	EALLOW	Go
7Eh	GPBCR	GPIO B Lock Commit Register (GPIO32 to 63)	EALLOW	Go
80h	GPCCTRL	GPIO C Qualification Sampling Period Control (GPIO64 to 95)	EALLOW	Go
82h	GPCQSEL1	GPIO C Qualifier Select 1 Register (GPIO64 to 79)	EALLOW	Go
84h	GPCQSEL2	GPIO C Qualifier Select 2 Register (GPIO80 to 95)	EALLOW	Go
86h	GPCMUX1	GPIO C Mux 1 Register (GPIO64 to 79)	EALLOW	Go
88h	GPCMUX2	GPIO C Mux 2 Register (GPIO80 to 95)	EALLOW	Go
8Ah	GPCDIR	GPIO C Direction Register (GPIO64 to 95)	EALLOW	Go
8Ch	GPCPUD	GPIO C Pull Up Disable Register (GPIO64 to 95)	EALLOW	Go
90h	GPCINV	GPIO C Input Polarity Invert Registers (GPIO64 to 95)	EALLOW	Go
92h	GPCODR	GPIO C Open Drain Output Register (GPIO64 to GPIO95)	EALLOW	Go
A0h	GPCGMUX1	GPIO C Peripheral Group Mux (GPIO64 to 79)	EALLOW	Go
A2h	GPCGMUX2	GPIO C Peripheral Group Mux (GPIO80 to 95)	EALLOW	Go
A8h	GPCCSEL1	GPIO C Core Select Register (GPIO64 to 71)	EALLOW	Go
AAh	GPCCSEL2	GPIO C Core Select Register (GPIO72 to 79)	EALLOW	Go
ACh	GPCCSEL3	GPIO C Core Select Register (GPIO80 to 87)	EALLOW	Go
AEh	GPCCSEL4	GPIO C Core Select Register (GPIO88 to 95)	EALLOW	Go
BCh	GPCLOCK	GPIO C Lock Configuration Register (GPIO64 to 95)	EALLOW	Go
BEh	GPCCR	GPIO C Lock Commit Register (GPIO64 to 95)	EALLOW	Go
C0h	GPDCTRL	GPIO D Qualification Sampling Period Control (GPIO96 to 127)	EALLOW	Go
C2h	GPDQSEL1	GPIO D Qualifier Select 1 Register (GPIO96 to 111)	EALLOW	Go
C4h	GPDQSEL2	GPIO D Qualifier Select 2 Register (GPIO112 to 127)	EALLOW	Go
C6h	GPDMUX1	GPIO D Mux 1 Register (GPIO96 to 111)	EALLOW	Go
C8h	GPDMUX2	GPIO D Mux 2 Register (GPIO112 to 127)	EALLOW	Go
CAh	GPDDIR	GPIO D Direction Register (GPIO96 to 127)	EALLOW	Go
CCh	GPDPUD	GPIO D Pull Up Disable Register (GPIO96 to 127)	EALLOW	Go
D0h	GPDINV	GPIO D Input Polarity Invert Registers (GPIO96 to 127)	EALLOW	Go
D2h	GPDODR	GPIO D Open Drain Output Register (GPIO96 to GPIO127)	EALLOW	Go
E0h	GPDGMUX1	GPIO D Peripheral Group Mux (GPIO96 to 111)	EALLOW	Go
E2h	GPDGMUX2	GPIO D Peripheral Group Mux (GPIO112 to 127)	EALLOW	Go
E8h	GPDCSEL1	GPIO D Core Select Register (GPIO96 to 103)	EALLOW	Go
EAh	GPDCSEL2	GPIO D Core Select Register (GPIO104 to 111)	EALLOW	Go
ECh	GPDCSEL3	GPIO D Core Select Register (GPIO112 to 119)	EALLOW	Go
EEh	GPDCSEL4	GPIO D Core Select Register (GPIO120 to 127)	EALLOW	Go
FCh	GPDLOCK	GPIO D Lock Configuration Register (GPIO96 to 127)	EALLOW	Go
FEh	GPDCR	GPIO D Lock Commit Register (GPIO96 to 127)	EALLOW	Go

Table 15-13. GPIO_CTRL_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
100h	GPECTRL	GPIO E Qualification Sampling Period Control (GPIO128 to 159)	EALLOW	Go
102h	GPEQSEL1	GPIO E Qualifier Select 1 Register (GPIO128 to 143)	EALLOW	Go
104h	GPEQSEL2	GPIO E Qualifier Select 2 Register (GPIO144 to 159)	EALLOW	Go
106h	GPEMUX1	GPIO E Mux 1 Register (GPIO128 to 143)	EALLOW	Go
108h	GPEMUX2	GPIO E Mux 2 Register (GPIO144 to 159)	EALLOW	Go
10Ah	GPEDIR	GPIO E Direction Register (GPIO128 to 159)	EALLOW	Go
10Ch	GPEPUD	GPIO E Pull Up Disable Register (GPIO128 to 159)	EALLOW	Go
110h	GPEINV	GPIO E Input Polarity Invert Registers (GPIO128 to 159)	EALLOW	Go
112h	GPEODR	GPIO E Open Drain Output Register (GPIO128 to GPIO159)	EALLOW	Go
120h	GPEGMUX1	GPIO E Peripheral Group Mux (GPIO128 to 143)	EALLOW	Go
122h	GPEGMUX2	GPIO E Peripheral Group Mux (GPIO144 to 159)	EALLOW	Go
128h	GPECSEL1	GPIO E Core Select Register (GPIO128 to 135)	EALLOW	Go
12Ah	GPECSEL2	GPIO E Core Select Register (GPIO136 to 143)	EALLOW	Go
12Ch	GPECSEL3	GPIO E Core Select Register (GPIO144 to 151)	EALLOW	Go
12Eh	GPECSEL4	GPIO E Core Select Register (GPIO152 to 159)	EALLOW	Go
13Ch	GPELOCK	GPIO E Lock Configuration Register (GPIO128 to 159)	EALLOW	Go
13Eh	GPECR	GPIO E Lock Commit Register (GPIO128 to 159)	EALLOW	Go
140h	GPFCTRL	GPIO F Qualification Sampling Period Control (GPIO160 to 168)	EALLOW	Go
142h	GPFQSEL1	GPIO F Qualifier Select 1 Register (GPIO160 to 168)	EALLOW	Go
146h	GPFMUX1	GPIO F Mux 1 Register (GPIO160 to 168)	EALLOW	Go
14Ah	GPFDIR	GPIO F Direction Register (GPIO160 to 168)	EALLOW	Go
14Ch	GPFPUD	GPIO F Pull Up Disable Register (GPIO160 to 168)	EALLOW	Go
150h	GPFINV	GPIO F Input Polarity Invert Registers (GPIO160 to 168)	EALLOW	Go
152h	GPFODR	GPIO F Open Drain Output Register (GPIO160 to GPIO168)	EALLOW	Go
160h	GPFGMUX1	GPIO F Peripheral Group Mux (GPIO160 to 168)	EALLOW	Go
168h	GPFCSEL1	GPIO F Core Select Register (GPIO160 to 167)	EALLOW	Go
16Ah	GPFCSEL2	GPIO F Core Select Register (GPIO168)	EALLOW	Go
17Ch	GPFLOCK	GPIO F Lock Configuration Register (GPIO160 to 168)	EALLOW	Go
17Eh	GPFCR	GPIO F Lock Commit Register (GPIO160 to 168)	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 15-14 shows the codes that are used for access types in this section.

Table 15-14. GPIO_CTRL_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read

Table 15-14. GPIO_CTRL_REGS Access Type Codes (continued)

Access Type	Code	Description
Write Type		
W	W	Write
WOnce	W Once	Write Write once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

15.10.3.1 GPACTRL Register (Offset = 0h) [Reset = 0h]

GPACTRL is shown in Figure 15-4 and described in Table 15-15.

Return to the Summary Table.

GPIO A Qualification Sampling Period Control (GPIO0 to 31)

Figure 15-4. GPACTRL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUALPRD3								QUALPRD2								QUALPRD1								QUALPRD0							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 15-15. GPACTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	QUALPRD3	R/W	0h	Qualification sampling period for GPIO24 to GPIO31: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn
23-16	QUALPRD2	R/W	0h	Qualification sampling period for GPIO16 to GPIO23: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn
15-8	QUALPRD1	R/W	0h	Qualification sampling period for GPIO8 to GPIO15: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn
7-0	QUALPRD0	R/W	0h	Qualification sampling period for GPIO0 to GPIO7: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn

15.10.3.2 GPAQSEL1 Register (Offset = 2h) [Reset = 0h]

GPAQSEL1 is shown in Figure 15-5 and described in Table 15-16.

Return to the Summary Table.

GPIO A Qualifier Select 1 Register (GPIO0 to 15)
Input qualification type:
0,0 Sync
0,1 Qualification (3 samples)
1,0 Qualification (6 samples)
1,1 Async (no Sync or Qualification)

Figure 15-5. GPAQSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO15		GPIO14		GPIO13		GPIO12		GPIO11		GPIO10		GPIO9		GPIO8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO7		GPIO6		GPIO5		GPIO4		GPIO3		GPIO2		GPIO1		GPIO0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-16. GPAQSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO15	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
29-28	GPIO14	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
27-26	GPIO13	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
25-24	GPIO12	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
23-22	GPIO11	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
21-20	GPIO10	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
19-18	GPIO9	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
17-16	GPIO8	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
15-14	GPIO7	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
13-12	GPIO6	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
11-10	GPIO5	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
9-8	GPIO4	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
7-6	GPIO3	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
5-4	GPIO2	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
3-2	GPIO1	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
1-0	GPIO0	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn

15.10.3.3 GPAQSEL2 Register (Offset = 4h) [Reset = 0h]

GPAQSEL2 is shown in Figure 15-6 and described in Table 15-17.

Return to the Summary Table.

GPIO A Qualifier Select 2 Register (GPIO16 to 31)
Input qualification type:
0,0 Sync
0,1 Qualification (3 samples)
1,0 Qualification (6 samples)
1,1 Async (no Sync or Qualification)

Figure 15-6. GPAQSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO31		GPIO30		GPIO29		GPIO28		GPIO27		GPIO26		GPIO25		GPIO24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO23		GPIO22		GPIO21		GPIO20		GPIO19		GPIO18		GPIO17		GPIO16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-17. GPAQSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO31	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
29-28	GPIO30	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
27-26	GPIO29	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
25-24	GPIO28	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
23-22	GPIO27	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
21-20	GPIO26	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
19-18	GPIO25	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
17-16	GPIO24	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
15-14	GPIO23	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
13-12	GPIO22	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
11-10	GPIO21	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
9-8	GPIO20	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
7-6	GPIO19	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
5-4	GPIO18	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
3-2	GPIO17	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
1-0	GPIO16	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn

15.10.3.4 GPAMUX1 Register (Offset = 6h) [Reset = 0h]

GPAMUX1 is shown in Figure 15-7 and described in Table 15-18.

Return to the Summary Table.

GPIO A Mux 1 Register (GPIO0 to 15)
Defines pin-muxing selection for GPIO.
Notes:
The respective GPyGMUXn.GPIOz must be configured prior to this register to avoid intermediate peripheral selects being mapped to the GPIO.

Figure 15-7. GPAMUX1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO15		GPIO14		GPIO13		GPIO12		GPIO11		GPIO10		GPIO9		GPIO8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO7		GPIO6		GPIO5		GPIO4		GPIO3		GPIO2		GPIO1		GPIO0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-18. GPAMUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO15	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO14	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO13	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO12	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO11	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO10	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO9	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO8	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO7	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO6	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO5	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO4	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO3	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
5-4	GPIO2	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
3-2	GPIO1	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO0	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.5 GPAMUX2 Register (Offset = 8h) [Reset = 0h]

GPAMUX2 is shown in Figure 15-8 and described in Table 15-19.

Return to the Summary Table.

GPIO A Mux 2 Register (GPIO16 to 31)
Defines pin-muxing selection for GPIO.
Notes:
The respective GPyGMUXn.GPIOz must be configured prior to this register to avoid intermediate peripheral selects being mapped to the GPIO.

Figure 15-8. GPAMUX2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO31		GPIO30		GPIO29		GPIO28		GPIO27		GPIO26		GPIO25		GPIO24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO23		GPIO22		GPIO21		GPIO20		GPIO19		GPIO18		GPIO17		GPIO16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-19. GPAMUX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO31	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO30	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO29	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO28	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO27	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO26	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO25	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO24	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO23	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO22	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO21	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO20	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO19	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
5-4	GPIO18	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
3-2	GPIO17	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO16	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.6 GPADIR Register (Offset = Ah) [Reset = 0h]

GPADIR is shown in Figure 15-9 and described in Table 15-20.

Return to the Summary Table.

GPIO A Direction Register (GPIO0 to 31)
Controls direction of GPIO pins when the specified pin is configured in GPIO mode.
0: Configures pin as input.
1: Configures pin as output.
Reading the register returns the current value of the register setting.

Figure 15-9. GPADIR Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-20. GPADIR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
30	GPIO30	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
29	GPIO29	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
28	GPIO28	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
27	GPIO27	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
26	GPIO26	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
25	GPIO25	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
24	GPIO24	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
23	GPIO23	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
22	GPIO22	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
21	GPIO21	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
20	GPIO20	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
19	GPIO19	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn

Table 15-20. GPADIR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO18	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
17	GPIO17	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
16	GPIO16	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
15	GPIO15	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
14	GPIO14	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
13	GPIO13	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
12	GPIO12	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
11	GPIO11	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
10	GPIO10	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
9	GPIO9	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
8	GPIO8	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
7	GPIO7	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
6	GPIO6	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
5	GPIO5	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
4	GPIO4	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
3	GPIO3	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
2	GPIO2	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
1	GPIO1	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
0	GPIO0	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn

15.10.3.7 GPAPUD Register (Offset = Ch) [Reset = FFFFFFFFh]

GPAPUD is shown in Figure 15-10 and described in Table 15-21.

Return to the Summary Table.

GPIO A Pull Up Disable Register (GPIO0 to 31)
Disables the Pull-Up on GPIO.
0: Enables the Pull-Up.
1: Disables the Pull-Up.
Reading the register returns the current value of the register setting.

Figure 15-10. GPAPUD Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 15-21. GPAPUD Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
30	GPIO30	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
29	GPIO29	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
28	GPIO28	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
27	GPIO27	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
26	GPIO26	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
25	GPIO25	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
24	GPIO24	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
23	GPIO23	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
22	GPIO22	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
21	GPIO21	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
20	GPIO20	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
19	GPIO19	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn

Table 15-21. GPAPUD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO18	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
17	GPIO17	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
16	GPIO16	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
15	GPIO15	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
14	GPIO14	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
13	GPIO13	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
12	GPIO12	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
11	GPIO11	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
10	GPIO10	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
9	GPIO9	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
8	GPIO8	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
7	GPIO7	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
6	GPIO6	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
5	GPIO5	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
4	GPIO4	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
3	GPIO3	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
2	GPIO2	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
1	GPIO1	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
0	GPIO0	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn

15.10.3.8 GPAINV Register (Offset = 10h) [Reset = 0h]

GPAINV is shown in Figure 15-11 and described in Table 15-22.

Return to the Summary Table.

GPIO A Input Polarity Invert Registers (GPIO0 to 31)
Selects between non-inverted and inverted GPIO input to the device.
0: selects non-inverted GPIO input
1: selects inverted GPIO input
Reading the register returns the current value of the register setting.

Figure 15-11. GPAINV Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-22. GPAINV Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
30	GPIO30	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
29	GPIO29	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
28	GPIO28	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
27	GPIO27	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
26	GPIO26	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
25	GPIO25	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
24	GPIO24	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
23	GPIO23	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
22	GPIO22	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
21	GPIO21	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
20	GPIO20	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
19	GPIO19	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn

Table 15-22. GPAINV Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO18	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
17	GPIO17	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
16	GPIO16	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
15	GPIO15	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
14	GPIO14	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
13	GPIO13	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
12	GPIO12	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
11	GPIO11	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
10	GPIO10	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
9	GPIO9	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
8	GPIO8	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
7	GPIO7	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
6	GPIO6	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
5	GPIO5	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
4	GPIO4	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
3	GPIO3	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
2	GPIO2	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
1	GPIO1	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
0	GPIO0	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn

15.10.3.9 GPAODR Register (Offset = 12h) [Reset = 0h]

GPAODR is shown in Figure 15-12 and described in Table 15-23.

Return to the Summary Table.

GPIO Open Drain Output Register
Selects between normal and open-drain output for the GPIO pin.
0: Normal Output
1: Open Drain Output
CAUTION: Open Drain mode is not supported on this device. Keep this bit as 0. Open drain can be emulated by writing 0 to GPxDAT and enabling or disabling the output buffer with GPxDIR. (Peripherals such as I2C support Open Drain mode without issue).

Figure 15-12. GPAODR Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-23. GPAODR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
30	GPIO30	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
29	GPIO29	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
28	GPIO28	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
27	GPIO27	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
26	GPIO26	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
25	GPIO25	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
24	GPIO24	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
23	GPIO23	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
22	GPIO22	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
21	GPIO21	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
20	GPIO20	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn

Table 15-23. GPAODR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	GPIO19	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
18	GPIO18	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
17	GPIO17	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
16	GPIO16	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
15	GPIO15	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
14	GPIO14	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
13	GPIO13	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
12	GPIO12	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
11	GPIO11	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
10	GPIO10	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
9	GPIO9	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
8	GPIO8	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
7	GPIO7	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
6	GPIO6	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
5	GPIO5	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
4	GPIO4	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
3	GPIO3	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
2	GPIO2	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
1	GPIO1	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
0	GPIO0	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn

15.10.3.10 GPAGMUX1 Register (Offset = 20h) [Reset = 0h]

GPAGMUX1 is shown in Figure 15-13 and described in Table 15-24.

Return to the Summary Table.

GPIO A Peripheral Group Mux (GPIO0 to 15)
Defines pin-muxing selection for GPIO.
Note: For complete pin-mux selection on GPIOx, GPAMUXy.GPIOx configuration is also required.

Figure 15-13. GPAGMUX1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO15		GPIO14		GPIO13		GPIO12		GPIO11		GPIO10		GPIO9		GPIO8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO7		GPIO6		GPIO5		GPIO4		GPIO3		GPIO2		GPIO1		GPIO0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-24. GPAGMUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO15	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO14	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO13	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO12	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO11	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO10	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO9	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO8	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO7	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO6	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO5	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO4	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO3	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
5-4	GPIO2	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
3-2	GPIO1	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO0	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.11 GPAGMUX2 Register (Offset = 22h) [Reset = 0h]

GPAGMUX2 is shown in Figure 15-14 and described in Table 15-25.

Return to the Summary Table.

GPIO A Peripheral Group Mux (GPIO16 to 31)
Defines pin-muxing selection for GPIO.
Note: For complete pin-mux selection on GPIOx, GPAMUXy.GPIOx configuration is also required.

Figure 15-14. GPAGMUX2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO31		GPIO30		GPIO29		GPIO28		GPIO27		GPIO26		GPIO25		GPIO24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO23		GPIO22		GPIO21		GPIO20		GPIO19		GPIO18		GPIO17		GPIO16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-25. GPAGMUX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO31	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO30	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO29	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO28	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO27	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO26	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO25	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO24	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO23	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO22	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO21	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO20	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO19	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
5-4	GPIO18	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
3-2	GPIO17	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO16	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.12 GPACSEL1 Register (Offset = 28h) [Reset = 0h]

GPACSEL1 is shown in Figure 15-15 and described in Table 15-26.

Return to the Summary Table.

GPIO A Core Select Register (GPIO0 to 7)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPACSEL[0] is used. Writing to GPACSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-15. GPACSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO7				GPIO6				GPIO5				GPIO4			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO3				GPIO2				GPIO1				GPIO0			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-26. GPACSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO7	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO6	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO5	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO4	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO3	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO2	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO1	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO0	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.13 GPACSEL2 Register (Offset = 2Ah) [Reset = 0h]

GPACSEL2 is shown in Figure 15-16 and described in Table 15-27.

Return to the Summary Table.

GPIO A Core Select Register (GPIO8 to 15)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPACSEL[0] is used. Writing to GPACSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-16. GPACSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO15				GPIO14				GPIO13				GPIO12			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO11				GPIO10				GPIO9				GPIO8			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-27. GPACSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO15	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO14	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO13	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO12	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO11	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO10	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO9	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO8	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.14 GPACSEL3 Register (Offset = 2Ch) [Reset = 0h]

GPACSEL3 is shown in Figure 15-17 and described in Table 15-28.

Return to the Summary Table.

GPIO A Core Select Register (GPIO16 to 23)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPACSEL[0] is used. Writing to GPACSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-17. GPACSEL3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO23				GPIO22				GPIO21				GPIO20			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO19				GPIO18				GPIO17				GPIO16			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-28. GPACSEL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO23	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO22	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO21	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO20	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO19	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO18	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO17	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO16	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.15 GPACSEL4 Register (Offset = 2Eh) [Reset = 0h]

GPACSEL4 is shown in Figure 15-18 and described in Table 15-29.

Return to the Summary Table.

GPIO A Core Select Register (GPIO24 to 31)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPACSEL[0] is used. Writing to GPACSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-18. GPACSEL4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO31				GPIO30				GPIO29				GPIO28			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO27				GPIO26				GPIO25				GPIO24			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-29. GPACSEL4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO31	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO30	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO29	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO28	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO27	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO26	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO25	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO24	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.16 GPALOCK Register (Offset = 3Ch) [Reset = 0h]

GPALOCK is shown in Figure 15-19 and described in Table 15-30.

Return to the Summary Table.

GPIO A Lock Configuration Register (GPIO0 to 31)
GPIO Configuration Lock for GPIO.
0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed
1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin

Figure 15-19. GPALOCK Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-30. GPALOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
30	GPIO30	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
29	GPIO29	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
28	GPIO28	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
27	GPIO27	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
26	GPIO26	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
25	GPIO25	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
24	GPIO24	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
23	GPIO23	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
22	GPIO22	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
21	GPIO21	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
20	GPIO20	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn

Table 15-30. GPALOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	GPIO19	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
18	GPIO18	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
17	GPIO17	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
16	GPIO16	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
15	GPIO15	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
14	GPIO14	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
13	GPIO13	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
12	GPIO12	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
11	GPIO11	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
10	GPIO10	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
9	GPIO9	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
8	GPIO8	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
7	GPIO7	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
6	GPIO6	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
5	GPIO5	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
4	GPIO4	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
3	GPIO3	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
2	GPIO2	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
1	GPIO1	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
0	GPIO0	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn

15.10.3.17 GPACR Register (Offset = 3Eh) [Reset = 0h]

GPACR is shown in Figure 15-20 and described in Table 15-31.

Return to the Summary Table.

GPIO A Lock Commit Register (GPIO0 to 31)
GPIO Configuration Lock Commit for GPIO:
1: Locks changes to the bit in GPyLOCK register which controls the same pin
0: Bit in the GPyLOCK register which controls the same pin can be changed

Figure 15-20. GPACR Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h

Table 15-31. GPACR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
30	GPIO30	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
29	GPIO29	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
28	GPIO28	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
27	GPIO27	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
26	GPIO26	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
25	GPIO25	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
24	GPIO24	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
23	GPIO23	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
22	GPIO22	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
21	GPIO21	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
20	GPIO20	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
19	GPIO19	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn

Table 15-31. GPACR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO18	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
17	GPIO17	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
16	GPIO16	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
15	GPIO15	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
14	GPIO14	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
13	GPIO13	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
12	GPIO12	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
11	GPIO11	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
10	GPIO10	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
9	GPIO9	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
8	GPIO8	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
7	GPIO7	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
6	GPIO6	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
5	GPIO5	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
4	GPIO4	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
3	GPIO3	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
2	GPIO2	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
1	GPIO1	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
0	GPIO0	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn

15.10.3.18 GPBCTRL Register (Offset = 40h) [Reset = 0h]

GPBCTRL is shown in Figure 15-21 and described in Table 15-32.

Return to the Summary Table.

GPIO B Qualification Sampling Period Control (GPIO32 to 63)

Figure 15-21. GPBCTRL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUALPRD3								QUALPRD2								QUALPRD1								QUALPRD0							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 15-32. GPBCTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	QUALPRD3	R/W	0h	Qualification sampling period for GPIO56 to GPIO63: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn
23-16	QUALPRD2	R/W	0h	Qualification sampling period for GPIO48 to GPIO55: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn
15-8	QUALPRD1	R/W	0h	Qualification sampling period for GPIO40 to GPIO47: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn
7-0	QUALPRD0	R/W	0h	Qualification sampling period for GPIO32 to GPIO39: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn

15.10.3.19 GPBQSEL1 Register (Offset = 42h) [Reset = 0h]

GPBQSEL1 is shown in Figure 15-22 and described in Table 15-33.

Return to the Summary Table.

GPIO B Qualifier Select 1 Register (GPIO32 to 47)
Input qualification type:
0,0 Sync
0,1 Qualification (3 samples)
1,0 Qualification (6 samples)
1,1 Async (no Sync or Qualification)

Figure 15-22. GPBQSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO47		GPIO46		GPIO45		GPIO44		GPIO43		GPIO42		GPIO41		GPIO40	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO39		GPIO38		GPIO37		GPIO36		GPIO35		GPIO34		GPIO33		GPIO32	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-33. GPBQSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO47	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
29-28	GPIO46	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
27-26	GPIO45	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
25-24	GPIO44	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
23-22	GPIO43	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
21-20	GPIO42	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
19-18	GPIO41	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
17-16	GPIO40	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
15-14	GPIO39	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
13-12	GPIO38	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
11-10	GPIO37	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
9-8	GPIO36	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
7-6	GPIO35	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
5-4	GPIO34	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
3-2	GPIO33	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
1-0	GPIO32	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn

15.10.3.20 GPBQSEL2 Register (Offset = 44h) [Reset = 0h]

GPBQSEL2 is shown in Figure 15-23 and described in Table 15-34.

Return to the Summary Table.

GPIO B Qualifier Select 2 Register (GPIO48 to 63)
Input qualification type:
0,0 Sync
0,1 Qualification (3 samples)
1,0 Qualification (6 samples)
1,1 Async (no Sync or Qualification)

Figure 15-23. GPBQSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO63		GPIO62		GPIO61		GPIO60		GPIO59		GPIO58		GPIO57		GPIO56	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO55		GPIO54		GPIO53		GPIO52		GPIO51		GPIO50		GPIO49		GPIO48	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-34. GPBQSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO63	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
29-28	GPIO62	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
27-26	GPIO61	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
25-24	GPIO60	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
23-22	GPIO59	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
21-20	GPIO58	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
19-18	GPIO57	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
17-16	GPIO56	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
15-14	GPIO55	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
13-12	GPIO54	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
11-10	GPIO53	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
9-8	GPIO52	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
7-6	GPIO51	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
5-4	GPIO50	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
3-2	GPIO49	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
1-0	GPIO48	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn

15.10.3.21 GPBMUX1 Register (Offset = 46h) [Reset = 0h]

GPBMUX1 is shown in Figure 15-24 and described in Table 15-35.

Return to the Summary Table.

GPIO B Mux 1 Register (GPIO32 to 47)
Defines pin-muxing selection for GPIO.
Notes:
The respective GPyGMUXn.GPIOz must be configured prior to this register to avoid intermediate peripheral selects being mapped to the GPIO.

Figure 15-24. GPBMUX1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO47		GPIO46		GPIO45		GPIO44		GPIO43		GPIO42		GPIO41		GPIO40	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO39		GPIO38		GPIO37		GPIO36		GPIO35		GPIO34		GPIO33		GPIO32	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-35. GPBMUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO47	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO46	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO45	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO44	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO43	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO42	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO41	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO40	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO39	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO38	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO37	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO36	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO35	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
5-4	GPIO34	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
3-2	GPIO33	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO32	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.22 GPBMUX2 Register (Offset = 48h) [Reset = 0h]

GPBMUX2 is shown in Figure 15-25 and described in Table 15-36.

Return to the Summary Table.

GPIO B Mux 2 Register (GPIO48 to 63)
Defines pin-muxing selection for GPIO.
Notes:
The respective GPyGMUXn.GPIOz must be configured prior to this register to avoid intermediate peripheral selects being mapped to the GPIO.

Figure 15-25. GPBMUX2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO63		GPIO62		GPIO61		GPIO60		GPIO59		GPIO58		GPIO57		GPIO56	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO55		GPIO54		GPIO53		GPIO52		GPIO51		GPIO50		GPIO49		GPIO48	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-36. GPBMUX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO63	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO62	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO61	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO60	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO59	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO58	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO57	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO56	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO55	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO54	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO53	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO52	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO51	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
5-4	GPIO50	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
3-2	GPIO49	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO48	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.23 GPBDIR Register (Offset = 4Ah) [Reset = 0h]

GPBDIR is shown in Figure 15-26 and described in Table 15-37.

Return to the Summary Table.

GPIO A Direction Register (GPIO0 to 31)
Controls direction of GPIO pins when the specified pin is configured in GPIO mode.
0: Configures pin as input.
1: Configures pin as output.
Reading the register returns the current value of the register setting.

Figure 15-26. GPBDIR Register

31	30	29	28	27	26	25	24
GPIO63	GPIO62	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	GPIO38	GPIO37	GPIO36	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-37. GPBDIR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO63	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
30	GPIO62	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
29	GPIO61	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
28	GPIO60	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
27	GPIO59	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
26	GPIO58	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
25	GPIO57	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
24	GPIO56	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
23	GPIO55	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
22	GPIO54	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
21	GPIO53	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
20	GPIO52	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
19	GPIO51	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn

Table 15-37. GPBDIR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO50	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
17	GPIO49	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
16	GPIO48	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
15	GPIO47	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
14	GPIO46	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
13	GPIO45	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
12	GPIO44	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
11	GPIO43	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
10	GPIO42	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
9	GPIO41	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
8	GPIO40	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
7	GPIO39	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
6	GPIO38	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
5	GPIO37	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
4	GPIO36	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
3	GPIO35	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
2	GPIO34	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
1	GPIO33	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
0	GPIO32	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn

15.10.3.24 GPBPUD Register (Offset = 4Ch) [Reset = FFFFFFFFh]

GPBPUD is shown in Figure 15-27 and described in Table 15-38.

Return to the Summary Table.

GPIO B Pull Up Disable Register (GPIO32 to 63)
Disables the Pull-Up on GPIO.
0: Enables the Pull-Up.
1: Disables the Pull-Up.
Reading the register returns the current value of the register setting.

Figure 15-27. GPBPUD Register

31	30	29	28	27	26	25	24
GPIO63	GPIO62	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
GPIO39	GPIO38	GPIO37	GPIO36	GPIO35	GPIO34	GPIO33	GPIO32
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 15-38. GPBPUD Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO63	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
30	GPIO62	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
29	GPIO61	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
28	GPIO60	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
27	GPIO59	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
26	GPIO58	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
25	GPIO57	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
24	GPIO56	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
23	GPIO55	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
22	GPIO54	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
21	GPIO53	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
20	GPIO52	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
19	GPIO51	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn

Table 15-38. GPBPUD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO50	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
17	GPIO49	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
16	GPIO48	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
15	GPIO47	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
14	GPIO46	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
13	GPIO45	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
12	GPIO44	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
11	GPIO43	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
10	GPIO42	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
9	GPIO41	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
8	GPIO40	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
7	GPIO39	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
6	GPIO38	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
5	GPIO37	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
4	GPIO36	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
3	GPIO35	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
2	GPIO34	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
1	GPIO33	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
0	GPIO32	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn

15.10.3.25 GPBINV Register (Offset = 50h) [Reset = 0h]

GPBINV is shown in Figure 15-28 and described in Table 15-39.

Return to the Summary Table.

GPIO B Input Polarity Invert Registers (GPIO32 to 63)
Selects between non-inverted and inverted GPIO input to the device.
0: selects non-inverted GPIO input
1: selects inverted GPIO input
Reading the register returns the current value of the register setting.

Figure 15-28. GPBINV Register

31	30	29	28	27	26	25	24
GPIO63	GPIO62	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	GPIO38	GPIO37	GPIO36	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-39. GPBINV Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO63	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
30	GPIO62	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
29	GPIO61	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
28	GPIO60	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
27	GPIO59	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
26	GPIO58	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
25	GPIO57	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
24	GPIO56	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
23	GPIO55	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
22	GPIO54	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
21	GPIO53	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
20	GPIO52	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
19	GPIO51	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn

Table 15-39. GPBINV Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO50	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
17	GPIO49	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
16	GPIO48	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
15	GPIO47	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
14	GPIO46	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
13	GPIO45	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
12	GPIO44	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
11	GPIO43	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
10	GPIO42	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
9	GPIO41	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
8	GPIO40	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
7	GPIO39	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
6	GPIO38	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
5	GPIO37	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
4	GPIO36	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
3	GPIO35	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
2	GPIO34	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
1	GPIO33	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
0	GPIO32	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn

15.10.3.26 GPBODR Register (Offset = 52h) [Reset = 0h]

GPBODR is shown in Figure 15-29 and described in Table 15-40.

Return to the Summary Table.

GPIO B Open Drain Output Register (GPIO32 to GPIO63)
Selects between normal and open-drain output for the GPIO pin.
0: Normal Output
1: Open Drain Output
Reading the register returns the current value of the register setting.
Note:
[1] In the Open Drain output mode, if the buffer is configured for output mode, a 0 value to be driven out comes out on the on the PAD while a 1 value to be driven out tri-states the buffer.

Figure 15-29. GPBODR Register

31	30	29	28	27	26	25	24
GPIO63	GPIO62	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	GPIO38	GPIO37	GPIO36	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-40. GPBODR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO63	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
30	GPIO62	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
29	GPIO61	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
28	GPIO60	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
27	GPIO59	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
26	GPIO58	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
25	GPIO57	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
24	GPIO56	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
23	GPIO55	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
22	GPIO54	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
21	GPIO53	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn

Table 15-40. GPBODR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	GPIO52	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
19	GPIO51	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
18	GPIO50	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
17	GPIO49	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
16	GPIO48	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
15	GPIO47	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
14	GPIO46	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
13	GPIO45	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
12	GPIO44	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
11	GPIO43	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
10	GPIO42	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
9	GPIO41	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
8	GPIO40	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
7	GPIO39	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
6	GPIO38	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
5	GPIO37	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
4	GPIO36	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
3	GPIO35	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
2	GPIO34	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
1	GPIO33	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
0	GPIO32	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn

15.10.3.27 GPBAMSEL Register (Offset = 54h) [Reset = 0h]

GPBAMSEL is shown in Figure 15-30 and described in Table 15-41.

Return to the Summary Table.

GPIO B Analog Mode Select register
Selects between digital and analog functionality for GPIO pins.
0: The pin is configured to digital functions according to the other GPIO configuration registers
1: The analog function of the pin is enabled

Figure 15-30. GPBAMSEL Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	GPIO43	GPIO42	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-41. GPBAMSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	RESERVED	R/W	0h	Reserved
28	RESERVED	R/W	0h	Reserved
27	RESERVED	R/W	0h	Reserved
26	RESERVED	R/W	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23	RESERVED	R/W	0h	Reserved
22	RESERVED	R/W	0h	Reserved
21	RESERVED	R/W	0h	Reserved
20	RESERVED	R/W	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	GPIO43	R/W	0h	Selects the USB0DP function Reset type: CPU1.SYSRSn

Table 15-41. GPBAMSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	GPIO42	R/W	0h	Selects the USB0DM function Reset type: CPU1.SYSRSn
9	RESERVED	R/W	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	RESERVED	R/W	0h	Reserved

15.10.3.28 GPBGMUX1 Register (Offset = 60h) [Reset = 0h]

GPBGMUX1 is shown in Figure 15-31 and described in Table 15-42.

Return to the Summary Table.

GPIO B Peripheral Group Mux (GPIO32 to 47)
Defines pin-muxing selection for GPIO.
Note: For complete pin-mux selection on GPIOx, GPBMUXy.GPIOx configuration is also required.

Figure 15-31. GPBGMUX1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO47		GPIO46		GPIO45		GPIO44		GPIO43		GPIO42		GPIO41		GPIO40	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO39		GPIO38		GPIO37		GPIO36		GPIO35		GPIO34		GPIO33		GPIO32	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-42. GPBGMUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO47	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO46	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO45	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO44	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO43	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO42	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO41	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO40	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO39	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO38	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO37	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO36	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO35	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
5-4	GPIO34	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
3-2	GPIO33	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO32	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.29 GPBGMUX2 Register (Offset = 62h) [Reset = 0h]

GPBGMUX2 is shown in Figure 15-32 and described in Table 15-43.

Return to the Summary Table.

GPIO B Peripheral Group Mux (GPIO48 to 63)
Defines pin-muxing selection for GPIO.
Note: For complete pin-mux selection on GPIOx, GPBMUXy.GPIOx configuration is also required.

Figure 15-32. GPBGMUX2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO63		GPIO62		GPIO61		GPIO60		GPIO59		GPIO58		GPIO57		GPIO56	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO55		GPIO54		GPIO53		GPIO52		GPIO51		GPIO50		GPIO49		GPIO48	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-43. GPBGMUX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO63	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO62	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO61	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO60	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO59	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO58	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO57	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO56	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO55	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO54	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO53	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO52	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO51	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
5-4	GPIO50	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
3-2	GPIO49	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO48	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.30 GPBCSEL1 Register (Offset = 68h) [Reset = 0h]

GPBCSEL1 is shown in Figure 15-33 and described in Table 15-44.

Return to the Summary Table.

GPIO B Core Select Register (GPIO0 to 7)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPBCSEL[0] is used. Writing to GPBCSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-33. GPBCSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO39				GPIO38				GPIO37				GPIO36			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO35				GPIO34				GPIO33				GPIO32			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-44. GPBCSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO39	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO38	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO37	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO36	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO35	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO34	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO33	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO32	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.31 GPBCSEL2 Register (Offset = 6Ah) [Reset = 0h]

GPBCSEL2 is shown in Figure 15-34 and described in Table 15-45.

Return to the Summary Table.

GPIO B Core Select Register (GPIO8 to 15)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPBCSEL[0] is used. Writing to GPBCSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-34. GPBCSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO47				GPIO46				GPIO45				GPIO44			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO43				GPIO42				GPIO41				GPIO40			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-45. GPBCSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO47	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO46	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO45	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO44	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO43	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO42	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO41	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO40	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.32 GPBCSEL3 Register (Offset = 6Ch) [Reset = 0h]

GPBCSEL3 is shown in Figure 15-35 and described in Table 15-46.

Return to the Summary Table.

GPIO B Core Select Register (GPIO16 to 23)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPBCSEL[0] is used. Writing to GPBCSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-35. GPBCSEL3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO55				GPIO54				GPIO53				GPIO52			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO51				GPIO50				GPIO49				GPIO48			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-46. GPBCSEL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO55	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO54	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO53	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO52	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO51	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO50	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO49	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO48	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.33 GPBCSEL4 Register (Offset = 6Eh) [Reset = 0h]

GPBCSEL4 is shown in Figure 15-36 and described in Table 15-47.

Return to the Summary Table.

GPIO B Core Select Register (GPIO24 to 31)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPBCSEL[0] is used. Writing to GPBCSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-36. GPBCSEL4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO63				GPIO62				GPIO61				GPIO60			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO59				GPIO58				GPIO57				GPIO56			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-47. GPBCSEL4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO63	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO62	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO61	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO60	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO59	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO58	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO57	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO56	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.34 GPBLOCK Register (Offset = 7Ch) [Reset = 0h]

GPBLOCK is shown in Figure 15-37 and described in Table 15-48.

Return to the Summary Table.

GPIO B Lock Configuration Register (GPIO32 to 63)
GPIO Configuration Lock for GPIO.
0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed
1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin

Figure 15-37. GPBLOCK Register

31	30	29	28	27	26	25	24
GPIO63	GPIO62	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	GPIO38	GPIO37	GPIO36	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-48. GPBLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO63	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
30	GPIO62	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
29	GPIO61	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
28	GPIO60	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
27	GPIO59	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
26	GPIO58	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
25	GPIO57	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
24	GPIO56	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
23	GPIO55	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
22	GPIO54	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
21	GPIO53	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
20	GPIO52	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn

Table 15-48. GPBLOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	GPIO51	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
18	GPIO50	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
17	GPIO49	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
16	GPIO48	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
15	GPIO47	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
14	GPIO46	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
13	GPIO45	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
12	GPIO44	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
11	GPIO43	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
10	GPIO42	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
9	GPIO41	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
8	GPIO40	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
7	GPIO39	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
6	GPIO38	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
5	GPIO37	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
4	GPIO36	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
3	GPIO35	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
2	GPIO34	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
1	GPIO33	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
0	GPIO32	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn

15.10.3.35 GPBCR Register (Offset = 7Eh) [Reset = 0h]

GPBCR is shown in Figure 15-38 and described in Table 15-49.

Return to the Summary Table.

GPIO B Lock Commit Register (GPIO32 to 63)
GPIO Configuration Lock Commit for GPIO:
1: Locks changes to the bit in GPyLOCK register which controls the same pin
0: Bit in the GPyLOCK register which controls the same pin can be changed

Figure 15-38. GPBCR Register

31	30	29	28	27	26	25	24
GPIO63	GPIO62	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h
7	6	5	4	3	2	1	0
GPIO39	GPIO38	GPIO37	GPIO36	GPIO35	GPIO34	GPIO33	GPIO32
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h

Table 15-49. GPBCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO63	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
30	GPIO62	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
29	GPIO61	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
28	GPIO60	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
27	GPIO59	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
26	GPIO58	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
25	GPIO57	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
24	GPIO56	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
23	GPIO55	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
22	GPIO54	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
21	GPIO53	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
20	GPIO52	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
19	GPIO51	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn

Table 15-49. GPBCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO50	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
17	GPIO49	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
16	GPIO48	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
15	GPIO47	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
14	GPIO46	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
13	GPIO45	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
12	GPIO44	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
11	GPIO43	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
10	GPIO42	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
9	GPIO41	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
8	GPIO40	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
7	GPIO39	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
6	GPIO38	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
5	GPIO37	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
4	GPIO36	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
3	GPIO35	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
2	GPIO34	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
1	GPIO33	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
0	GPIO32	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn

15.10.3.36 GPCCTRL Register (Offset = 80h) [Reset = 0h]

GPCCTRL is shown in Figure 15-39 and described in Table 15-50.

Return to the Summary Table.

GPIO C Qualification Sampling Period Control (GPIO64 to 95)

Figure 15-39. GPCCTRL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUALPRD3								QUALPRD2								QUALPRD1								QUALPRD0							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 15-50. GPCCTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	QUALPRD3	R/W	0h	Qualification sampling period for GPIO88 to GPIO95: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn
23-16	QUALPRD2	R/W	0h	Qualification sampling period for GPIO80 to GPIO87: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn
15-8	QUALPRD1	R/W	0h	Qualification sampling period for GPIO72 to GPIO79: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn
7-0	QUALPRD0	R/W	0h	Qualification sampling period for GPIO64 to GPIO71: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn

15.10.3.37 GPCQSEL1 Register (Offset = 82h) [Reset = 0h]

GPCQSEL1 is shown in Figure 15-40 and described in Table 15-51.

Return to the Summary Table.

GPIO C Qualifier Select 1 Register (GPIO64 to 79)
Input qualification type:
0,0 Sync
0,1 Qualification (3 samples)
1,0 Qualification (6 samples)
1,1 Async (no Sync or Qualification)

Figure 15-40. GPCQSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO79		GPIO78		GPIO77		GPIO76		GPIO75		GPIO74		GPIO73		GPIO72	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO71		GPIO70		GPIO69		GPIO68		GPIO67		GPIO66		GPIO65		GPIO64	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-51. GPCQSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO79	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
29-28	GPIO78	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
27-26	GPIO77	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
25-24	GPIO76	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
23-22	GPIO75	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
21-20	GPIO74	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
19-18	GPIO73	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
17-16	GPIO72	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
15-14	GPIO71	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
13-12	GPIO70	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
11-10	GPIO69	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
9-8	GPIO68	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
7-6	GPIO67	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
5-4	GPIO66	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
3-2	GPIO65	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
1-0	GPIO64	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn

15.10.3.38 GPCQSEL2 Register (Offset = 84h) [Reset = 0h]

GPCQSEL2 is shown in Figure 15-41 and described in Table 15-52.

Return to the Summary Table.

GPIO C Qualifier Select 2 Register (GPIO80 to 95)
Input qualification type:
0,0 Sync
0,1 Qualification (3 samples)
1,0 Qualification (6 samples)
1,1 Async (no Sync or Qualification)

Figure 15-41. GPCQSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO95		GPIO94		GPIO93		GPIO92		GPIO91		GPIO90		GPIO89		GPIO88	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO87		GPIO86		GPIO85		GPIO84		GPIO83		GPIO82		GPIO81		GPIO80	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-52. GPCQSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO95	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
29-28	GPIO94	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
27-26	GPIO93	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
25-24	GPIO92	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
23-22	GPIO91	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
21-20	GPIO90	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
19-18	GPIO89	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
17-16	GPIO88	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
15-14	GPIO87	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
13-12	GPIO86	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
11-10	GPIO85	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
9-8	GPIO84	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
7-6	GPIO83	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
5-4	GPIO82	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
3-2	GPIO81	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
1-0	GPIO80	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn

15.10.3.39 GPCMUX1 Register (Offset = 86h) [Reset = 0h]

GPCMUX1 is shown in Figure 15-42 and described in Table 15-53.

Return to the Summary Table.

GPIO C Mux 1 Register (GPIO64 to 79)
Defines pin-muxing selection for GPIO.
Notes:
The respective GPyGMUXn.GPIOz must be configured prior to this register to avoid intermediate peripheral selects being mapped to the GPIO.

Figure 15-42. GPCMUX1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO79		GPIO78		GPIO77		GPIO76		GPIO75		GPIO74		GPIO73		GPIO72	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO71		GPIO70		GPIO69		GPIO68		GPIO67		GPIO66		GPIO65		GPIO64	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-53. GPCMUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO79	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO78	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO77	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO76	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO75	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO74	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO73	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO72	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO71	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO70	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO69	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO68	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO67	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
5-4	GPIO66	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
3-2	GPIO65	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO64	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.40 GPCMUX2 Register (Offset = 88h) [Reset = 0h]

GPCMUX2 is shown in Figure 15-43 and described in Table 15-54.

Return to the Summary Table.

GPIO C Mux 2 Register (GPIO80 to 95)
Defines pin-muxing selection for GPIO.
Notes:
The respective GPyGMUXn.GPIOz must be configured prior to this register to avoid intermediate peripheral selects being mapped to the GPIO.

Figure 15-43. GPCMUX2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO95		GPIO94		GPIO93		GPIO92		GPIO91		GPIO90		GPIO89		GPIO88	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO87		GPIO86		GPIO85		GPIO84		GPIO83		GPIO82		GPIO81		GPIO80	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-54. GPCMUX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO95	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO94	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO93	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO92	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO91	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO90	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO89	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO88	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO87	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO86	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO85	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO84	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO83	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
5-4	GPIO82	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
3-2	GPIO81	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO80	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.41 GPCDIR Register (Offset = 8Ah) [Reset = 0h]

GPCDIR is shown in Figure 15-44 and described in Table 15-55.

Return to the Summary Table.

GPIO C Direction Register (GPIO64 to 95)
Controls direction of GPIO pins when the specified pin is configured in GPIO mode.
0: Configures pin as input.
1: Configures pin as output.
Reading the register returns the current value of the register setting.

Figure 15-44. GPCDIR Register

31	30	29	28	27	26	25	24
GPIO95	GPIO94	GPIO93	GPIO92	GPIO91	GPIO90	GPIO89	GPIO88
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO87	GPIO86	GPIO85	GPIO84	GPIO83	GPIO82	GPIO81	GPIO80
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO79	GPIO78	GPIO77	GPIO76	GPIO75	GPIO74	GPIO73	GPIO72
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO71	GPIO70	GPIO69	GPIO68	GPIO67	GPIO66	GPIO65	GPIO64
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-55. GPCDIR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO95	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
30	GPIO94	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
29	GPIO93	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
28	GPIO92	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
27	GPIO91	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
26	GPIO90	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
25	GPIO89	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
24	GPIO88	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
23	GPIO87	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
22	GPIO86	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
21	GPIO85	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
20	GPIO84	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
19	GPIO83	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn

Table 15-55. GPCDIR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO82	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
17	GPIO81	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
16	GPIO80	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
15	GPIO79	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
14	GPIO78	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
13	GPIO77	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
12	GPIO76	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
11	GPIO75	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
10	GPIO74	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
9	GPIO73	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
8	GPIO72	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
7	GPIO71	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
6	GPIO70	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
5	GPIO69	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
4	GPIO68	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
3	GPIO67	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
2	GPIO66	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
1	GPIO65	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
0	GPIO64	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn

15.10.3.42 GPCPUD Register (Offset = 8Ch) [Reset = FFFFFFFFh]

GPCPUD is shown in Figure 15-45 and described in Table 15-56.

Return to the Summary Table.

GPIO C Pull Up Disable Register (GPIO64 to 95)
Disables the Pull-Up on GPIO.
0: Enables the Pull-Up.
1: Disables the Pull-Up.
Reading the register returns the current value of the register setting.

Figure 15-45. GPCPUD Register

31	30	29	28	27	26	25	24
GPIO95	GPIO94	GPIO93	GPIO92	GPIO91	GPIO90	GPIO89	GPIO88
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
GPIO87	GPIO86	GPIO85	GPIO84	GPIO83	GPIO82	GPIO81	GPIO80
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
GPIO79	GPIO78	GPIO77	GPIO76	GPIO75	GPIO74	GPIO73	GPIO72
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
GPIO71	GPIO70	GPIO69	GPIO68	GPIO67	GPIO66	GPIO65	GPIO64
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 15-56. GPCPUD Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO95	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
30	GPIO94	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
29	GPIO93	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
28	GPIO92	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
27	GPIO91	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
26	GPIO90	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
25	GPIO89	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
24	GPIO88	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
23	GPIO87	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
22	GPIO86	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
21	GPIO85	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
20	GPIO84	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
19	GPIO83	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn

Table 15-56. GPCPUD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO82	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
17	GPIO81	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
16	GPIO80	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
15	GPIO79	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
14	GPIO78	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
13	GPIO77	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
12	GPIO76	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
11	GPIO75	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
10	GPIO74	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
9	GPIO73	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
8	GPIO72	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
7	GPIO71	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
6	GPIO70	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
5	GPIO69	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
4	GPIO68	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
3	GPIO67	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
2	GPIO66	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
1	GPIO65	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
0	GPIO64	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn

15.10.3.43 GPCINV Register (Offset = 90h) [Reset = 0h]

GPCINV is shown in Figure 15-46 and described in Table 15-57.

Return to the Summary Table.

GPIO C Input Polarity Invert Registers (GPIO64 to 95)
Selects between non-inverted and inverted GPIO input to the device.
0: selects non-inverted GPIO input
1: selects inverted GPIO input
Reading the register returns the current value of the register setting.

Figure 15-46. GPCINV Register

31	30	29	28	27	26	25	24
GPIO95	GPIO94	GPIO93	GPIO92	GPIO91	GPIO90	GPIO89	GPIO88
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO87	GPIO86	GPIO85	GPIO84	GPIO83	GPIO82	GPIO81	GPIO80
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO79	GPIO78	GPIO77	GPIO76	GPIO75	GPIO74	GPIO73	GPIO72
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO71	GPIO70	GPIO69	GPIO68	GPIO67	GPIO66	GPIO65	GPIO64
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-57. GPCINV Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO95	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
30	GPIO94	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
29	GPIO93	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
28	GPIO92	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
27	GPIO91	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
26	GPIO90	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
25	GPIO89	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
24	GPIO88	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
23	GPIO87	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
22	GPIO86	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
21	GPIO85	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
20	GPIO84	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
19	GPIO83	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn

Table 15-57. GPCINV Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO82	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
17	GPIO81	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
16	GPIO80	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
15	GPIO79	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
14	GPIO78	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
13	GPIO77	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
12	GPIO76	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
11	GPIO75	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
10	GPIO74	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
9	GPIO73	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
8	GPIO72	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
7	GPIO71	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
6	GPIO70	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
5	GPIO69	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
4	GPIO68	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
3	GPIO67	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
2	GPIO66	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
1	GPIO65	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
0	GPIO64	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn

15.10.3.44 GPCODR Register (Offset = 92h) [Reset = 0h]

GPCODR is shown in Figure 15-47 and described in Table 15-58.

Return to the Summary Table.

GPIO Open Drain Output Register
Selects between normal and open-drain output for the GPIO pin.
0: Normal Output
1: Open Drain Output
CAUTION: Open Drain mode is not supported on this device. Keep this bit as 0. Open drain can be emulated by writing 0 to GPxDAT and enabling or disabling the output buffer with GPxDIR. (Peripherals such as I2C support Open Drain mode without issue).

Figure 15-47. GPCODR Register

31	30	29	28	27	26	25	24
GPIO95	GPIO94	GPIO93	GPIO92	GPIO91	GPIO90	GPIO89	GPIO88
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO87	GPIO86	GPIO85	GPIO84	GPIO83	GPIO82	GPIO81	GPIO80
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO79	GPIO78	GPIO77	GPIO76	GPIO75	GPIO74	GPIO73	GPIO72
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO71	GPIO70	GPIO69	GPIO68	GPIO67	GPIO66	GPIO65	GPIO64
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-58. GPCODR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO95	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
30	GPIO94	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
29	GPIO93	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
28	GPIO92	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
27	GPIO91	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
26	GPIO90	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
25	GPIO89	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
24	GPIO88	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
23	GPIO87	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
22	GPIO86	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
21	GPIO85	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
20	GPIO84	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn

Table 15-58. GPCODR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	GPIO83	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
18	GPIO82	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
17	GPIO81	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
16	GPIO80	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
15	GPIO79	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
14	GPIO78	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
13	GPIO77	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
12	GPIO76	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
11	GPIO75	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
10	GPIO74	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
9	GPIO73	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
8	GPIO72	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
7	GPIO71	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
6	GPIO70	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
5	GPIO69	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
4	GPIO68	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
3	GPIO67	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
2	GPIO66	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
1	GPIO65	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
0	GPIO64	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn

15.10.3.45 GPCGMUX1 Register (Offset = A0h) [Reset = 0h]

GPCGMUX1 is shown in Figure 15-48 and described in Table 15-59.

Return to the Summary Table.

GPIO C Peripheral Group Mux (GPIO64 to 79)
Defines pin-muxing selection for GPIO.
Note: For complete pin-mux selection on GPIOx, GPCMUXy.GPIOx configuration is also required.

Figure 15-48. GPCGMUX1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO79		GPIO78		GPIO77		GPIO76		GPIO75		GPIO74		GPIO73		GPIO72	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO71		GPIO70		GPIO69		GPIO68		GPIO67		GPIO66		GPIO65		GPIO64	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-59. GPCGMUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO79	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO78	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO77	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO76	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO75	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO74	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO73	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO72	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO71	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO70	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO69	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO68	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO67	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
5-4	GPIO66	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
3-2	GPIO65	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO64	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.46 GPCGMUX2 Register (Offset = A2h) [Reset = 0h]

GPCGMUX2 is shown in Figure 15-49 and described in Table 15-60.

Return to the Summary Table.

GPIO C Peripheral Group Mux (GPIO80 to 95)
Defines pin-muxing selection for GPIO.
Note: For complete pin-mux selection on GPIOx, GPCMUXy.GPIOx configuration is also required.

Figure 15-49. GPCGMUX2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO95		GPIO94		GPIO93		GPIO92		GPIO91		GPIO90		GPIO89		GPIO88	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO87		GPIO86		GPIO85		GPIO84		GPIO83		GPIO82		GPIO81		GPIO80	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-60. GPCGMUX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO95	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO94	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO93	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO92	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO91	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO90	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO89	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO88	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO87	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO86	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO85	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO84	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO83	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
5-4	GPIO82	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
3-2	GPIO81	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO80	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.47 GPCCSEL1 Register (Offset = A8h) [Reset = 0h]

GPCCSEL1 is shown in Figure 15-50 and described in Table 15-61.

Return to the Summary Table.

GPIO C Core Select Register (GPIO0 to 7)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPCCSEL[0] is used. Writing to GPCCSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-50. GPCCSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO71				GPIO70				GPIO69				GPIO68			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO67				GPIO66				GPIO65				GPIO64			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-61. GPCCSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO71	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO70	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO69	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO68	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO67	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO66	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO65	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO64	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.48 GPCCSEL2 Register (Offset = AAh) [Reset = 0h]

GPCCSEL2 is shown in Figure 15-51 and described in Table 15-62.

Return to the Summary Table.

GPIO C Core Select Register (GPIO8 to 15)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPCCSEL[0] is used. Writing to GPCCSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-51. GPCCSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO79				GPIO78				GPIO77				GPIO76			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO75				GPIO74				GPIO73				GPIO72			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-62. GPCCSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO79	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO78	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO77	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO76	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO75	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO74	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO73	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO72	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.49 GPCCSEL3 Register (Offset = ACh) [Reset = 0h]

GPCCSEL3 is shown in Figure 15-52 and described in Table 15-63.

Return to the Summary Table.

GPIO C Core Select Register (GPIO16 to 23)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPCCSEL[0] is used. Writing to GPCCSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-52. GPCCSEL3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO87				GPIO86				GPIO85				GPIO84			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO83				GPIO82				GPIO81				GPIO80			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-63. GPCCSEL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO87	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO86	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO85	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO84	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO83	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO82	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO81	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO80	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.50 GPCCSEL4 Register (Offset = AEh) [Reset = 0h]

GPCCSEL4 is shown in Figure 15-53 and described in Table 15-64.

Return to the Summary Table.

GPIO C Core Select Register (GPIO24 to 31)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPCCSEL[0] is used. Writing to GPCCSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-53. GPCCSEL4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO95				GPIO94				GPIO93				GPIO92			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO91				GPIO90				GPIO89				GPIO88			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-64. GPCCSEL4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO95	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO94	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO93	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO92	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO91	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO90	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO89	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO88	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.51 GPCLOCK Register (Offset = BCh) [Reset = 0h]

GPCLOCK is shown in Figure 15-54 and described in Table 15-65.

Return to the Summary Table.

GPIO C Lock Configuration Register (GPIO64 to 95)
GPIO Configuration Lock for GPIO.
0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed
1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin

Figure 15-54. GPCLOCK Register

31	30	29	28	27	26	25	24
GPIO95	GPIO94	GPIO93	GPIO92	GPIO91	GPIO90	GPIO89	GPIO88
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO87	GPIO86	GPIO85	GPIO84	GPIO83	GPIO82	GPIO81	GPIO80
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO79	GPIO78	GPIO77	GPIO76	GPIO75	GPIO74	GPIO73	GPIO72
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO71	GPIO70	GPIO69	GPIO68	GPIO67	GPIO66	GPIO65	GPIO64
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-65. GPCLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO95	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
30	GPIO94	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
29	GPIO93	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
28	GPIO92	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
27	GPIO91	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
26	GPIO90	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
25	GPIO89	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
24	GPIO88	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
23	GPIO87	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
22	GPIO86	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
21	GPIO85	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
20	GPIO84	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn

Table 15-65. GPCLOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	GPIO83	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
18	GPIO82	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
17	GPIO81	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
16	GPIO80	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
15	GPIO79	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
14	GPIO78	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
13	GPIO77	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
12	GPIO76	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
11	GPIO75	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
10	GPIO74	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
9	GPIO73	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
8	GPIO72	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
7	GPIO71	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
6	GPIO70	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
5	GPIO69	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
4	GPIO68	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
3	GPIO67	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
2	GPIO66	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
1	GPIO65	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
0	GPIO64	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn

15.10.3.52 GPCCR Register (Offset = BEh) [Reset = 0h]

GPCCR is shown in Figure 15-55 and described in Table 15-66.

Return to the Summary Table.

GPIO C Lock Commit Register (GPIO64 to 95)
GPIO Configuration Lock Commit for GPIO:
1: Locks changes to the bit in GPyLOCK register which controls the same pin
0: Bit in the GPyLOCK register which controls the same pin can be changed

Figure 15-55. GPCCR Register

31	30	29	28	27	26	25	24
GPIO95	GPIO94	GPIO93	GPIO92	GPIO91	GPIO90	GPIO89	GPIO88
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h
23	22	21	20	19	18	17	16
GPIO87	GPIO86	GPIO85	GPIO84	GPIO83	GPIO82	GPIO81	GPIO80
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h
15	14	13	12	11	10	9	8
GPIO79	GPIO78	GPIO77	GPIO76	GPIO75	GPIO74	GPIO73	GPIO72
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h
7	6	5	4	3	2	1	0
GPIO71	GPIO70	GPIO69	GPIO68	GPIO67	GPIO66	GPIO65	GPIO64
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h

Table 15-66. GPCCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO95	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
30	GPIO94	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
29	GPIO93	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
28	GPIO92	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
27	GPIO91	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
26	GPIO90	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
25	GPIO89	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
24	GPIO88	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
23	GPIO87	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
22	GPIO86	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
21	GPIO85	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
20	GPIO84	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
19	GPIO83	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn

Table 15-66. GPCCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO82	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
17	GPIO81	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
16	GPIO80	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
15	GPIO79	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
14	GPIO78	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
13	GPIO77	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
12	GPIO76	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
11	GPIO75	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
10	GPIO74	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
9	GPIO73	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
8	GPIO72	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
7	GPIO71	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
6	GPIO70	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
5	GPIO69	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
4	GPIO68	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
3	GPIO67	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
2	GPIO66	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
1	GPIO65	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
0	GPIO64	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn

15.10.3.53 GPDCTRL Register (Offset = C0h) [Reset = 0h]

GPDCTRL is shown in Figure 15-56 and described in Table 15-67.

Return to the Summary Table.

GPIO D Qualification Sampling Period Control (GPIO96 to 127)

Figure 15-56. GPDCTRL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUALPRD3								QUALPRD2								QUALPRD1								QUALPRD0							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 15-67. GPDCTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	QUALPRD3	R/W	0h	Qualification sampling period for GPIO120 to GPIO127: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn
23-16	QUALPRD2	R/W	0h	Qualification sampling period for GPIO112 to GPIO119: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn
15-8	QUALPRD1	R/W	0h	Qualification sampling period for GPIO104 to GPIO111: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn
7-0	QUALPRD0	R/W	0h	Qualification sampling period for GPIO96 to GPIO103: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn

15.10.3.54 GPDQSEL1 Register (Offset = C2h) [Reset = 0h]

GPDQSEL1 is shown in Figure 15-57 and described in Table 15-68.

Return to the Summary Table.

GPIO D Qualifier Select 1 Register (GPIO96 to 111)
Input qualification type:
0,0 Sync
0,1 Qualification (3 samples)
1,0 Qualification (6 samples)
1,1 Async (no Sync or Qualification)

Figure 15-57. GPDQSEL1 Register

31	30	29	28	27	26	25	24
GPIO111		GPIO110		GPIO109		GPIO108	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
GPIO107		GPIO106		GPIO105		GPIO104	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO103		GPIO102		GPIO101		GPIO100	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
GPIO99		GPIO98		GPIO97		GPIO96	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-68. GPDQSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO111	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
29-28	GPIO110	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
27-26	GPIO109	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
25-24	GPIO108	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
23-22	GPIO107	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
21-20	GPIO106	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
19-18	GPIO105	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
17-16	GPIO104	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
15-14	GPIO103	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
13-12	GPIO102	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
11-10	GPIO101	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
9-8	GPIO100	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn

Table 15-68. GPDQSEL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	GPIO99	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
5-4	GPIO98	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
3-2	GPIO97	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
1-0	GPIO96	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn

15.10.3.55 GPDQSEL2 Register (Offset = C4h) [Reset = 0h]

GPDQSEL2 is shown in Figure 15-58 and described in Table 15-69.

Return to the Summary Table.

GPIO D Qualifier Select 2 Register (GPIO112 to 127)
Input qualification type:
0,0 Sync
0,1 Qualification (3 samples)
1,0 Qualification (6 samples)
1,1 Async (no Sync or Qualification)

Figure 15-58. GPDQSEL2 Register

31	30	29	28	27	26	25	24
GPIO127		GPIO126		GPIO125		GPIO124	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
GPIO123		GPIO122		GPIO121		GPIO120	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO119		GPIO118		GPIO117		GPIO116	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
GPIO115		GPIO114		GPIO113		GPIO112	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-69. GPDQSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO127	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
29-28	GPIO126	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
27-26	GPIO125	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
25-24	GPIO124	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
23-22	GPIO123	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
21-20	GPIO122	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
19-18	GPIO121	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
17-16	GPIO120	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
15-14	GPIO119	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
13-12	GPIO118	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
11-10	GPIO117	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
9-8	GPIO116	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn

Table 15-69. GPDQSEL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	GPIO115	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
5-4	GPIO114	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
3-2	GPIO113	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
1-0	GPIO112	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn

15.10.3.56 GPDMUX1 Register (Offset = C6h) [Reset = 0h]

GPDMUX1 is shown in Figure 15-59 and described in Table 15-70.

Return to the Summary Table.

GPIO D Mux 1 Register (GPIO96 to 111)
Defines pin-muxing selection for GPIO.
Notes:
The respective GPyGMUXn.GPIOz must be configured prior to this register to avoid intermediate peripheral selects being mapped to the GPIO.

Figure 15-59. GPDMUX1 Register

31	30	29	28	27	26	25	24
GPIO111		GPIO110		GPIO109		GPIO108	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
GPIO107		GPIO106		GPIO105		GPIO104	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO103		GPIO102		GPIO101		GPIO100	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
GPIO99		GPIO98		GPIO97		GPIO96	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-70. GPDMUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO111	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO110	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO109	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO108	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO107	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO106	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO105	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO104	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO103	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO102	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO101	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO100	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO99	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

Table 15-70. GPDMUX1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-4	GPIO98	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
3-2	GPIO97	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO96	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.57 GPDMUX2 Register (Offset = C8h) [Reset = 0h]

GPDMUX2 is shown in Figure 15-60 and described in Table 15-71.

Return to the Summary Table.

GPIO D Mux 2 Register (GPIO112 to 127)
Defines pin-muxing selection for GPIO.
Notes:
The respective GPyGMUXn.GPIOz must be configured prior to this register to avoid intermediate peripheral selects being mapped to the GPIO.

Figure 15-60. GPDMUX2 Register

31	30	29	28	27	26	25	24
GPIO127		GPIO126		GPIO125		GPIO124	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
GPIO123		GPIO122		GPIO121		GPIO120	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO119		GPIO118		GPIO117		GPIO116	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
GPIO115		GPIO114		GPIO113		GPIO112	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-71. GPDMUX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO127	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO126	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO125	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO124	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO123	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO122	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO121	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO120	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO119	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO118	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO117	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO116	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO115	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

Table 15-71. GPDMUX2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-4	GPIO114	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
3-2	GPIO113	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO112	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.58 GPDDIR Register (Offset = CAh) [Reset = 0h]

GPDDIR is shown in Figure 15-61 and described in Table 15-72.

Return to the Summary Table.

GPIO D Direction Register (GPIO96 to 127)
Controls direction of GPIO pins when the specified pin is configured in GPIO mode.
0: Configures pin as input.
1: Configures pin as output.
Reading the register returns the current value of the register setting.

Figure 15-61. GPDDIR Register

31	30	29	28	27	26	25	24
GPIO127	GPIO126	GPIO125	GPIO124	GPIO123	GPIO122	GPIO121	GPIO120
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO119	GPIO118	GPIO117	GPIO116	GPIO115	GPIO114	GPIO113	GPIO112
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO111	GPIO110	GPIO109	GPIO108	GPIO107	GPIO106	GPIO105	GPIO104
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO103	GPIO102	GPIO101	GPIO100	GPIO99	GPIO98	GPIO97	GPIO96
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-72. GPDDIR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO127	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
30	GPIO126	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
29	GPIO125	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
28	GPIO124	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
27	GPIO123	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
26	GPIO122	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
25	GPIO121	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
24	GPIO120	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
23	GPIO119	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
22	GPIO118	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
21	GPIO117	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
20	GPIO116	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
19	GPIO115	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn

Table 15-72. GPDDIR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO114	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
17	GPIO113	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
16	GPIO112	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
15	GPIO111	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
14	GPIO110	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
13	GPIO109	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
12	GPIO108	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
11	GPIO107	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
10	GPIO106	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
9	GPIO105	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
8	GPIO104	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
7	GPIO103	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
6	GPIO102	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
5	GPIO101	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
4	GPIO100	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
3	GPIO99	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
2	GPIO98	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
1	GPIO97	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
0	GPIO96	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn

15.10.3.59 GPDPUD Register (Offset = CCh) [Reset = FFFFFFFFh]

GPDPUD is shown in Figure 15-62 and described in Table 15-73.

Return to the Summary Table.

GPIO D Pull Up Disable Register (GPIO96 to 127)
Disables the Pull-Up on GPIO.
0: Enables the Pull-Up.
1: Disables the Pull-Up.
Reading the register returns the current value of the register setting.

Figure 15-62. GPDPUD Register

31	30	29	28	27	26	25	24
GPIO127	GPIO126	GPIO125	GPIO124	GPIO123	GPIO122	GPIO121	GPIO120
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
GPIO119	GPIO118	GPIO117	GPIO116	GPIO115	GPIO114	GPIO113	GPIO112
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
GPIO111	GPIO110	GPIO109	GPIO108	GPIO107	GPIO106	GPIO105	GPIO104
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
GPIO103	GPIO102	GPIO101	GPIO100	GPIO99	GPIO98	GPIO97	GPIO96
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 15-73. GPDPUD Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO127	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
30	GPIO126	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
29	GPIO125	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
28	GPIO124	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
27	GPIO123	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
26	GPIO122	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
25	GPIO121	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
24	GPIO120	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
23	GPIO119	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
22	GPIO118	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
21	GPIO117	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
20	GPIO116	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
19	GPIO115	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn

Table 15-73. GPDPUD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO114	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
17	GPIO113	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
16	GPIO112	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
15	GPIO111	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
14	GPIO110	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
13	GPIO109	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
12	GPIO108	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
11	GPIO107	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
10	GPIO106	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
9	GPIO105	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
8	GPIO104	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
7	GPIO103	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
6	GPIO102	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
5	GPIO101	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
4	GPIO100	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
3	GPIO99	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
2	GPIO98	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
1	GPIO97	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
0	GPIO96	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn

15.10.3.60 GPDINV Register (Offset = D0h) [Reset = 0h]

GPDINV is shown in Figure 15-63 and described in Table 15-74.

Return to the Summary Table.

GPIO D Input Polarity Invert Registers (GPIO96 to 127)
Selects between non-inverted and inverted GPIO input to the device.
0: selects non-inverted GPIO input
1: selects inverted GPIO input
Reading the register returns the current value of the register setting.

Figure 15-63. GPDINV Register

31	30	29	28	27	26	25	24
GPIO127	GPIO126	GPIO125	GPIO124	GPIO123	GPIO122	GPIO121	GPIO120
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO119	GPIO118	GPIO117	GPIO116	GPIO115	GPIO114	GPIO113	GPIO112
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO111	GPIO110	GPIO109	GPIO108	GPIO107	GPIO106	GPIO105	GPIO104
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO103	GPIO102	GPIO101	GPIO100	GPIO99	GPIO98	GPIO97	GPIO96
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-74. GPDINV Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO127	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
30	GPIO126	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
29	GPIO125	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
28	GPIO124	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
27	GPIO123	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
26	GPIO122	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
25	GPIO121	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
24	GPIO120	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
23	GPIO119	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
22	GPIO118	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
21	GPIO117	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
20	GPIO116	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
19	GPIO115	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn

Table 15-74. GPDINV Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO114	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
17	GPIO113	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
16	GPIO112	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
15	GPIO111	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
14	GPIO110	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
13	GPIO109	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
12	GPIO108	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
11	GPIO107	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
10	GPIO106	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
9	GPIO105	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
8	GPIO104	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
7	GPIO103	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
6	GPIO102	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
5	GPIO101	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
4	GPIO100	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
3	GPIO99	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
2	GPIO98	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
1	GPIO97	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
0	GPIO96	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn

15.10.3.61 GPDODR Register (Offset = D2h) [Reset = 0h]

GPDODR is shown in Figure 15-64 and described in Table 15-75.

Return to the Summary Table.

GPIO Open Drain Output Register
Selects between normal and open-drain output for the GPIO pin.
0: Normal Output
1: Open Drain Output
CAUTION: Open Drain mode is not supported on this device. Keep this bit as 0. Open drain can be emulated by writing 0 to GPxDAT and enabling or disabling the output buffer with GPxDIR. (Peripherals such as I2C support Open Drain mode without issue).

Figure 15-64. GPDODR Register

31	30	29	28	27	26	25	24
GPIO127	GPIO126	GPIO125	GPIO124	GPIO123	GPIO122	GPIO121	GPIO120
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO119	GPIO118	GPIO117	GPIO116	GPIO115	GPIO114	GPIO113	GPIO112
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO111	GPIO110	GPIO109	GPIO108	GPIO107	GPIO106	GPIO105	GPIO104
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO103	GPIO102	GPIO101	GPIO100	GPIO99	GPIO98	GPIO97	GPIO96
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-75. GPDODR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO127	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
30	GPIO126	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
29	GPIO125	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
28	GPIO124	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
27	GPIO123	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
26	GPIO122	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
25	GPIO121	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
24	GPIO120	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
23	GPIO119	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
22	GPIO118	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
21	GPIO117	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
20	GPIO116	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn

Table 15-75. GPDODR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	GPIO115	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
18	GPIO114	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
17	GPIO113	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
16	GPIO112	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
15	GPIO111	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
14	GPIO110	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
13	GPIO109	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
12	GPIO108	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
11	GPIO107	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
10	GPIO106	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
9	GPIO105	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
8	GPIO104	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
7	GPIO103	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
6	GPIO102	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
5	GPIO101	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
4	GPIO100	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
3	GPIO99	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
2	GPIO98	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
1	GPIO97	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
0	GPIO96	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn

15.10.3.62 GPDGMUX1 Register (Offset = E0h) [Reset = 0h]

GPDGMUX1 is shown in Figure 15-65 and described in Table 15-76.

Return to the Summary Table.

GPIO D Peripheral Group Mux (GPIO96 to 111)
Defines pin-muxing selection for GPIO.
Note: For complete pin-mux selection on GPIOx, GPDMUXy.GPIOx configuration is also required.

Figure 15-65. GPDGMUX1 Register

31	30	29	28	27	26	25	24
GPIO111		GPIO110		GPIO109		GPIO108	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
GPIO107		GPIO106		GPIO105		GPIO104	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO103		GPIO102		GPIO101		GPIO100	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
GPIO99		GPIO98		GPIO97		GPIO96	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-76. GPDGMUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO111	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO110	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO109	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO108	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO107	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO106	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO105	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO104	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO103	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO102	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO101	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO100	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO99	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
5-4	GPIO98	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

Table 15-76. GPDGMUX1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	GPIO97	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO96	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.63 GPDGMUX2 Register (Offset = E2h) [Reset = 0h]

GPDGMUX2 is shown in Figure 15-66 and described in Table 15-77.

Return to the Summary Table.

GPIO D Peripheral Group Mux (GPIO112 to 127)
Defines pin-muxing selection for GPIO.
Note: For complete pin-mux selection on GPIOx, GPDMUXy.GPIOx configuration is also required.

Figure 15-66. GPDGMUX2 Register

31	30	29	28	27	26	25	24
GPIO127		GPIO126		GPIO125		GPIO124	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
GPIO123		GPIO122		GPIO121		GPIO120	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO119		GPIO118		GPIO117		GPIO116	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
GPIO115		GPIO114		GPIO113		GPIO112	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-77. GPDGMUX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO127	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO126	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO125	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO124	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO123	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO122	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO121	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO120	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO119	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO118	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO117	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO116	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO115	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
5-4	GPIO114	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

Table 15-77. GPDGMUX2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	GPIO113	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO112	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.64 GPDCSEL1 Register (Offset = E8h) [Reset = 0h]

GPDCSEL1 is shown in Figure 15-67 and described in Table 15-78.

Return to the Summary Table.

GPIO D Core Select Register (GPIO0 to 7)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPDCSEL[0] is used. Writing to GPDCSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-67. GPDCSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO103				GPIO102				GPIO101				GPIO100			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO99				GPIO98				GPIO97				GPIO96			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-78. GPDCSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO103	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO102	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO101	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO100	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO99	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO98	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO97	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO96	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.65 GPDCSEL2 Register (Offset = EAh) [Reset = 0h]

GPDCSEL2 is shown in Figure 15-68 and described in Table 15-79.

Return to the Summary Table.

GPIO D Core Select Register (GPIO8 to 15)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPDCSEL[0] is used. Writing to GPDCSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-68. GPDCSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO111				GPIO110				GPIO109				GPIO108			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO107				GPIO106				GPIO105				GPIO104			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-79. GPDCSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO111	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO110	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO109	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO108	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO107	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO106	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO105	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO104	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.66 GPDCSEL3 Register (Offset = ECh) [Reset = 0h]

GPDCSEL3 is shown in Figure 15-69 and described in Table 15-80.

Return to the Summary Table.

GPIO D Core Select Register (GPIO16 to 23)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPDCSEL[0] is used. Writing to GPDCSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-69. GPDCSEL3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO119				GPIO118				GPIO117				GPIO116			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO115				GPIO114				GPIO113				GPIO112			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-80. GPDCSEL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO119	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO118	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO117	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO116	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO115	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO114	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO113	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO112	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.67 GPDCSEL4 Register (Offset = EEh) [Reset = 0h]

GPDCSEL4 is shown in Figure 15-70 and described in Table 15-81.

Return to the Summary Table.

GPIO D Core Select Register (GPIO24 to 31)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPDCSEL[0] is used. Writing to GPDCSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-70. GPDCSEL4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO127				GPIO126				GPIO125				GPIO124			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO123				GPIO122				GPIO121				GPIO120			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-81. GPDCSEL4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO127	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO126	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO125	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO124	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO123	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO122	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO121	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO120	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.68 GPDLOCK Register (Offset = FCh) [Reset = 0h]

GPDLOCK is shown in Figure 15-71 and described in Table 15-82.

Return to the Summary Table.

GPIO D Lock Configuration Register (GPIO96 to 127)
GPIO Configuration Lock for GPIO.
0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed
1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin

Figure 15-71. GPDLOCK Register

31	30	29	28	27	26	25	24
GPIO127	GPIO126	GPIO125	GPIO124	GPIO123	GPIO122	GPIO121	GPIO120
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO119	GPIO118	GPIO117	GPIO116	GPIO115	GPIO114	GPIO113	GPIO112
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO111	GPIO110	GPIO109	GPIO108	GPIO107	GPIO106	GPIO105	GPIO104
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO103	GPIO102	GPIO101	GPIO100	GPIO99	GPIO98	GPIO97	GPIO96
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-82. GPDLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO127	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
30	GPIO126	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
29	GPIO125	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
28	GPIO124	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
27	GPIO123	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
26	GPIO122	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
25	GPIO121	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
24	GPIO120	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
23	GPIO119	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
22	GPIO118	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
21	GPIO117	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
20	GPIO116	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn

Table 15-82. GPDLOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	GPIO115	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
18	GPIO114	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
17	GPIO113	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
16	GPIO112	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
15	GPIO111	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
14	GPIO110	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
13	GPIO109	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
12	GPIO108	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
11	GPIO107	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
10	GPIO106	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
9	GPIO105	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
8	GPIO104	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
7	GPIO103	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
6	GPIO102	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
5	GPIO101	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
4	GPIO100	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
3	GPIO99	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
2	GPIO98	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
1	GPIO97	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
0	GPIO96	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn

15.10.3.69 GPDCR Register (Offset = FEh) [Reset = 0h]

GPDCR is shown in Figure 15-72 and described in Table 15-83.

Return to the Summary Table.

GPIO D Lock Commit Register (GPIO96 to 127)
GPIO Configuration Lock Commit for GPIO:
1: Locks changes to the bit in GPyLOCK register which controls the same pin
0: Bit in the GPyLOCK register which controls the same pin can be changed

Figure 15-72. GPDCR Register

31	30	29	28	27	26	25	24
GPIO127	GPIO126	GPIO125	GPIO124	GPIO123	GPIO122	GPIO121	GPIO120
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h
23	22	21	20	19	18	17	16
GPIO119	GPIO118	GPIO117	GPIO116	GPIO115	GPIO114	GPIO113	GPIO112
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h
15	14	13	12	11	10	9	8
GPIO111	GPIO110	GPIO109	GPIO108	GPIO107	GPIO106	GPIO105	GPIO104
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h
7	6	5	4	3	2	1	0
GPIO103	GPIO102	GPIO101	GPIO100	GPIO99	GPIO98	GPIO97	GPIO96
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h

Table 15-83. GPDCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO127	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
30	GPIO126	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
29	GPIO125	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
28	GPIO124	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
27	GPIO123	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
26	GPIO122	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
25	GPIO121	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
24	GPIO120	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
23	GPIO119	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
22	GPIO118	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
21	GPIO117	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
20	GPIO116	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
19	GPIO115	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn

Table 15-83. GPDCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO114	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
17	GPIO113	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
16	GPIO112	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
15	GPIO111	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
14	GPIO110	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
13	GPIO109	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
12	GPIO108	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
11	GPIO107	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
10	GPIO106	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
9	GPIO105	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
8	GPIO104	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
7	GPIO103	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
6	GPIO102	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
5	GPIO101	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
4	GPIO100	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
3	GPIO99	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
2	GPIO98	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
1	GPIO97	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
0	GPIO96	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn

15.10.3.70 GPECTRL Register (Offset = 100h) [Reset = 0h]

GPECTRL is shown in Figure 15-73 and described in Table 15-84.

Return to the Summary Table.

GPIO E Qualification Sampling Period Control (GPIO128 to 159)

Figure 15-73. GPECTRL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUALPRD3								QUALPRD2								QUALPRD1								QUALPRD0							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 15-84. GPECTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	QUALPRD3	R/W	0h	Qualification sampling period for GPIO152 to GPIO159: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn
23-16	QUALPRD2	R/W	0h	Qualification sampling period for GPIO144 to GPIO151: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn
15-8	QUALPRD1	R/W	0h	Qualification sampling period for GPIO136 to GPIO143: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn
7-0	QUALPRD0	R/W	0h	Qualification sampling period for GPIO128 to GPIO135: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn

15.10.3.71 GPEQSEL1 Register (Offset = 102h) [Reset = 0h]

GPEQSEL1 is shown in Figure 15-74 and described in Table 15-85.

Return to the Summary Table.

GPIO E Qualifier Select 1 Register (GPIO128 to 143)
Input qualification type:
0,0 Sync
0,1 Qualification (3 samples)
1,0 Qualification (6 samples)
1,1 Async (no Sync or Qualification)

Figure 15-74. GPEQSEL1 Register

31	30	29	28	27	26	25	24
GPIO143		GPIO142		GPIO141		GPIO140	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
GPIO139		GPIO138		GPIO137		GPIO136	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO135		GPIO134		GPIO133		GPIO132	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
GPIO131		GPIO130		GPIO129		GPIO128	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-85. GPEQSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO143	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
29-28	GPIO142	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
27-26	GPIO141	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
25-24	GPIO140	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
23-22	GPIO139	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
21-20	GPIO138	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
19-18	GPIO137	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
17-16	GPIO136	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
15-14	GPIO135	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
13-12	GPIO134	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
11-10	GPIO133	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
9-8	GPIO132	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn

Table 15-85. GPEQSEL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	GPIO131	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
5-4	GPIO130	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
3-2	GPIO129	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
1-0	GPIO128	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn

15.10.3.72 GPEQSEL2 Register (Offset = 104h) [Reset = 0h]

GPEQSEL2 is shown in Figure 15-75 and described in Table 15-86.

Return to the Summary Table.

GPIO E Qualifier Select 2 Register (GPIO144 to 159)
Input qualification type:
0,0 Sync
0,1 Qualification (3 samples)
1,0 Qualification (6 samples)
1,1 Async (no Sync or Qualification)

Figure 15-75. GPEQSEL2 Register

31	30	29	28	27	26	25	24
GPIO159		GPIO158		GPIO157		GPIO156	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
GPIO155		GPIO154		GPIO153		GPIO152	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO151		GPIO150		GPIO149		GPIO148	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
GPIO147		GPIO146		GPIO145		GPIO144	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-86. GPEQSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO159	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
29-28	GPIO158	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
27-26	GPIO157	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
25-24	GPIO156	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
23-22	GPIO155	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
21-20	GPIO154	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
19-18	GPIO153	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
17-16	GPIO152	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
15-14	GPIO151	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
13-12	GPIO150	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
11-10	GPIO149	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
9-8	GPIO148	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn

Table 15-86. GPEQSEL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	GPIO147	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
5-4	GPIO146	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
3-2	GPIO145	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
1-0	GPIO144	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn

15.10.3.73 GPEMUX1 Register (Offset = 106h) [Reset = 0h]

GPEMUX1 is shown in Figure 15-76 and described in Table 15-87.

Return to the Summary Table.

GPIO E Mux 1 Register (GPIO128 to 143)
Defines pin-muxing selection for GPIO.
Notes:
The respective GPyGMUXn.GPIOz must be configured prior to this register to avoid intermediate peripheral selects being mapped to the GPIO.

Figure 15-76. GPEMUX1 Register

31	30	29	28	27	26	25	24
GPIO143		GPIO142		GPIO141		GPIO140	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
GPIO139		GPIO138		GPIO137		GPIO136	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO135		GPIO134		GPIO133		GPIO132	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
GPIO131		GPIO130		GPIO129		GPIO128	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-87. GPEMUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO143	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO142	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO141	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO140	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO139	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO138	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO137	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO136	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO135	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO134	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO133	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO132	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO131	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

Table 15-87. GPEMUX1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-4	GPIO130	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
3-2	GPIO129	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO128	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.74 GPEMUX2 Register (Offset = 108h) [Reset = 0h]

GPEMUX2 is shown in Figure 15-77 and described in Table 15-88.

Return to the Summary Table.

GPIO E Mux 2 Register (GPIO144 to 159)
Defines pin-muxing selection for GPIO.
Notes:
The respective GPyGMUXn.GPIOz must be configured prior to this register to avoid intermediate peripheral selects being mapped to the GPIO.

Figure 15-77. GPEMUX2 Register

31	30	29	28	27	26	25	24
GPIO159		GPIO158		GPIO157		GPIO156	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
GPIO155		GPIO154		GPIO153		GPIO152	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO151		GPIO150		GPIO149		GPIO148	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
GPIO147		GPIO146		GPIO145		GPIO144	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-88. GPEMUX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO159	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO158	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO157	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO156	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO155	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO154	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO153	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO152	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO151	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO150	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO149	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO148	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO147	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

Table 15-88. GPEMUX2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-4	GPIO146	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
3-2	GPIO145	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO144	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.75 GPEDIR Register (Offset = 10Ah) [Reset = 0h]

GPEDIR is shown in Figure 15-78 and described in Table 15-89.

Return to the Summary Table.

GPIO E Direction Register (GPIO128 to 159)
Controls direction of GPIO pins when the specified pin is configured in GPIO mode.
0: Configures pin as input.
1: Configures pin as output.
Reading the register returns the current value of the register setting.

Figure 15-78. GPEDIR Register

31	30	29	28	27	26	25	24
GPIO159	GPIO158	GPIO157	GPIO156	GPIO155	GPIO154	GPIO153	GPIO152
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO151	GPIO150	GPIO149	GPIO148	GPIO147	GPIO146	GPIO145	GPIO144
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO143	GPIO142	GPIO141	GPIO140	GPIO139	GPIO138	GPIO137	GPIO136
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO135	GPIO134	GPIO133	GPIO132	GPIO131	GPIO130	GPIO129	GPIO128
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-89. GPEDIR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO159	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
30	GPIO158	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
29	GPIO157	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
28	GPIO156	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
27	GPIO155	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
26	GPIO154	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
25	GPIO153	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
24	GPIO152	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
23	GPIO151	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
22	GPIO150	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
21	GPIO149	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
20	GPIO148	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
19	GPIO147	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn

Table 15-89. GPEDIR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO146	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
17	GPIO145	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
16	GPIO144	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
15	GPIO143	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
14	GPIO142	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
13	GPIO141	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
12	GPIO140	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
11	GPIO139	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
10	GPIO138	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
9	GPIO137	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
8	GPIO136	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
7	GPIO135	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
6	GPIO134	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
5	GPIO133	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
4	GPIO132	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
3	GPIO131	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
2	GPIO130	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
1	GPIO129	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
0	GPIO128	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn

15.10.3.76 GPEPUD Register (Offset = 10Ch) [Reset = FFFFFFFFh]

GPEPUD is shown in Figure 15-79 and described in Table 15-90.

Return to the Summary Table.

GPIO E Pull Up Disable Register (GPIO128 to 159)
Disables the Pull-Up on GPIO.
0: Enables the Pull-Up.
1: Disables the Pull-Up.
Reading the register returns the current value of the register setting.

Figure 15-79. GPEPUD Register

31	30	29	28	27	26	25	24
GPIO159	GPIO158	GPIO157	GPIO156	GPIO155	GPIO154	GPIO153	GPIO152
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
GPIO151	GPIO150	GPIO149	GPIO148	GPIO147	GPIO146	GPIO145	GPIO144
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
GPIO143	GPIO142	GPIO141	GPIO140	GPIO139	GPIO138	GPIO137	GPIO136
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
GPIO135	GPIO134	GPIO133	GPIO132	GPIO131	GPIO130	GPIO129	GPIO128
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 15-90. GPEPUD Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO159	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
30	GPIO158	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
29	GPIO157	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
28	GPIO156	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
27	GPIO155	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
26	GPIO154	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
25	GPIO153	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
24	GPIO152	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
23	GPIO151	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
22	GPIO150	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
21	GPIO149	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
20	GPIO148	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
19	GPIO147	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn

Table 15-90. GPEPUD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO146	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
17	GPIO145	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
16	GPIO144	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
15	GPIO143	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
14	GPIO142	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
13	GPIO141	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
12	GPIO140	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
11	GPIO139	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
10	GPIO138	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
9	GPIO137	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
8	GPIO136	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
7	GPIO135	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
6	GPIO134	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
5	GPIO133	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
4	GPIO132	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
3	GPIO131	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
2	GPIO130	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
1	GPIO129	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
0	GPIO128	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn

15.10.3.77 GPEINV Register (Offset = 110h) [Reset = 0h]

GPEINV is shown in Figure 15-80 and described in Table 15-91.

Return to the Summary Table.

GPIO E Input Polarity Invert Registers (GPIO128 to 159)
Selects between non-inverted and inverted GPIO input to the device.
0: selects non-inverted GPIO input
1: selects inverted GPIO input
Reading the register returns the current value of the register setting.

Figure 15-80. GPEINV Register

31	30	29	28	27	26	25	24
GPIO159	GPIO158	GPIO157	GPIO156	GPIO155	GPIO154	GPIO153	GPIO152
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO151	GPIO150	GPIO149	GPIO148	GPIO147	GPIO146	GPIO145	GPIO144
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO143	GPIO142	GPIO141	GPIO140	GPIO139	GPIO138	GPIO137	GPIO136
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO135	GPIO134	GPIO133	GPIO132	GPIO131	GPIO130	GPIO129	GPIO128
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-91. GPEINV Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO159	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
30	GPIO158	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
29	GPIO157	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
28	GPIO156	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
27	GPIO155	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
26	GPIO154	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
25	GPIO153	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
24	GPIO152	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
23	GPIO151	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
22	GPIO150	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
21	GPIO149	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
20	GPIO148	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
19	GPIO147	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn

Table 15-91. GPEINV Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO146	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
17	GPIO145	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
16	GPIO144	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
15	GPIO143	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
14	GPIO142	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
13	GPIO141	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
12	GPIO140	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
11	GPIO139	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
10	GPIO138	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
9	GPIO137	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
8	GPIO136	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
7	GPIO135	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
6	GPIO134	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
5	GPIO133	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
4	GPIO132	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
3	GPIO131	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
2	GPIO130	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
1	GPIO129	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
0	GPIO128	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn

15.10.3.78 GPEODR Register (Offset = 112h) [Reset = 0h]

GPEODR is shown in Figure 15-81 and described in Table 15-92.

Return to the Summary Table.

GPIO Open Drain Output Register
Selects between normal and open-drain output for the GPIO pin.
0: Normal Output
1: Open Drain Output
CAUTION: Open Drain mode is not supported on this device. Keep this bit as 0. Open drain can be emulated by writing 0 to GPxDAT and enabling or disabling the output buffer with GPxDIR. (Peripherals such as I2C support Open Drain mode without issue).

Figure 15-81. GPEODR Register

31	30	29	28	27	26	25	24
GPIO159	GPIO158	GPIO157	GPIO156	GPIO155	GPIO154	GPIO153	GPIO152
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO151	GPIO150	GPIO149	GPIO148	GPIO147	GPIO146	GPIO145	GPIO144
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO143	GPIO142	GPIO141	GPIO140	GPIO139	GPIO138	GPIO137	GPIO136
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO135	GPIO134	GPIO133	GPIO132	GPIO131	GPIO130	GPIO129	GPIO128
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-92. GPEODR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO159	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
30	GPIO158	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
29	GPIO157	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
28	GPIO156	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
27	GPIO155	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
26	GPIO154	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
25	GPIO153	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
24	GPIO152	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
23	GPIO151	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
22	GPIO150	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
21	GPIO149	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
20	GPIO148	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn

Table 15-92. GPEODR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	GPIO147	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
18	GPIO146	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
17	GPIO145	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
16	GPIO144	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
15	GPIO143	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
14	GPIO142	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
13	GPIO141	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
12	GPIO140	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
11	GPIO139	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
10	GPIO138	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
9	GPIO137	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
8	GPIO136	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
7	GPIO135	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
6	GPIO134	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
5	GPIO133	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
4	GPIO132	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
3	GPIO131	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
2	GPIO130	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
1	GPIO129	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
0	GPIO128	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn

15.10.3.79 GPEGMUX1 Register (Offset = 120h) [Reset = 0h]

GPEGMUX1 is shown in Figure 15-82 and described in Table 15-93.

Return to the Summary Table.

GPIO E Peripheral Group Mux (GPIO128 to 143)
Defines pin-muxing selection for GPIO.
Note: For complete pin-mux selection on GPIOx, GPEMUXy.GPIOx configuration is also required.

Figure 15-82. GPEGMUX1 Register

31	30	29	28	27	26	25	24
GPIO143		GPIO142		GPIO141		GPIO140	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
GPIO139		GPIO138		GPIO137		GPIO136	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO135		GPIO134		GPIO133		GPIO132	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
GPIO131		GPIO130		GPIO129		GPIO128	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-93. GPEGMUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO143	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO142	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO141	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO140	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO139	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO138	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO137	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO136	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO135	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO134	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO133	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO132	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO131	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
5-4	GPIO130	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

Table 15-93. GPEGMUX1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	GPIO129	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO128	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.80 GPEGMUX2 Register (Offset = 122h) [Reset = 0h]

GPEGMUX2 is shown in Figure 15-83 and described in Table 15-94.

Return to the Summary Table.

GPIO E Peripheral Group Mux (GPIO144 to 159)
Defines pin-muxing selection for GPIO.
Note: For complete pin-mux selection on GPIOx, GPEMUXy.GPIOx configuration is also required.

Figure 15-83. GPEGMUX2 Register

31	30	29	28	27	26	25	24
GPIO159		GPIO158		GPIO157		GPIO156	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
GPIO155		GPIO154		GPIO153		GPIO152	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO151		GPIO150		GPIO149		GPIO148	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
GPIO147		GPIO146		GPIO145		GPIO144	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-94. GPEGMUX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO159	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
29-28	GPIO158	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
27-26	GPIO157	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
25-24	GPIO156	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
23-22	GPIO155	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
21-20	GPIO154	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
19-18	GPIO153	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
17-16	GPIO152	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO151	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO150	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO149	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO148	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO147	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
5-4	GPIO146	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

Table 15-94. GPEGMUX2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	GPIO145	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO144	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.81 GPECSEL1 Register (Offset = 128h) [Reset = 0h]

GPECSEL1 is shown in Figure 15-84 and described in Table 15-95.

Return to the Summary Table.

GPIO E Core Select Register (GPIO0 to 7)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPECSEL[0] is used. Writing to GPECSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-84. GPECSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO135				GPIO134				GPIO133				GPIO132			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO131				GPIO130				GPIO129				GPIO128			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-95. GPECSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO135	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO134	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO133	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO132	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO131	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO130	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO129	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO128	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.82 GPECSEL2 Register (Offset = 12Ah) [Reset = 0h]

GPECSEL2 is shown in Figure 15-85 and described in Table 15-96.

Return to the Summary Table.

GPIO E Core Select Register (GPIO8 to 15)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPECSEL[0] is used. Writing to GPECSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-85. GPECSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO143				GPIO142				GPIO141				GPIO140			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO139				GPIO138				GPIO137				GPIO136			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-96. GPECSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO143	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO142	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO141	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO140	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO139	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO138	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO137	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO136	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.83 GPECSEL3 Register (Offset = 12Ch) [Reset = 0h]

GPECSEL3 is shown in Figure 15-86 and described in Table 15-97.

Return to the Summary Table.

GPIO E Core Select Register (GPIO16 to 23)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPECSEL[0] is used. Writing to GPECSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-86. GPECSEL3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO151				GPIO150				GPIO149				GPIO148			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO147				GPIO146				GPIO145				GPIO144			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-97. GPECSEL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO151	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO150	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO149	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO148	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO147	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO146	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO145	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO144	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.84 GPECSEL4 Register (Offset = 12Eh) [Reset = 0h]

GPECSEL4 is shown in Figure 15-87 and described in Table 15-98.

Return to the Summary Table.

GPIO E Core Select Register (GPIO24 to 31)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPECSEL[0] is used. Writing to GPECSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-87. GPECSEL4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO159				GPIO158				GPIO157				GPIO156			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO155				GPIO154				GPIO153				GPIO152			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-98. GPECSEL4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO159	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO158	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO157	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO156	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO155	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO154	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO153	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO152	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.85 GPELOCK Register (Offset = 13Ch) [Reset = 0h]

GPELOCK is shown in Figure 15-88 and described in Table 15-99.

Return to the Summary Table.

GPIO E Lock Configuration Register (GPIO128 to 159)
GPIO Configuration Lock for GPIO.
0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed
1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin

Figure 15-88. GPELOCK Register

31	30	29	28	27	26	25	24
GPIO159	GPIO158	GPIO157	GPIO156	GPIO155	GPIO154	GPIO153	GPIO152
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO151	GPIO150	GPIO149	GPIO148	GPIO147	GPIO146	GPIO145	GPIO144
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO143	GPIO142	GPIO141	GPIO140	GPIO139	GPIO138	GPIO137	GPIO136
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO135	GPIO134	GPIO133	GPIO132	GPIO131	GPIO130	GPIO129	GPIO128
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-99. GPELOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO159	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
30	GPIO158	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
29	GPIO157	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
28	GPIO156	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
27	GPIO155	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
26	GPIO154	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
25	GPIO153	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
24	GPIO152	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
23	GPIO151	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
22	GPIO150	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
21	GPIO149	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
20	GPIO148	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn

Table 15-99. GPELOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	GPIO147	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
18	GPIO146	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
17	GPIO145	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
16	GPIO144	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
15	GPIO143	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
14	GPIO142	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
13	GPIO141	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
12	GPIO140	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
11	GPIO139	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
10	GPIO138	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
9	GPIO137	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
8	GPIO136	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
7	GPIO135	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
6	GPIO134	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
5	GPIO133	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
4	GPIO132	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
3	GPIO131	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
2	GPIO130	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
1	GPIO129	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
0	GPIO128	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn

15.10.3.86 GPECR Register (Offset = 13Eh) [Reset = 0h]

GPECR is shown in Figure 15-89 and described in Table 15-100.

Return to the Summary Table.

GPIO E Lock Commit Register (GPIO128 to 159)
GPIO Configuration Lock Commit for GPIO:
1: Locks changes to the bit in GPyLOCK register which controls the same pin
0: Bit in the GPyLOCK register which controls the same pin can be changed

Figure 15-89. GPECR Register

31	30	29	28	27	26	25	24
GPIO159	GPIO158	GPIO157	GPIO156	GPIO155	GPIO154	GPIO153	GPIO152
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h
23	22	21	20	19	18	17	16
GPIO151	GPIO150	GPIO149	GPIO148	GPIO147	GPIO146	GPIO145	GPIO144
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h
15	14	13	12	11	10	9	8
GPIO143	GPIO142	GPIO141	GPIO140	GPIO139	GPIO138	GPIO137	GPIO136
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h
7	6	5	4	3	2	1	0
GPIO135	GPIO134	GPIO133	GPIO132	GPIO131	GPIO130	GPIO129	GPIO128
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h

Table 15-100. GPECR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO159	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
30	GPIO158	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
29	GPIO157	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
28	GPIO156	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
27	GPIO155	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
26	GPIO154	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
25	GPIO153	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
24	GPIO152	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
23	GPIO151	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
22	GPIO150	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
21	GPIO149	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
20	GPIO148	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
19	GPIO147	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn

Table 15-100. GPECR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO146	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
17	GPIO145	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
16	GPIO144	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
15	GPIO143	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
14	GPIO142	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
13	GPIO141	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
12	GPIO140	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
11	GPIO139	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
10	GPIO138	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
9	GPIO137	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
8	GPIO136	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
7	GPIO135	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
6	GPIO134	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
5	GPIO133	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
4	GPIO132	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
3	GPIO131	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
2	GPIO130	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
1	GPIO129	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
0	GPIO128	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn

15.10.3.87 GPFCTRL Register (Offset = 140h) [Reset = 0h]

GPFCTRL is shown in Figure 15-90 and described in Table 15-101.

Return to the Summary Table.

GPIO F Qualification Sampling Period Control (GPIO160 to 168)

Figure 15-90. GPFCTRL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RESERVED								QUALPRD1								QUALPRD0							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 15-101. GPFCTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R/W	0h	Reserved
23-16	RESERVED	R/W	0h	Reserved
15-8	QUALPRD1	R/W	0h	Qualification sampling period for GPIO168: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn
7-0	QUALPRD0	R/W	0h	Qualification sampling period for GPIO160 to GPIO167: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: CPU1.SYSRSn

15.10.3.88 GPFQSEL1 Register (Offset = 142h) [Reset = 0h]

GPFQSEL1 is shown in Figure 15-91 and described in Table 15-102.

Return to the Summary Table.

GPIO F Qualifier Select 1 Register (GPIO160 to 168)
Input qualification type:
0,0 Sync
0,1 Qualification (3 samples)
1,0 Qualification (6 samples)
1,1 Async (no Sync or Qualification)

Figure 15-91. GPFQSEL1 Register

31	30	29	28	27	26	25	24
RESERVED		RESERVED		RESERVED		RESERVED	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
RESERVED		RESERVED		RESERVED		GPIO168	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO167		GPIO166		GPIO165		GPIO164	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
GPIO163		GPIO162		GPIO161		GPIO160	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-102. GPFQSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R/W	0h	Reserved
29-28	RESERVED	R/W	0h	Reserved
27-26	RESERVED	R/W	0h	Reserved
25-24	RESERVED	R/W	0h	Reserved
23-22	RESERVED	R/W	0h	Reserved
21-20	RESERVED	R/W	0h	Reserved
19-18	RESERVED	R/W	0h	Reserved
17-16	GPIO168	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
15-14	GPIO167	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
13-12	GPIO166	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
11-10	GPIO165	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
9-8	GPIO164	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
7-6	GPIO163	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
5-4	GPIO162	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn
3-2	GPIO161	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn

Table 15-102. GPFQSEL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	GPIO160	R/W	0h	Input qualification type Reset type: CPU1.SYSRSn

15.10.3.89 GPFMUX1 Register (Offset = 146h) [Reset = 0h]

GPFMUX1 is shown in Figure 15-92 and described in Table 15-103.

Return to the Summary Table.

GPIO F Mux 1 Register (GPIO160 to 168)
Defines pin-muxing selection for GPIO.
Notes:
The respective GPyGMUXn.GPIOz must be configured prior to this register to avoid intermediate peripheral selects being mapped to the GPIO.

Figure 15-92. GPFMUX1 Register

31	30	29	28	27	26	25	24
RESERVED		RESERVED		RESERVED		RESERVED	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
RESERVED		RESERVED		RESERVED		GPIO168	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO167		GPIO166		GPIO165		GPIO164	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
GPIO163		GPIO162		GPIO161		GPIO160	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-103. GPFMUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R/W	0h	Reserved
29-28	RESERVED	R/W	0h	Reserved
27-26	RESERVED	R/W	0h	Reserved
25-24	RESERVED	R/W	0h	Reserved
23-22	RESERVED	R/W	0h	Reserved
21-20	RESERVED	R/W	0h	Reserved
19-18	RESERVED	R/W	0h	Reserved
17-16	GPIO168	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO167	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO166	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO165	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO164	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO163	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
5-4	GPIO162	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
3-2	GPIO161	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

Table 15-103. GPFMUX1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	GPIO160	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.90 GPFDIR Register (Offset = 14Ah) [Reset = 0h]

GPFDIR is shown in Figure 15-93 and described in Table 15-104.

Return to the Summary Table.

GPIO F Direction Register (GPIO160 to 168)
Controls direction of GPIO pins when the specified pin is configured in GPIO mode.
0: Configures pin as input.
1: Configures pin as output.
Reading the register returns the current value of the register setting.

Figure 15-93. GPFDIR Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	GPIO168
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO167	GPIO166	GPIO165	GPIO164	GPIO163	GPIO162	GPIO161	GPIO160
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-104. GPFDIR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	RESERVED	R/W	0h	Reserved
28	RESERVED	R/W	0h	Reserved
27	RESERVED	R/W	0h	Reserved
26	RESERVED	R/W	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23	RESERVED	R/W	0h	Reserved
22	RESERVED	R/W	0h	Reserved
21	RESERVED	R/W	0h	Reserved
20	RESERVED	R/W	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved

Table 15-104. GPFDIR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	RESERVED	R/W	0h	Reserved
8	GPIO168	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
7	GPIO167	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
6	GPIO166	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
5	GPIO165	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
4	GPIO164	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
3	GPIO163	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
2	GPIO162	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
1	GPIO161	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn
0	GPIO160	R/W	0h	Defines direction for this pin in GPIO mode Reset type: CPU1.SYSRSn

15.10.3.91 GPFPUD Register (Offset = 14Ch) [Reset = FFFFFFFFh]

GPFPUD is shown in Figure 15-94 and described in Table 15-105.

Return to the Summary Table.

GPIO F Pull Up Disable Register (GPIO160 to 168)
Disables the Pull-Up on GPIO.
0: Enables the Pull-Up.
1: Disables the Pull-Up.
Reading the register returns the current value of the register setting.

Figure 15-94. GPFPUD Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	GPIO168
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
GPIO167	GPIO166	GPIO165	GPIO164	GPIO163	GPIO162	GPIO161	GPIO160
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 15-105. GPFPUD Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	1h	Reserved
30	RESERVED	R/W	1h	Reserved
29	RESERVED	R/W	1h	Reserved
28	RESERVED	R/W	1h	Reserved
27	RESERVED	R/W	1h	Reserved
26	RESERVED	R/W	1h	Reserved
25	RESERVED	R/W	1h	Reserved
24	RESERVED	R/W	1h	Reserved
23	RESERVED	R/W	1h	Reserved
22	RESERVED	R/W	1h	Reserved
21	RESERVED	R/W	1h	Reserved
20	RESERVED	R/W	1h	Reserved
19	RESERVED	R/W	1h	Reserved
18	RESERVED	R/W	1h	Reserved
17	RESERVED	R/W	1h	Reserved
16	RESERVED	R/W	1h	Reserved
15	RESERVED	R/W	1h	Reserved
14	RESERVED	R/W	1h	Reserved
13	RESERVED	R/W	1h	Reserved
12	RESERVED	R/W	1h	Reserved
11	RESERVED	R/W	1h	Reserved
10	RESERVED	R/W	1h	Reserved

Table 15-105. GPFPUD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	RESERVED	R/W	1h	Reserved
8	GPIO168	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
7	GPIO167	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
6	GPIO166	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
5	GPIO165	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
4	GPIO164	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
3	GPIO163	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
2	GPIO162	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
1	GPIO161	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn
0	GPIO160	R/W	1h	Pull-Up Disable control for this pin Reset type: CPU1.SYSRSn

15.10.3.92 GPFINV Register (Offset = 150h) [Reset = 0h]

GPFINV is shown in Figure 15-95 and described in Table 15-106.

Return to the Summary Table.

GPIO F Input Polarity Invert Registers (GPIO160 to 168)
Selects between non-inverted and inverted GPIO input to the device.
0: selects non-inverted GPIO input
1: selects inverted GPIO input
Reading the register returns the current value of the register setting.

Figure 15-95. GPFINV Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	GPIO168
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO167	GPIO166	GPIO165	GPIO164	GPIO163	GPIO162	GPIO161	GPIO160
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-106. GPFINV Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	RESERVED	R/W	0h	Reserved
28	RESERVED	R/W	0h	Reserved
27	RESERVED	R/W	0h	Reserved
26	RESERVED	R/W	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23	RESERVED	R/W	0h	Reserved
22	RESERVED	R/W	0h	Reserved
21	RESERVED	R/W	0h	Reserved
20	RESERVED	R/W	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved

Table 15-106. GPFINV Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	RESERVED	R/W	0h	Reserved
8	GPIO168	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
7	GPIO167	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
6	GPIO166	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
5	GPIO165	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
4	GPIO164	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
3	GPIO163	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
2	GPIO162	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
1	GPIO161	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn
0	GPIO160	R/W	0h	Input inversion control for this pin Reset type: CPU1.SYSRSn

15.10.3.93 GPFODR Register (Offset = 152h) [Reset = 0h]

GPFODR is shown in Figure 15-96 and described in Table 15-107.

Return to the Summary Table.

GPIO Open Drain Output Register
Selects between normal and open-drain output for the GPIO pin.
0: Normal Output
1: Open Drain Output
CAUTION: Open Drain mode is not supported on this device. Keep this bit as 0. Open drain can be emulated by writing 0 to GPxDAT and enabling or disabling the output buffer with GPxDIR. (Peripherals such as I2C support Open Drain mode without issue).

Figure 15-96. GPFODR Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	GPIO168
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO167	GPIO166	GPIO165	GPIO164	GPIO163	GPIO162	GPIO161	GPIO160
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-107. GPFODR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	RESERVED	R/W	0h	Reserved
28	RESERVED	R/W	0h	Reserved
27	RESERVED	R/W	0h	Reserved
26	RESERVED	R/W	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23	RESERVED	R/W	0h	Reserved
22	RESERVED	R/W	0h	Reserved
21	RESERVED	R/W	0h	Reserved
20	RESERVED	R/W	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved

Table 15-107. GPFODR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved
8	GPIO168	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
7	GPIO167	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
6	GPIO166	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
5	GPIO165	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
4	GPIO164	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
3	GPIO163	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
2	GPIO162	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
1	GPIO161	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn
0	GPIO160	R/W	0h	Output Open-Drain control for this pin Reset type: CPU1.SYSRSn

15.10.3.94 GPFGMUX1 Register (Offset = 160h) [Reset = 0h]

GPFGMUX1 is shown in Figure 15-97 and described in Table 15-108.

Return to the Summary Table.

GPIO F Peripheral Group Mux (GPIO160 to 168)
Defines pin-muxing selection for GPIO.
Note: For complete pin-mux selection on GPIOx, GPFMUXy.GPIOx configuration is also required.

Figure 15-97. GPFGMUX1 Register

31	30	29	28	27	26	25	24
RESERVED		RESERVED		RESERVED		RESERVED	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
RESERVED		RESERVED		RESERVED		GPIO168	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO167		GPIO166		GPIO165		GPIO164	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
GPIO163		GPIO162		GPIO161		GPIO160	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 15-108. GPFGMUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R/W	0h	Reserved
29-28	RESERVED	R/W	0h	Reserved
27-26	RESERVED	R/W	0h	Reserved
25-24	RESERVED	R/W	0h	Reserved
23-22	RESERVED	R/W	0h	Reserved
21-20	RESERVED	R/W	0h	Reserved
19-18	RESERVED	R/W	0h	Reserved
17-16	GPIO168	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
15-14	GPIO167	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
13-12	GPIO166	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
11-10	GPIO165	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
9-8	GPIO164	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
7-6	GPIO163	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
5-4	GPIO162	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
3-2	GPIO161	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn
1-0	GPIO160	R/W	0h	Defines pin-muxing selection for GPIO Reset type: CPU1.SYSRSn

15.10.3.95 GPFCSEL1 Register (Offset = 168h) [Reset = 0h]

GPFCSEL1 is shown in Figure 15-98 and described in Table 15-109.

Return to the Summary Table.

GPIO F Core Select Register (GPIO0 to 7)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPFCSEL[0] is used. Writing to GPFCSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-98. GPFCSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO167				GPIO166				GPIO165				GPIO164			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO163				GPIO162				GPIO161				GPIO160			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-109. GPFCSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO167	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
27-24	GPIO166	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
23-20	GPIO165	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
19-16	GPIO164	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
15-12	GPIO163	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
11-8	GPIO162	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
7-4	GPIO161	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn
3-0	GPIO160	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.96 GPFCSEL2 Register (Offset = 16Ah) [Reset = 0h]

GPFCSEL2 is shown in Figure 15-99 and described in Table 15-110.

Return to the Summary Table.

GPIO F Core Select Register (GPIO8 to 15)
Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected
0011: CPU2.CLA1 selected
0100: CM selected
In single Core systems only GPFCSEL[0] is used. Writing to GPFCSEL[1] does not have effect and it will defailt to corresponding CPU or CLA control selection.

Figure 15-99. GPFCSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				RESERVED				RESERVED				RESERVED			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				RESERVED				RESERVED				GPIO168			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 15-110. GPFCSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	Reserved
27-24	RESERVED	R/W	0h	Reserved
23-20	RESERVED	R/W	0h	Reserved
19-16	RESERVED	R/W	0h	Reserved
15-12	RESERVED	R/W	0h	Reserved
11-8	RESERVED	R/W	0h	Reserved
7-4	RESERVED	R/W	0h	Reserved
3-0	GPIO168	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: CPU1.SYSRSn

15.10.3.97 GPFLOCK Register (Offset = 17Ch) [Reset = 0h]

GPFLOCK is shown in Figure 15-100 and described in Table 15-111.

Return to the Summary Table.

GPIO F Lock Configuration Register (GPIO160 to 168)
GPIO Configuration Lock for GPIO.
0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed
1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin

Figure 15-100. GPFLOCK Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	GPIO168
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO167	GPIO166	GPIO165	GPIO164	GPIO163	GPIO162	GPIO161	GPIO160
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-111. GPFLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	RESERVED	R/W	0h	Reserved
28	RESERVED	R/W	0h	Reserved
27	RESERVED	R/W	0h	Reserved
26	RESERVED	R/W	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23	RESERVED	R/W	0h	Reserved
22	RESERVED	R/W	0h	Reserved
21	RESERVED	R/W	0h	Reserved
20	RESERVED	R/W	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved

Table 15-111. GPFLOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved
8	GPIO168	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
7	GPIO167	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
6	GPIO166	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
5	GPIO165	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
4	GPIO164	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
3	GPIO163	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
2	GPIO162	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
1	GPIO161	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn
0	GPIO160	R/W	0h	Configuration Lock bit for this pin Reset type: CPU1.SYSRSn

15.10.3.98 GPFCR Register (Offset = 17Eh) [Reset = 0h]

GPFCR is shown in Figure 15-101 and described in Table 15-112.

Return to the Summary Table.

GPIO F Lock Commit Register (GPIO160 to 168)
GPIO Configuration Lock Commit for GPIO:
1: Locks changes to the bit in GPyLOCK register which controls the same pin
0: Bit in the GPyLOCK register which controls the same pin can be changed

Figure 15-101. GPFCR Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h
23	22	21	20	19	18	17	16
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	GPIO168
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h
7	6	5	4	3	2	1	0
GPIO167	GPIO166	GPIO165	GPIO164	GPIO163	GPIO162	GPIO161	GPIO160
R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h

Table 15-112. GPFCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/WOnce	0h	Reserved
30	RESERVED	R/WOnce	0h	Reserved
29	RESERVED	R/WOnce	0h	Reserved
28	RESERVED	R/WOnce	0h	Reserved
27	RESERVED	R/WOnce	0h	Reserved
26	RESERVED	R/WOnce	0h	Reserved
25	RESERVED	R/WOnce	0h	Reserved
24	RESERVED	R/WOnce	0h	Reserved
23	RESERVED	R/WOnce	0h	Reserved
22	RESERVED	R/WOnce	0h	Reserved
21	RESERVED	R/WOnce	0h	Reserved
20	RESERVED	R/WOnce	0h	Reserved
19	RESERVED	R/WOnce	0h	Reserved
18	RESERVED	R/WOnce	0h	Reserved
17	RESERVED	R/WOnce	0h	Reserved
16	RESERVED	R/WOnce	0h	Reserved
15	RESERVED	R/WOnce	0h	Reserved
14	RESERVED	R/WOnce	0h	Reserved
13	RESERVED	R/WOnce	0h	Reserved
12	RESERVED	R/WOnce	0h	Reserved
11	RESERVED	R/WOnce	0h	Reserved
10	RESERVED	R/WOnce	0h	Reserved
9	RESERVED	R/WOnce	0h	Reserved

Table 15-112. GPFCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	GPIO168	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
7	GPIO167	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
6	GPIO166	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
5	GPIO165	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
4	GPIO164	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
3	GPIO163	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
2	GPIO162	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
1	GPIO161	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn
0	GPIO160	R/WOnce	0h	Configuration lock commit bit for this pin Reset type: CPU1.SYSRSn

15.10.4 GPIO_DATA_REGS Registers

Table 15-113 lists the memory-mapped registers for the GPIO_DATA_REGS registers. All register offset addresses not listed in Table 15-113 should be considered as reserved locations and the register contents should not be modified.

Table 15-113. GPIO_DATA_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	GPADAT	GPIO A Data Register (GPIO0 to 31)		Go
2h	GPASET	GPIO A Data Set Register (GPIO0 to 31)		Go
4h	GPACLEAR	GPIO A Data Clear Register (GPIO0 to 31)		Go
6h	GPATOGGLE	GPIO A Data Toggle Register (GPIO0 to 31)		Go
8h	GPBDAT	GPIO B Data Register (GPIO32 to 63)		Go
Ah	GPBSET	GPIO B Data Set Register (GPIO32 to 63)		Go
Ch	GPBCLEAR	GPIO B Data Clear Register (GPIO32 to 63)		Go
Eh	GPBTOGGLE	GPIO B Data Toggle Register (GPIO32 to 63)		Go
10h	GPCDAT	GPIO C Data Register (GPIO64 to 95)		Go
12h	GPCSET	GPIO C Data Set Register (GPIO64 to 95)		Go
14h	GPCCLEAR	GPIO C Data Clear Register (GPIO64 to 95)		Go
16h	GPCTOGGLE	GPIO C Data Toggle Register (GPIO64 to 95)		Go
18h	GPDDAT	GPIO D Data Register (GPIO96 to 127)		Go
1Ah	GPDSET	GPIO D Data Set Register (GPIO96 to 127)		Go
1Ch	GPDCLEAR	GPIO D Data Clear Register (GPIO96 to 127)		Go
1Eh	GPDTOGGLE	GPIO D Data Toggle Register (GPIO96 to 127)		Go
20h	GPEDAT	GPIO E Data Register (GPIO128 to 159)		Go
22h	GPESET	GPIO E Data Set Register (GPIO128 to 159)		Go
24h	GPECLEAR	GPIO E Data Clear Register (GPIO128 to 159)		Go
26h	GPETOGGLE	GPIO E Data Toggle Register (GPIO128 to 159)		Go
28h	GPFDAT	GPIO F Data Register (GPIO160 to 168)		Go
2Ah	GPFSET	GPIO F Data Set Register (GPIO160 to 168)		Go
2Ch	GPFCLEAR	GPIO F Data Clear Register (GPIO160 to 168)		Go
2Eh	GPFTOGGLE	GPIO F Data Toggle Register (GPIO160 to 168)		Go

Complex bit access types are encoded to fit into small table cells. Table 15-114 shows the codes that are used for access types in this section.

Table 15-114. GPIO_DATA_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 15-114. GPIO_DATA_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

15.10.4.1 GPADAT Register (Offset = 0h) [Reset = 0h]

GPADAT is shown in Figure 15-102 and described in Table 15-115.

Return to the Summary Table.

GPIO A Data Register (GPIO0 to 31)
Reading this register reflects the current state of the GPIO pin regardless of which mode the GPIO is in.
Writing to this register will set the GPIO pin high or low if the pin is enabled for GPIO output mode. If the GPIO is not in output mode the value written is latched but will not be reflected on the GPIO pin or reads of the GPxDAT register. The written value latched will become active when the GPIO is put into GPIO Output mode
A system reset will clear all bits and latched values to zero.
NOTE: Bit-wise read-modify-write operations should not be performed on this register. For bit-wise operations the GPxSET, GPxCLEAR, or GPxTOGGLE registers should be used instead. If direct writes to GPxDAT are necessary, the entire register should be written at one time.

Figure 15-102. GPADAT Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-115. GPADAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
30	GPIO30	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
29	GPIO29	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
28	GPIO28	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
27	GPIO27	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
26	GPIO26	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
25	GPIO25	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
24	GPIO24	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
23	GPIO23	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
22	GPIO22	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
21	GPIO21	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn

Table 15-115. GPADAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	GPIO20	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
19	GPIO19	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
18	GPIO18	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
17	GPIO17	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
16	GPIO16	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
15	GPIO15	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
14	GPIO14	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
13	GPIO13	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
12	GPIO12	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
11	GPIO11	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
10	GPIO10	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
9	GPIO9	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
8	GPIO8	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
7	GPIO7	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
6	GPIO6	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
5	GPIO5	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
4	GPIO4	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
3	GPIO3	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
2	GPIO2	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
1	GPIO1	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
0	GPIO0	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn

15.10.4.2 GPASET Register (Offset = 2h) [Reset = 0h]

GPASET is shown in Figure 15-103 and described in Table 15-116.

Return to the Summary Table.

GPIO A Data Set Register (GPIO0 to 31)
Writing a 1 will force GPIO output data latch to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-103. GPASET Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-116. GPASET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
30	GPIO30	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
29	GPIO29	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
28	GPIO28	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
27	GPIO27	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
26	GPIO26	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
25	GPIO25	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
24	GPIO24	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
23	GPIO23	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
22	GPIO22	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
21	GPIO21	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
20	GPIO20	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
19	GPIO19	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn

Table 15-116. GPASET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO18	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
17	GPIO17	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
16	GPIO16	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
15	GPIO15	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
14	GPIO14	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
13	GPIO13	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
12	GPIO12	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
11	GPIO11	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
10	GPIO10	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
9	GPIO9	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
8	GPIO8	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
7	GPIO7	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
6	GPIO6	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
5	GPIO5	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
4	GPIO4	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
3	GPIO3	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
2	GPIO2	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
1	GPIO1	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
0	GPIO0	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn

15.10.4.3 GPACLEAR Register (Offset = 4h) [Reset = 0h]

GPACLEAR is shown in Figure 15-104 and described in Table 15-117.

Return to the Summary Table.

GPIO A Data Clear Register (GPIO0 to 31)
Writing a 1 will force GPIO0 output data latch to 0.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-104. GPACLEAR Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-117. GPACLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
30	GPIO30	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
29	GPIO29	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
28	GPIO28	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
27	GPIO27	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
26	GPIO26	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
25	GPIO25	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
24	GPIO24	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
23	GPIO23	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
22	GPIO22	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
21	GPIO21	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
20	GPIO20	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
19	GPIO19	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn

Table 15-117. GPACLEAR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO18	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
17	GPIO17	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
16	GPIO16	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
15	GPIO15	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
14	GPIO14	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
13	GPIO13	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
12	GPIO12	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
11	GPIO11	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
10	GPIO10	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
9	GPIO9	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
8	GPIO8	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
7	GPIO7	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
6	GPIO6	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
5	GPIO5	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
4	GPIO4	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
3	GPIO3	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
2	GPIO2	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
1	GPIO1	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
0	GPIO0	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn

15.10.4.4 GPATOGGLE Register (Offset = 6h) [Reset = 0h]

GPATOGGLE is shown in Figure 15-105 and described in Table 15-118.

Return to the Summary Table.

GPIO A Data Toggle Register (GPIO0 to 31)
Writing a 1 will toggle GPIO0 output data latch 1 to 0 or 0 to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-105. GPATOGGLE Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-118. GPATOGGLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
30	GPIO30	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
29	GPIO29	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
28	GPIO28	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
27	GPIO27	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
26	GPIO26	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
25	GPIO25	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
24	GPIO24	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
23	GPIO23	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
22	GPIO22	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
21	GPIO21	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
20	GPIO20	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
19	GPIO19	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn

Table 15-118. GPATOGGLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO18	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
17	GPIO17	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
16	GPIO16	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
15	GPIO15	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
14	GPIO14	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
13	GPIO13	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
12	GPIO12	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
11	GPIO11	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
10	GPIO10	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
9	GPIO9	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
8	GPIO8	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
7	GPIO7	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
6	GPIO6	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
5	GPIO5	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
4	GPIO4	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
3	GPIO3	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
2	GPIO2	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
1	GPIO1	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
0	GPIO0	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn

15.10.4.5 GPBDAT Register (Offset = 8h) [Reset = 0h]

GPBDAT is shown in Figure 15-106 and described in Table 15-119.

Return to the Summary Table.

GPIO B Data Register (GPIO32 to 63)
Reading this register reflects the current state of the GPIO pin regardless of which mode the GPIO is in.
Writing to this register will set the GPIO pin high or low if the pin is enabled for GPIO output mode. If the GPIO is not in output mode the value written is latched but will not be reflected on the GPIO pin or reads of the GPxDAT register. The written value latched will become active when the GPIO is put into GPIO Output mode
A system reset will clear all bits and latched values to zero.
NOTE: Bit-wise read-modify-write operations should not be performed on this register. For bit-wise operations the GPxSET, GPxCLEAR, or GPxTOGGLE registers should be used instead. If direct writes to GPxDAT are necessary, the entire register should be written at one time.

Figure 15-106. GPBDAT Register

31	30	29	28	27	26	25	24
GPIO63	GPIO62	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	GPIO38	GPIO37	GPIO36	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-119. GPBDAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO63	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
30	GPIO62	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
29	GPIO61	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
28	GPIO60	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
27	GPIO59	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
26	GPIO58	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
25	GPIO57	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
24	GPIO56	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
23	GPIO55	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
22	GPIO54	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
21	GPIO53	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn

Table 15-119. GPBDAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	GPIO52	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
19	GPIO51	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
18	GPIO50	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
17	GPIO49	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
16	GPIO48	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
15	GPIO47	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
14	GPIO46	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
13	GPIO45	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
12	GPIO44	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
11	GPIO43	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
10	GPIO42	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
9	GPIO41	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
8	GPIO40	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
7	GPIO39	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
6	GPIO38	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
5	GPIO37	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
4	GPIO36	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
3	GPIO35	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
2	GPIO34	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
1	GPIO33	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
0	GPIO32	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn

15.10.4.6 GPBSET Register (Offset = Ah) [Reset = 0h]

GPBSET is shown in Figure 15-107 and described in Table 15-120.

Return to the Summary Table.

GPIO B Data Set Register (GPIO32 to 63)
Writing a 1 will force GPIO output data latch to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-107. GPBSET Register

31	30	29	28	27	26	25	24
GPIO63	GPIO62	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	GPIO38	GPIO37	GPIO36	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-120. GPBSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO63	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
30	GPIO62	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
29	GPIO61	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
28	GPIO60	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
27	GPIO59	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
26	GPIO58	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
25	GPIO57	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
24	GPIO56	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
23	GPIO55	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
22	GPIO54	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
21	GPIO53	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
20	GPIO52	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
19	GPIO51	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn

Table 15-120. GPBSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO50	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
17	GPIO49	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
16	GPIO48	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
15	GPIO47	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
14	GPIO46	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
13	GPIO45	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
12	GPIO44	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
11	GPIO43	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
10	GPIO42	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
9	GPIO41	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
8	GPIO40	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
7	GPIO39	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
6	GPIO38	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
5	GPIO37	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
4	GPIO36	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
3	GPIO35	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
2	GPIO34	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
1	GPIO33	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
0	GPIO32	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn

15.10.4.7 GPBCLEAR Register (Offset = Ch) [Reset = 0h]

GPBCLEAR is shown in Figure 15-108 and described in Table 15-121.

Return to the Summary Table.

GPIO B Data Clear Register (GPIO32 to 63)
Writing a 1 will force GPIO0 output data latch to 0.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-108. GPBCLEAR Register

31	30	29	28	27	26	25	24
GPIO63	GPIO62	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	GPIO38	GPIO37	GPIO36	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-121. GPBCLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO63	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
30	GPIO62	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
29	GPIO61	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
28	GPIO60	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
27	GPIO59	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
26	GPIO58	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
25	GPIO57	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
24	GPIO56	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
23	GPIO55	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
22	GPIO54	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
21	GPIO53	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
20	GPIO52	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
19	GPIO51	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn

Table 15-121. GPBCLEAR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO50	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
17	GPIO49	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
16	GPIO48	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
15	GPIO47	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
14	GPIO46	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
13	GPIO45	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
12	GPIO44	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
11	GPIO43	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
10	GPIO42	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
9	GPIO41	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
8	GPIO40	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
7	GPIO39	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
6	GPIO38	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
5	GPIO37	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
4	GPIO36	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
3	GPIO35	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
2	GPIO34	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
1	GPIO33	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
0	GPIO32	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn

15.10.4.8 GPBTOGGLE Register (Offset = Eh) [Reset = 0h]

GPBTOGGLE is shown in Figure 15-109 and described in Table 15-122.

Return to the Summary Table.

GPIO B Data Toggle Register (GPIO32 to 63)
Writing a 1 will toggle GPIO0 output data latch 1 to 0 or 0 to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-109. GPBTOGGLE Register

31	30	29	28	27	26	25	24
GPIO63	GPIO62	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	GPIO38	GPIO37	GPIO36	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-122. GPBTOGGLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO63	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
30	GPIO62	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
29	GPIO61	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
28	GPIO60	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
27	GPIO59	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
26	GPIO58	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
25	GPIO57	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
24	GPIO56	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
23	GPIO55	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
22	GPIO54	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
21	GPIO53	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
20	GPIO52	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
19	GPIO51	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn

Table 15-122. GPBTOGGLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO50	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
17	GPIO49	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
16	GPIO48	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
15	GPIO47	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
14	GPIO46	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
13	GPIO45	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
12	GPIO44	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
11	GPIO43	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
10	GPIO42	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
9	GPIO41	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
8	GPIO40	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
7	GPIO39	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
6	GPIO38	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
5	GPIO37	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
4	GPIO36	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
3	GPIO35	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
2	GPIO34	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
1	GPIO33	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
0	GPIO32	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn

15.10.4.9 GPCDAT Register (Offset = 10h) [Reset = 0h]

GPCDAT is shown in Figure 15-110 and described in Table 15-123.

Return to the Summary Table.

GPIO C Data Register (GPIO64 to 95)
Reading this register reflects the current state of the GPIO pin regardless of which mode the GPIO is in.
Writing to this register will set the GPIO pin high or low if the pin is enabled for GPIO output mode. If the GPIO is not in output mode the value written is latched but will not be reflected on the GPIO pin or reads of the GPxDAT register. The written value latched will become active when the GPIO is put into GPIO Output mode
A system reset will clear all bits and latched values to zero.
NOTE: Bit-wise read-modify-write operations should not be performed on this register. For bit-wise operations the GPxSET, GPxCLEAR, or GPxTOGGLE registers should be used instead. If direct writes to GPxDAT are necessary, the entire register should be written at one time.

Figure 15-110. GPCDAT Register

31	30	29	28	27	26	25	24
GPIO95	GPIO94	GPIO93	GPIO92	GPIO91	GPIO90	GPIO89	GPIO88
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO87	GPIO86	GPIO85	GPIO84	GPIO83	GPIO82	GPIO81	GPIO80
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO79	GPIO78	GPIO77	GPIO76	GPIO75	GPIO74	GPIO73	GPIO72
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO71	GPIO70	GPIO69	GPIO68	GPIO67	GPIO66	GPIO65	GPIO64
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-123. GPCDAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO95	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
30	GPIO94	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
29	GPIO93	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
28	GPIO92	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
27	GPIO91	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
26	GPIO90	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
25	GPIO89	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
24	GPIO88	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
23	GPIO87	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
22	GPIO86	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
21	GPIO85	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn

Table 15-123. GPCDAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	GPIO84	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
19	GPIO83	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
18	GPIO82	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
17	GPIO81	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
16	GPIO80	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
15	GPIO79	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
14	GPIO78	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
13	GPIO77	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
12	GPIO76	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
11	GPIO75	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
10	GPIO74	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
9	GPIO73	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
8	GPIO72	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
7	GPIO71	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
6	GPIO70	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
5	GPIO69	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
4	GPIO68	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
3	GPIO67	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
2	GPIO66	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
1	GPIO65	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
0	GPIO64	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn

15.10.4.10 GPCSET Register (Offset = 12h) [Reset = 0h]

GPCSET is shown in Figure 15-111 and described in Table 15-124.

Return to the Summary Table.

GPIO C Data Set Register (GPIO64 to 95)
Writing a 1 will force GPIO output data latch to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-111. GPCSET Register

31	30	29	28	27	26	25	24
GPIO95	GPIO94	GPIO93	GPIO92	GPIO91	GPIO90	GPIO89	GPIO88
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO87	GPIO86	GPIO85	GPIO84	GPIO83	GPIO82	GPIO81	GPIO80
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO79	GPIO78	GPIO77	GPIO76	GPIO75	GPIO74	GPIO73	GPIO72
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO71	GPIO70	GPIO69	GPIO68	GPIO67	GPIO66	GPIO65	GPIO64
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-124. GPCSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO95	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
30	GPIO94	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
29	GPIO93	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
28	GPIO92	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
27	GPIO91	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
26	GPIO90	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
25	GPIO89	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
24	GPIO88	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
23	GPIO87	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
22	GPIO86	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
21	GPIO85	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
20	GPIO84	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
19	GPIO83	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn

Table 15-124. GPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO82	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
17	GPIO81	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
16	GPIO80	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
15	GPIO79	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
14	GPIO78	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
13	GPIO77	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
12	GPIO76	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
11	GPIO75	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
10	GPIO74	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
9	GPIO73	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
8	GPIO72	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
7	GPIO71	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
6	GPIO70	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
5	GPIO69	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
4	GPIO68	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
3	GPIO67	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
2	GPIO66	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
1	GPIO65	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
0	GPIO64	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn

15.10.4.11 GPCCLEAR Register (Offset = 14h) [Reset = 0h]

GPCCLEAR is shown in Figure 15-112 and described in Table 15-125.

Return to the Summary Table.

GPIO C Data Clear Register (GPIO64 to 95)
Writing a 1 will force GPIO0 output data latch to 0.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-112. GPCCLEAR Register

31	30	29	28	27	26	25	24
GPIO95	GPIO94	GPIO93	GPIO92	GPIO91	GPIO90	GPIO89	GPIO88
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO87	GPIO86	GPIO85	GPIO84	GPIO83	GPIO82	GPIO81	GPIO80
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO79	GPIO78	GPIO77	GPIO76	GPIO75	GPIO74	GPIO73	GPIO72
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO71	GPIO70	GPIO69	GPIO68	GPIO67	GPIO66	GPIO65	GPIO64
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-125. GPCCLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO95	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
30	GPIO94	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
29	GPIO93	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
28	GPIO92	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
27	GPIO91	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
26	GPIO90	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
25	GPIO89	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
24	GPIO88	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
23	GPIO87	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
22	GPIO86	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
21	GPIO85	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
20	GPIO84	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
19	GPIO83	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn

Table 15-125. GPCCLEAR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO82	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
17	GPIO81	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
16	GPIO80	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
15	GPIO79	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
14	GPIO78	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
13	GPIO77	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
12	GPIO76	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
11	GPIO75	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
10	GPIO74	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
9	GPIO73	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
8	GPIO72	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
7	GPIO71	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
6	GPIO70	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
5	GPIO69	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
4	GPIO68	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
3	GPIO67	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
2	GPIO66	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
1	GPIO65	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
0	GPIO64	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn

15.10.4.12 GPCTOGGLE Register (Offset = 16h) [Reset = 0h]

GPCTOGGLE is shown in Figure 15-113 and described in Table 15-126.

Return to the Summary Table.

GPIO C Data Toggle Register (GPIO64 to 95)
Writing a 1 will toggle GPIO0 output data latch 1 to 0 or 0 to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-113. GPCTOGGLE Register

31	30	29	28	27	26	25	24
GPIO95	GPIO94	GPIO93	GPIO92	GPIO91	GPIO90	GPIO89	GPIO88
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO87	GPIO86	GPIO85	GPIO84	GPIO83	GPIO82	GPIO81	GPIO80
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO79	GPIO78	GPIO77	GPIO76	GPIO75	GPIO74	GPIO73	GPIO72
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO71	GPIO70	GPIO69	GPIO68	GPIO67	GPIO66	GPIO65	GPIO64
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-126. GPCTOGGLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO95	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
30	GPIO94	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
29	GPIO93	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
28	GPIO92	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
27	GPIO91	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
26	GPIO90	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
25	GPIO89	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
24	GPIO88	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
23	GPIO87	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
22	GPIO86	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
21	GPIO85	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
20	GPIO84	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
19	GPIO83	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn

Table 15-126. GPCTOGGLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO82	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
17	GPIO81	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
16	GPIO80	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
15	GPIO79	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
14	GPIO78	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
13	GPIO77	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
12	GPIO76	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
11	GPIO75	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
10	GPIO74	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
9	GPIO73	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
8	GPIO72	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
7	GPIO71	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
6	GPIO70	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
5	GPIO69	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
4	GPIO68	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
3	GPIO67	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
2	GPIO66	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
1	GPIO65	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
0	GPIO64	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn

15.10.4.13 GPDDAT Register (Offset = 18h) [Reset = 0h]

GPDDAT is shown in Figure 15-114 and described in Table 15-127.

Return to the Summary Table.

GPIO D Data Register (GPIO96 to 127)
Reading this register reflects the current state of the GPIO pin regardless of which mode the GPIO is in.
Writing to this register will set the GPIO pin high or low if the pin is enabled for GPIO output mode. If the GPIO is not in output mode the value written is latched but will not be reflected on the GPIO pin or reads of the GPxDAT register. The written value latched will become active when the GPIO is put into GPIO Output mode
A system reset will clear all bits and latched values to zero.
NOTE: Bit-wise read-modify-write operations should not be performed on this register. For bit-wise operations the GPxSET, GPxCLEAR, or GPxTOGGLE registers should be used instead. If direct writes to GPxDAT are necessary, the entire register should be written at one time.

Figure 15-114. GPDDAT Register

31	30	29	28	27	26	25	24
GPIO127	GPIO126	GPIO125	GPIO124	GPIO123	GPIO122	GPIO121	GPIO120
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO119	GPIO118	GPIO117	GPIO116	GPIO115	GPIO114	GPIO113	GPIO112
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO111	GPIO110	GPIO109	GPIO108	GPIO107	GPIO106	GPIO105	GPIO104
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO103	GPIO102	GPIO101	GPIO100	GPIO99	GPIO98	GPIO97	GPIO96
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-127. GPDDAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO127	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
30	GPIO126	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
29	GPIO125	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
28	GPIO124	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
27	GPIO123	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
26	GPIO122	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
25	GPIO121	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
24	GPIO120	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
23	GPIO119	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
22	GPIO118	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
21	GPIO117	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn

Table 15-127. GPDDAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	GPIO116	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
19	GPIO115	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
18	GPIO114	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
17	GPIO113	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
16	GPIO112	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
15	GPIO111	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
14	GPIO110	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
13	GPIO109	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
12	GPIO108	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
11	GPIO107	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
10	GPIO106	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
9	GPIO105	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
8	GPIO104	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
7	GPIO103	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
6	GPIO102	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
5	GPIO101	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
4	GPIO100	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
3	GPIO99	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
2	GPIO98	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
1	GPIO97	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
0	GPIO96	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn

15.10.4.14 GPDSET Register (Offset = 1Ah) [Reset = 0h]

GPDSET is shown in Figure 15-115 and described in Table 15-128.

Return to the Summary Table.

GPIO D Data Set Register (GPIO96 to 127)
Writing a 1 will force GPIO output data latch to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-115. GPDSET Register

31	30	29	28	27	26	25	24
GPIO127	GPIO126	GPIO125	GPIO124	GPIO123	GPIO122	GPIO121	GPIO120
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO119	GPIO118	GPIO117	GPIO116	GPIO115	GPIO114	GPIO113	GPIO112
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO111	GPIO110	GPIO109	GPIO108	GPIO107	GPIO106	GPIO105	GPIO104
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO103	GPIO102	GPIO101	GPIO100	GPIO99	GPIO98	GPIO97	GPIO96
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-128. GPDSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO127	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
30	GPIO126	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
29	GPIO125	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
28	GPIO124	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
27	GPIO123	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
26	GPIO122	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
25	GPIO121	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
24	GPIO120	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
23	GPIO119	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
22	GPIO118	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
21	GPIO117	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
20	GPIO116	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
19	GPIO115	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn

Table 15-128. GPDSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO114	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
17	GPIO113	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
16	GPIO112	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
15	GPIO111	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
14	GPIO110	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
13	GPIO109	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
12	GPIO108	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
11	GPIO107	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
10	GPIO106	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
9	GPIO105	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
8	GPIO104	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
7	GPIO103	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
6	GPIO102	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
5	GPIO101	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
4	GPIO100	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
3	GPIO99	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
2	GPIO98	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
1	GPIO97	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
0	GPIO96	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn

15.10.4.15 GPDCLEAR Register (Offset = 1Ch) [Reset = 0h]

GPDCLEAR is shown in Figure 15-116 and described in Table 15-129.

Return to the Summary Table.

GPIO D Data Clear Register (GPIO96 to 127)
Writing a 1 will force GPIO0 output data latch to 0.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-116. GPDCLEAR Register

31	30	29	28	27	26	25	24
GPIO127	GPIO126	GPIO125	GPIO124	GPIO123	GPIO122	GPIO121	GPIO120
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO119	GPIO118	GPIO117	GPIO116	GPIO115	GPIO114	GPIO113	GPIO112
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO111	GPIO110	GPIO109	GPIO108	GPIO107	GPIO106	GPIO105	GPIO104
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO103	GPIO102	GPIO101	GPIO100	GPIO99	GPIO98	GPIO97	GPIO96
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-129. GPDCLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO127	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
30	GPIO126	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
29	GPIO125	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
28	GPIO124	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
27	GPIO123	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
26	GPIO122	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
25	GPIO121	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
24	GPIO120	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
23	GPIO119	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
22	GPIO118	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
21	GPIO117	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
20	GPIO116	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
19	GPIO115	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn

Table 15-129. GPDCLEAR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO114	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
17	GPIO113	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
16	GPIO112	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
15	GPIO111	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
14	GPIO110	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
13	GPIO109	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
12	GPIO108	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
11	GPIO107	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
10	GPIO106	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
9	GPIO105	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
8	GPIO104	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
7	GPIO103	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
6	GPIO102	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
5	GPIO101	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
4	GPIO100	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
3	GPIO99	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
2	GPIO98	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
1	GPIO97	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
0	GPIO96	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn

15.10.4.16 GPDTOGGLE Register (Offset = 1Eh) [Reset = 0h]

GPDTOGGLE is shown in Figure 15-117 and described in Table 15-130.

Return to the Summary Table.

GPIO D Data Toggle Register (GPIO96 to 127)
Writing a 1 will toggle GPIO0 output data latch 1 to 0 or 0 to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-117. GPDTOGGLE Register

31	30	29	28	27	26	25	24
GPIO127	GPIO126	GPIO125	GPIO124	GPIO123	GPIO122	GPIO121	GPIO120
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO119	GPIO118	GPIO117	GPIO116	GPIO115	GPIO114	GPIO113	GPIO112
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO111	GPIO110	GPIO109	GPIO108	GPIO107	GPIO106	GPIO105	GPIO104
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO103	GPIO102	GPIO101	GPIO100	GPIO99	GPIO98	GPIO97	GPIO96
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-130. GPDTOGGLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO127	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
30	GPIO126	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
29	GPIO125	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
28	GPIO124	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
27	GPIO123	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
26	GPIO122	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
25	GPIO121	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
24	GPIO120	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
23	GPIO119	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
22	GPIO118	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
21	GPIO117	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
20	GPIO116	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
19	GPIO115	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn

Table 15-130. GPDTOGGLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO114	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
17	GPIO113	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
16	GPIO112	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
15	GPIO111	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
14	GPIO110	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
13	GPIO109	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
12	GPIO108	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
11	GPIO107	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
10	GPIO106	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
9	GPIO105	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
8	GPIO104	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
7	GPIO103	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
6	GPIO102	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
5	GPIO101	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
4	GPIO100	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
3	GPIO99	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
2	GPIO98	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
1	GPIO97	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
0	GPIO96	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn

15.10.4.17 GPEDAT Register (Offset = 20h) [Reset = 0h]

GPEDAT is shown in Figure 15-118 and described in Table 15-131.

Return to the Summary Table.

GPIO E Data Register (GPIO128 to 159)
Reading this register reflects the current state of the GPIO pin regardless of which mode the GPIO is in.
Writing to this register will set the GPIO pin high or low if the pin is enabled for GPIO output mode. If the GPIO is not in output mode the value written is latched but will not be reflected on the GPIO pin or reads of the GPxDAT register. The written value latched will become active when the GPIO is put into GPIO Output mode
A system reset will clear all bits and latched values to zero.
NOTE: Bit-wise read-modify-write operations should not be performed on this register. For bit-wise operations the GPxSET, GPxCLEAR, or GPxTOGGLE registers should be used instead. If direct writes to GPxDAT are necessary, the entire register should be written at one time.

Figure 15-118. GPEDAT Register

31	30	29	28	27	26	25	24
GPIO159	GPIO158	GPIO157	GPIO156	GPIO155	GPIO154	GPIO153	GPIO152
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO151	GPIO150	GPIO149	GPIO148	GPIO147	GPIO146	GPIO145	GPIO144
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO143	GPIO142	GPIO141	GPIO140	GPIO139	GPIO138	GPIO137	GPIO136
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO135	GPIO134	GPIO133	GPIO132	GPIO131	GPIO130	GPIO129	GPIO128
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-131. GPEDAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO159	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
30	GPIO158	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
29	GPIO157	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
28	GPIO156	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
27	GPIO155	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
26	GPIO154	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
25	GPIO153	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
24	GPIO152	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
23	GPIO151	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
22	GPIO150	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
21	GPIO149	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn

Table 15-131. GPEDAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	GPIO148	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
19	GPIO147	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
18	GPIO146	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
17	GPIO145	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
16	GPIO144	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
15	GPIO143	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
14	GPIO142	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
13	GPIO141	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
12	GPIO140	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
11	GPIO139	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
10	GPIO138	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
9	GPIO137	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
8	GPIO136	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
7	GPIO135	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
6	GPIO134	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
5	GPIO133	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
4	GPIO132	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
3	GPIO131	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
2	GPIO130	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
1	GPIO129	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
0	GPIO128	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn

15.10.4.18 GPESET Register (Offset = 22h) [Reset = 0h]

GPESET is shown in Figure 15-119 and described in Table 15-132.

Return to the Summary Table.

GPIO E Data Set Register (GPIO128 to 159)
Writing a 1 will force GPIO output data latch to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-119. GPESET Register

31	30	29	28	27	26	25	24
GPIO159	GPIO158	GPIO157	GPIO156	GPIO155	GPIO154	GPIO153	GPIO152
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO151	GPIO150	GPIO149	GPIO148	GPIO147	GPIO146	GPIO145	GPIO144
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO143	GPIO142	GPIO141	GPIO140	GPIO139	GPIO138	GPIO137	GPIO136
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO135	GPIO134	GPIO133	GPIO132	GPIO131	GPIO130	GPIO129	GPIO128
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-132. GPESET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO159	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
30	GPIO158	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
29	GPIO157	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
28	GPIO156	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
27	GPIO155	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
26	GPIO154	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
25	GPIO153	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
24	GPIO152	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
23	GPIO151	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
22	GPIO150	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
21	GPIO149	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
20	GPIO148	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
19	GPIO147	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn

Table 15-132. GPESET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO146	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
17	GPIO145	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
16	GPIO144	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
15	GPIO143	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
14	GPIO142	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
13	GPIO141	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
12	GPIO140	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
11	GPIO139	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
10	GPIO138	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
9	GPIO137	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
8	GPIO136	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
7	GPIO135	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
6	GPIO134	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
5	GPIO133	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
4	GPIO132	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
3	GPIO131	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
2	GPIO130	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
1	GPIO129	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
0	GPIO128	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn

15.10.4.19 GPECLEAR Register (Offset = 24h) [Reset = 0h]

GPECLEAR is shown in Figure 15-120 and described in Table 15-133.

Return to the Summary Table.

GPIO E Data Clear Register (GPIO128 to 159)
Writing a 1 will force GPIO0 output data latch to 0.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-120. GPECLEAR Register

31	30	29	28	27	26	25	24
GPIO159	GPIO158	GPIO157	GPIO156	GPIO155	GPIO154	GPIO153	GPIO152
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO151	GPIO150	GPIO149	GPIO148	GPIO147	GPIO146	GPIO145	GPIO144
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO143	GPIO142	GPIO141	GPIO140	GPIO139	GPIO138	GPIO137	GPIO136
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO135	GPIO134	GPIO133	GPIO132	GPIO131	GPIO130	GPIO129	GPIO128
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-133. GPECLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO159	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
30	GPIO158	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
29	GPIO157	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
28	GPIO156	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
27	GPIO155	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
26	GPIO154	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
25	GPIO153	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
24	GPIO152	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
23	GPIO151	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
22	GPIO150	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
21	GPIO149	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
20	GPIO148	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
19	GPIO147	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn

Table 15-133. GPECLEAR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO146	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
17	GPIO145	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
16	GPIO144	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
15	GPIO143	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
14	GPIO142	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
13	GPIO141	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
12	GPIO140	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
11	GPIO139	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
10	GPIO138	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
9	GPIO137	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
8	GPIO136	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
7	GPIO135	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
6	GPIO134	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
5	GPIO133	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
4	GPIO132	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
3	GPIO131	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
2	GPIO130	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
1	GPIO129	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
0	GPIO128	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn

15.10.4.20 GPETOGGLE Register (Offset = 26h) [Reset = 0h]

GPETOGGLE is shown in Figure 15-121 and described in Table 15-134.

Return to the Summary Table.

GPIO E Data Toggle Register (GPIO128 to 159)
Writing a 1 will toggle GPIO0 output data latch 1 to 0 or 0 to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-121. GPETOGGLE Register

31	30	29	28	27	26	25	24
GPIO159	GPIO158	GPIO157	GPIO156	GPIO155	GPIO154	GPIO153	GPIO152
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO151	GPIO150	GPIO149	GPIO148	GPIO147	GPIO146	GPIO145	GPIO144
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO143	GPIO142	GPIO141	GPIO140	GPIO139	GPIO138	GPIO137	GPIO136
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO135	GPIO134	GPIO133	GPIO132	GPIO131	GPIO130	GPIO129	GPIO128
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-134. GPETOGGLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO159	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
30	GPIO158	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
29	GPIO157	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
28	GPIO156	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
27	GPIO155	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
26	GPIO154	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
25	GPIO153	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
24	GPIO152	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
23	GPIO151	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
22	GPIO150	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
21	GPIO149	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
20	GPIO148	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
19	GPIO147	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn

Table 15-134. GPETOGGLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO146	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
17	GPIO145	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
16	GPIO144	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
15	GPIO143	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
14	GPIO142	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
13	GPIO141	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
12	GPIO140	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
11	GPIO139	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
10	GPIO138	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
9	GPIO137	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
8	GPIO136	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
7	GPIO135	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
6	GPIO134	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
5	GPIO133	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
4	GPIO132	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
3	GPIO131	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
2	GPIO130	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
1	GPIO129	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
0	GPIO128	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn

15.10.4.21 GPFDAT Register (Offset = 28h) [Reset = 0h]

GPFDAT is shown in Figure 15-122 and described in Table 15-135.

Return to the Summary Table.

GPIO F Data Register (GPIO160 to 168)
Reading this register reflects the current state of the GPIO pin regardless of which mode the GPIO is in.
Writing to this register will set the GPIO pin high or low if the pin is enabled for GPIO output mode. If the GPIO is not in output mode the value written is latched but will not be reflected on the GPIO pin or reads of the GPxDAT register. The written value latched will become active when the GPIO is put into GPIO Output mode
A system reset will clear all bits and latched values to zero.
NOTE: Bit-wise read-modify-write operations should not be performed on this register. For bit-wise operations the GPxSET, GPxCLEAR, or GPxTOGGLE registers should be used instead. If direct writes to GPxDAT are necessary, the entire register should be written at one time.

Figure 15-122. GPFDAT Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	GPIO168
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO167	GPIO166	GPIO165	GPIO164	GPIO163	GPIO162	GPIO161	GPIO160
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-135. GPFDAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	RESERVED	R/W	0h	Reserved
28	RESERVED	R/W	0h	Reserved
27	RESERVED	R/W	0h	Reserved
26	RESERVED	R/W	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23	RESERVED	R/W	0h	Reserved
22	RESERVED	R/W	0h	Reserved
21	RESERVED	R/W	0h	Reserved
20	RESERVED	R/W	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved

Table 15-135. GPFDAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved
8	GPIO168	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
7	GPIO167	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
6	GPIO166	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
5	GPIO165	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
4	GPIO164	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
3	GPIO163	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
2	GPIO162	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
1	GPIO161	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn
0	GPIO160	R/W	0h	Data Register for this pin Reset type: CPU1.SYSRSn

15.10.4.22 GPFSET Register (Offset = 2Ah) [Reset = 0h]

GPFSET is shown in Figure 15-123 and described in Table 15-136.

Return to the Summary Table.

GPIO F Data Set Register (GPIO160 to 168)
Writing a 1 will force GPIO output data latch to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-123. GPFSET Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	GPIO168
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO167	GPIO166	GPIO165	GPIO164	GPIO163	GPIO162	GPIO161	GPIO160
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-136. GPFSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	RESERVED	R/W	0h	Reserved
28	RESERVED	R/W	0h	Reserved
27	RESERVED	R/W	0h	Reserved
26	RESERVED	R/W	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23	RESERVED	R/W	0h	Reserved
22	RESERVED	R/W	0h	Reserved
21	RESERVED	R/W	0h	Reserved
20	RESERVED	R/W	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved

Table 15-136. GPFSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	GPIO168	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
7	GPIO167	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
6	GPIO166	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
5	GPIO165	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
4	GPIO164	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
3	GPIO163	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
2	GPIO162	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
1	GPIO161	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn
0	GPIO160	R/W	0h	Output Set bit for this pin Reset type: CPU1.SYSRSn

15.10.4.23 GPFCLEAR Register (Offset = 2Ch) [Reset = 0h]

GPFCLEAR is shown in Figure 15-124 and described in Table 15-137.

Return to the Summary Table.

GPIO F Data Clear Register (GPIO160 to 168)
Writing a 1 will force GPIO0 output data latch to 0.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-124. GPFCLEAR Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	GPIO168
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO167	GPIO166	GPIO165	GPIO164	GPIO163	GPIO162	GPIO161	GPIO160
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-137. GPFCLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	RESERVED	R/W	0h	Reserved
28	RESERVED	R/W	0h	Reserved
27	RESERVED	R/W	0h	Reserved
26	RESERVED	R/W	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23	RESERVED	R/W	0h	Reserved
22	RESERVED	R/W	0h	Reserved
21	RESERVED	R/W	0h	Reserved
20	RESERVED	R/W	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved

Table 15-137. GPFCLEAR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	GPIO168	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
7	GPIO167	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
6	GPIO166	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
5	GPIO165	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
4	GPIO164	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
3	GPIO163	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
2	GPIO162	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
1	GPIO161	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn
0	GPIO160	R/W	0h	Output Clear bit for this pin Reset type: CPU1.SYSRSn

15.10.4.24 GPFTOGGLE Register (Offset = 2Eh) [Reset = 0h]

GPFTOGGLE is shown in Figure 15-125 and described in Table 15-138.

Return to the Summary Table.

GPIO F Data Toggle Register (GPIO160 to 168)
Writing a 1 will toggle GPIO0 output data latch 1 to 0 or 0 to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-125. GPFTOGGLE Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	GPIO168
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO167	GPIO166	GPIO165	GPIO164	GPIO163	GPIO162	GPIO161	GPIO160
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-138. GPFTOGGLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	RESERVED	R/W	0h	Reserved
28	RESERVED	R/W	0h	Reserved
27	RESERVED	R/W	0h	Reserved
26	RESERVED	R/W	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23	RESERVED	R/W	0h	Reserved
22	RESERVED	R/W	0h	Reserved
21	RESERVED	R/W	0h	Reserved
20	RESERVED	R/W	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved

Table 15-138. GPFTOGGLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	GPIO168	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
7	GPIO167	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
6	GPIO166	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
5	GPIO165	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
4	GPIO164	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
3	GPIO163	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
2	GPIO162	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
1	GPIO161	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn
0	GPIO160	R/W	0h	Output Toggle Register GPIO pin Reset type: CPU1.SYSRSn

15.10.5 GPIO_DATA_READ_REGS Registers

Table 15-139 lists the memory-mapped registers for the GPIO_DATA_READ_REGS registers. All register offset addresses not listed in Table 15-139 should be considered as reserved locations and the register contents should not be modified.

Table 15-139. GPIO_DATA_READ_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	GPADAT_R	GPIO A Data Read Register		Go
2h	GPBDAT_R	GPIO B Data Read Register		Go
4h	GPCDAT_R	GPIO C Data Read Register		Go
6h	GPDDAT_R	GPIO D Data Read Register		Go
8h	GPEDAT_R	GPIO E Data Read Register		Go
Ah	GPFDAT_R	GPIO F Data Read Register		Go

Complex bit access types are encoded to fit into small table cells. Table 15-140 shows the codes that are used for access types in this section.

Table 15-140. GPIO_DATA_READ_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

15.10.5.1 GPADAT_R Register (Offset = 0h) [Reset = 0h]

GPADAT_R is shown in Figure 15-126 and described in Table 15-141.

Return to the Summary Table.

GPIO A Data Read Register.
Returns the contents of GPADAT register on a read, write to this register has no effect

Figure 15-126. GPADAT_R Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 15-141. GPADAT_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	A read from this register returns the contents of GPADAT register, writes have no impact Reset type: CPU1.SYSRSn

15.10.5.2 GPBDAT_R Register (Offset = 2h) [Reset = 0h]

GPBDAT_R is shown in Figure 15-127 and described in Table 15-142.

Return to the Summary Table.

GPIO B Data Read Register.
Returns the contents of GPBDAT register on a read, write to this register has no effect

Figure 15-127. GPBDAT_R Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 15-142. GPBDAT_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	A read from this register returns the contents of GPBDAT register, writes have no impact Reset type: CPU1.SYSRSn

15.10.5.3 GPCDAT_R Register (Offset = 4h) [Reset = 0h]

GPCDAT_R is shown in Figure 15-128 and described in Table 15-143.

Return to the Summary Table.

GPIO C Data Read Register.
Returns the contents of GPCDAT register on a read, write to this register has no effect

Figure 15-128. GPCDAT_R Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 15-143. GPCDAT_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	A read from this register returns the contents of GPCDAT register, writes have no impact Reset type: CPU1.SYSRSn

15.10.5.4 GPDDAT_R Register (Offset = 6h) [Reset = 0h]

GPDDAT_R is shown in Figure 15-129 and described in Table 15-144.

Return to the Summary Table.

GPIO D Data Read Register.
Returns the contents of GPDDAT register on a read, write to this register has no effect

Figure 15-129. GPDDAT_R Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 15-144. GPDDAT_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	A read from this register returns the contents of GPDDAT register, writes have no impact Reset type: CPU1.SYSRSn

15.10.5.5 GPEDAT_R Register (Offset = 8h) [Reset = 0h]

GPEDAT_R is shown in Figure 15-130 and described in Table 15-145.

Return to the Summary Table.

GPIO E Data Read Register.
Returns the contents of GPEDAT register on a read, write to this register has no effect

Figure 15-130. GPEDAT_R Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 15-145. GPEDAT_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	A read from this register returns the contents of GPEDAT register, writes have no impact Reset type: CPU1.SYSRSn

15.10.5.6 GPFDAT_R Register (Offset = Ah) [Reset = 0h]

GPFDAT_R is shown in Figure 15-131 and described in Table 15-146.

Return to the Summary Table.

GPIO F Data Read Register.
Returns the contents of GPFDAT register on a read, write to this register has no effect

Figure 15-131. GPFDAT_R Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 15-146. GPFDAT_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	A read from this register returns the contents of GPFDAT register, writes have no impact Reset type: CPU1.SYSRSn

15.10.6 CM_GPIO_DATA_REGS Registers

Table 15-147 lists the memory-mapped registers for the CM_GPIO_DATA_REGS registers. All register offset addresses not listed in Table 15-147 should be considered as reserved locations and the register contents should not be modified.

Table 15-147. CM_GPIO_DATA_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	GPADAT	GPIO A Data Register (GPIO0 to 31)		Go
4h	GPASET	GPIO A Data Set Register (GPIO0 to 31)		Go
8h	GPACLEAR	GPIO A Data Clear Register (GPIO0 to 31)		Go
Ch	GPATOGGLE	GPIO A Data Toggle Register (GPIO0 to 31)		Go
10h	GPBDAT	GPIO B Data Register (GPIO32 to 63)		Go
14h	GPBSET	GPIO B Data Set Register (GPIO32 to 63)		Go
18h	GPBCLEAR	GPIO B Data Clear Register (GPIO32 to 63)		Go
1Ch	GPBTOGGLE	GPIO B Data Toggle Register (GPIO32 to 63)		Go
20h	GPCDAT	GPIO C Data Register (GPIO64 to 95)		Go
24h	GPCSET	GPIO C Data Set Register (GPIO64 to 95)		Go
28h	GPCCLEAR	GPIO C Data Clear Register (GPIO64 to 95)		Go
2Ch	GPCTOGGLE	GPIO C Data Toggle Register (GPIO64 to 95)		Go
30h	GPDDAT	GPIO D Data Register (GPIO96 to 127)		Go
34h	GPDSET	GPIO D Data Set Register (GPIO96 to 127)		Go
38h	GPDCLEAR	GPIO D Data Clear Register (GPIO96 to 127)		Go
3Ch	GPDTOGGLE	GPIO D Data Toggle Register (GPIO96 to 127)		Go
40h	GPEDAT	GPIO E Data Register (GPIO128 to 159)		Go
44h	GPESET	GPIO E Data Set Register (GPIO128 to 159)		Go
48h	GPECLEAR	GPIO E Data Clear Register (GPIO128 to 159)		Go
4Ch	GPETOGGLE	GPIO E Data Toggle Register (GPIO128 to 159)		Go
50h	GPFDAT	GPIO F Data Register (GPIO160 to 191)		Go
54h	GPFSET	GPIO F Data Set Register (GPIO160 to 191)		Go
58h	GPFCLEAR	GPIO F Data Clear Register (GPIO160 to 191)		Go
5Ch	GPFTOGGLE	GPIO F Data Toggle Register (GPIO160 to 191)		Go

Complex bit access types are encoded to fit into small table cells. Table 15-148 shows the codes that are used for access types in this section.

Table 15-148. CM_GPIO_DATA_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 15-148. CM_GPIO_DATA_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

15.10.6.1 GPADAT Register (Offset = 0h) [Reset = 0h]

GPADAT is shown in Figure 15-132 and described in Table 15-149.

Return to the Summary Table.

GPIO A Data Register (GPIO0 to 31)
Reading this register indicates the current status of the GPIO pin, irrespective of which mode the pin is in. Writing to this register will set the GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero.
DESIGNER NOTE:
[1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register.

Figure 15-132. GPADAT Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-149. GPADAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	Data Register for this pin Reset type: SYSRSn
30	GPIO30	R/W	0h	Data Register for this pin Reset type: SYSRSn
29	GPIO29	R/W	0h	Data Register for this pin Reset type: SYSRSn
28	GPIO28	R/W	0h	Data Register for this pin Reset type: SYSRSn
27	GPIO27	R/W	0h	Data Register for this pin Reset type: SYSRSn
26	GPIO26	R/W	0h	Data Register for this pin Reset type: SYSRSn
25	GPIO25	R/W	0h	Data Register for this pin Reset type: SYSRSn
24	GPIO24	R/W	0h	Data Register for this pin Reset type: SYSRSn
23	GPIO23	R/W	0h	Data Register for this pin Reset type: SYSRSn
22	GPIO22	R/W	0h	Data Register for this pin Reset type: SYSRSn
21	GPIO21	R/W	0h	Data Register for this pin Reset type: SYSRSn

Table 15-149. GPADAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	GPIO20	R/W	0h	Data Register for this pin Reset type: SYSRSn
19	GPIO19	R/W	0h	Data Register for this pin Reset type: SYSRSn
18	GPIO18	R/W	0h	Data Register for this pin Reset type: SYSRSn
17	GPIO17	R/W	0h	Data Register for this pin Reset type: SYSRSn
16	GPIO16	R/W	0h	Data Register for this pin Reset type: SYSRSn
15	GPIO15	R/W	0h	Data Register for this pin Reset type: SYSRSn
14	GPIO14	R/W	0h	Data Register for this pin Reset type: SYSRSn
13	GPIO13	R/W	0h	Data Register for this pin Reset type: SYSRSn
12	GPIO12	R/W	0h	Data Register for this pin Reset type: SYSRSn
11	GPIO11	R/W	0h	Data Register for this pin Reset type: SYSRSn
10	GPIO10	R/W	0h	Data Register for this pin Reset type: SYSRSn
9	GPIO9	R/W	0h	Data Register for this pin Reset type: SYSRSn
8	GPIO8	R/W	0h	Data Register for this pin Reset type: SYSRSn
7	GPIO7	R/W	0h	Data Register for this pin Reset type: SYSRSn
6	GPIO6	R/W	0h	Data Register for this pin Reset type: SYSRSn
5	GPIO5	R/W	0h	Data Register for this pin Reset type: SYSRSn
4	GPIO4	R/W	0h	Data Register for this pin Reset type: SYSRSn
3	GPIO3	R/W	0h	Data Register for this pin Reset type: SYSRSn
2	GPIO2	R/W	0h	Data Register for this pin Reset type: SYSRSn
1	GPIO1	R/W	0h	Data Register for this pin Reset type: SYSRSn
0	GPIO0	R/W	0h	Data Register for this pin Reset type: SYSRSn

15.10.6.2 GPASET Register (Offset = 4h) [Reset = 0h]

GPASET is shown in Figure 15-133 and described in Table 15-150.

Return to the Summary Table.

GPIO A Data Set Register (GPIO0 to 31)
Writing a 1 will force GPIO output data latch to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-133. GPASET Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-150. GPASET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
30	GPIO30	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
29	GPIO29	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
28	GPIO28	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
27	GPIO27	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
26	GPIO26	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
25	GPIO25	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
24	GPIO24	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
23	GPIO23	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
22	GPIO22	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
21	GPIO21	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
20	GPIO20	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
19	GPIO19	R/W	0h	Output Set bit for this pin Reset type: SYSRSn

Table 15-150. GPASET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO18	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
17	GPIO17	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
16	GPIO16	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
15	GPIO15	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
14	GPIO14	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
13	GPIO13	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
12	GPIO12	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
11	GPIO11	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
10	GPIO10	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
9	GPIO9	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
8	GPIO8	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
7	GPIO7	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
6	GPIO6	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
5	GPIO5	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
4	GPIO4	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
3	GPIO3	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
2	GPIO2	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
1	GPIO1	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
0	GPIO0	R/W	0h	Output Set bit for this pin Reset type: SYSRSn

15.10.6.3 GPACLEAR Register (Offset = 8h) [Reset = 0h]

GPACLEAR is shown in Figure 15-134 and described in Table 15-151.

Return to the Summary Table.

GPIO A Data Clear Register (GPIO0 to 31)
Writing a 1 will force GPIO0 output data latch to 0.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-134. GPACLEAR Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-151. GPACLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
30	GPIO30	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
29	GPIO29	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
28	GPIO28	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
27	GPIO27	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
26	GPIO26	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
25	GPIO25	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
24	GPIO24	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
23	GPIO23	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
22	GPIO22	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
21	GPIO21	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
20	GPIO20	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
19	GPIO19	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn

Table 15-151. GPACLEAR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO18	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
17	GPIO17	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
16	GPIO16	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
15	GPIO15	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
14	GPIO14	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
13	GPIO13	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
12	GPIO12	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
11	GPIO11	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
10	GPIO10	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
9	GPIO9	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
8	GPIO8	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
7	GPIO7	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
6	GPIO6	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
5	GPIO5	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
4	GPIO4	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
3	GPIO3	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
2	GPIO2	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
1	GPIO1	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
0	GPIO0	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn

15.10.6.4 GPATOGGLE Register (Offset = Ch) [Reset = 0h]

GPATOGGLE is shown in Figure 15-135 and described in Table 15-152.

Return to the Summary Table.

GPIO A Data Toggle Register (GPIO0 to 31)
Writing a 1 will toggle GPIO0 output data latch 1 to 0 or 0 to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-135. GPATOGGLE Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-152. GPATOGGLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
30	GPIO30	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
29	GPIO29	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
28	GPIO28	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
27	GPIO27	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
26	GPIO26	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
25	GPIO25	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
24	GPIO24	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
23	GPIO23	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
22	GPIO22	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
21	GPIO21	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
20	GPIO20	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
19	GPIO19	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn

Table 15-152. GPATOGGLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO18	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
17	GPIO17	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
16	GPIO16	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
15	GPIO15	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
14	GPIO14	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
13	GPIO13	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
12	GPIO12	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
11	GPIO11	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
10	GPIO10	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
9	GPIO9	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
8	GPIO8	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
7	GPIO7	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
6	GPIO6	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
5	GPIO5	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
4	GPIO4	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
3	GPIO3	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
2	GPIO2	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
1	GPIO1	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
0	GPIO0	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn

15.10.6.5 GPBDAT Register (Offset = 10h) [Reset = 0h]

GPBDAT is shown in Figure 15-136 and described in Table 15-153.

Return to the Summary Table.

GPIO B Data Register (GPIO32 to 63)
Reading this register indicates the current status of the GPIO pin, irrespective of which mode the pin is in. Writing to this register will set the GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero.
DESIGNER NOTE:
[1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register.

Figure 15-136. GPBDAT Register

31	30	29	28	27	26	25	24
GPIO63	GPIO62	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	GPIO38	GPIO37	GPIO36	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-153. GPBDAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO63	R/W	0h	Data Register for this pin Reset type: SYSRSn
30	GPIO62	R/W	0h	Data Register for this pin Reset type: SYSRSn
29	GPIO61	R/W	0h	Data Register for this pin Reset type: SYSRSn
28	GPIO60	R/W	0h	Data Register for this pin Reset type: SYSRSn
27	GPIO59	R/W	0h	Data Register for this pin Reset type: SYSRSn
26	GPIO58	R/W	0h	Data Register for this pin Reset type: SYSRSn
25	GPIO57	R/W	0h	Data Register for this pin Reset type: SYSRSn
24	GPIO56	R/W	0h	Data Register for this pin Reset type: SYSRSn
23	GPIO55	R/W	0h	Data Register for this pin Reset type: SYSRSn
22	GPIO54	R/W	0h	Data Register for this pin Reset type: SYSRSn
21	GPIO53	R/W	0h	Data Register for this pin Reset type: SYSRSn

Table 15-153. GPBDAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	GPIO52	R/W	0h	Data Register for this pin Reset type: SYSRSn
19	GPIO51	R/W	0h	Data Register for this pin Reset type: SYSRSn
18	GPIO50	R/W	0h	Data Register for this pin Reset type: SYSRSn
17	GPIO49	R/W	0h	Data Register for this pin Reset type: SYSRSn
16	GPIO48	R/W	0h	Data Register for this pin Reset type: SYSRSn
15	GPIO47	R/W	0h	Data Register for this pin Reset type: SYSRSn
14	GPIO46	R/W	0h	Data Register for this pin Reset type: SYSRSn
13	GPIO45	R/W	0h	Data Register for this pin Reset type: SYSRSn
12	GPIO44	R/W	0h	Data Register for this pin Reset type: SYSRSn
11	GPIO43	R/W	0h	Data Register for this pin Reset type: SYSRSn
10	GPIO42	R/W	0h	Data Register for this pin Reset type: SYSRSn
9	GPIO41	R/W	0h	Data Register for this pin Reset type: SYSRSn
8	GPIO40	R/W	0h	Data Register for this pin Reset type: SYSRSn
7	GPIO39	R/W	0h	Data Register for this pin Reset type: SYSRSn
6	GPIO38	R/W	0h	Data Register for this pin Reset type: SYSRSn
5	GPIO37	R/W	0h	Data Register for this pin Reset type: SYSRSn
4	GPIO36	R/W	0h	Data Register for this pin Reset type: SYSRSn
3	GPIO35	R/W	0h	Data Register for this pin Reset type: SYSRSn
2	GPIO34	R/W	0h	Data Register for this pin Reset type: SYSRSn
1	GPIO33	R/W	0h	Data Register for this pin Reset type: SYSRSn
0	GPIO32	R/W	0h	Data Register for this pin Reset type: SYSRSn

15.10.6.6 GPBSET Register (Offset = 14h) [Reset = 0h]

GPBSET is shown in Figure 15-137 and described in Table 15-154.

Return to the Summary Table.

GPIO B Data Set Register (GPIO32 to 63)
Writing a 1 will force GPIO output data latch to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-137. GPBSET Register

31	30	29	28	27	26	25	24
GPIO63	GPIO62	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	GPIO38	GPIO37	GPIO36	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-154. GPBSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO63	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
30	GPIO62	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
29	GPIO61	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
28	GPIO60	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
27	GPIO59	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
26	GPIO58	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
25	GPIO57	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
24	GPIO56	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
23	GPIO55	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
22	GPIO54	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
21	GPIO53	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
20	GPIO52	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
19	GPIO51	R/W	0h	Output Set bit for this pin Reset type: SYSRSn

Table 15-154. GPBSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO50	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
17	GPIO49	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
16	GPIO48	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
15	GPIO47	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
14	GPIO46	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
13	GPIO45	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
12	GPIO44	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
11	GPIO43	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
10	GPIO42	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
9	GPIO41	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
8	GPIO40	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
7	GPIO39	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
6	GPIO38	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
5	GPIO37	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
4	GPIO36	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
3	GPIO35	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
2	GPIO34	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
1	GPIO33	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
0	GPIO32	R/W	0h	Output Set bit for this pin Reset type: SYSRSn

15.10.6.7 GPBCLEAR Register (Offset = 18h) [Reset = 0h]

GPBCLEAR is shown in Figure 15-138 and described in Table 15-155.

Return to the Summary Table.

GPIO B Data Clear Register (GPIO32 to 63)
Writing a 1 will force GPIO0 output data latch to 0.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-138. GPBCLEAR Register

31	30	29	28	27	26	25	24
GPIO63	GPIO62	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	GPIO38	GPIO37	GPIO36	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-155. GPBCLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO63	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
30	GPIO62	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
29	GPIO61	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
28	GPIO60	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
27	GPIO59	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
26	GPIO58	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
25	GPIO57	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
24	GPIO56	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
23	GPIO55	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
22	GPIO54	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
21	GPIO53	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
20	GPIO52	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
19	GPIO51	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn

Table 15-155. GPBCLEAR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO50	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
17	GPIO49	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
16	GPIO48	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
15	GPIO47	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
14	GPIO46	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
13	GPIO45	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
12	GPIO44	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
11	GPIO43	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
10	GPIO42	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
9	GPIO41	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
8	GPIO40	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
7	GPIO39	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
6	GPIO38	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
5	GPIO37	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
4	GPIO36	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
3	GPIO35	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
2	GPIO34	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
1	GPIO33	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
0	GPIO32	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn

15.10.6.8 GPBTOGGLE Register (Offset = 1Ch) [Reset = 0h]

GPBTOGGLE is shown in Figure 15-139 and described in Table 15-156.

Return to the Summary Table.

GPIO B Data Toggle Register (GPIO32 to 63)
Writing a 1 will toggle GPIO0 output data latch 1 to 0 or 0 to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-139. GPBTOGGLE Register

31	30	29	28	27	26	25	24
GPIO63	GPIO62	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	GPIO38	GPIO37	GPIO36	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-156. GPBTOGGLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO63	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
30	GPIO62	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
29	GPIO61	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
28	GPIO60	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
27	GPIO59	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
26	GPIO58	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
25	GPIO57	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
24	GPIO56	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
23	GPIO55	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
22	GPIO54	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
21	GPIO53	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
20	GPIO52	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
19	GPIO51	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn

Table 15-156. GPBTOGGLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO50	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
17	GPIO49	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
16	GPIO48	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
15	GPIO47	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
14	GPIO46	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
13	GPIO45	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
12	GPIO44	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
11	GPIO43	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
10	GPIO42	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
9	GPIO41	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
8	GPIO40	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
7	GPIO39	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
6	GPIO38	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
5	GPIO37	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
4	GPIO36	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
3	GPIO35	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
2	GPIO34	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
1	GPIO33	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
0	GPIO32	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn

15.10.6.9 GPCDAT Register (Offset = 20h) [Reset = 0h]

GPCDAT is shown in Figure 15-140 and described in Table 15-157.

Return to the Summary Table.

GPIO C Data Register (GPIO64 to 95)
Reading this register indicates the current status of the GPIO pin, irrespective of which mode the pin is in. Writing to this register will set the GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero.
DESIGNER NOTE:
[1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the

Figure 15-140. GPCDAT Register

31	30	29	28	27	26	25	24
GPIO95	GPIO94	GPIO93	GPIO92	GPIO91	GPIO90	GPIO89	GPIO88
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO87	GPIO86	GPIO85	GPIO84	GPIO83	GPIO82	GPIO81	GPIO80
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO79	GPIO78	GPIO77	GPIO76	GPIO75	GPIO74	GPIO73	GPIO72
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO71	GPIO70	GPIO69	GPIO68	GPIO67	GPIO66	GPIO65	GPIO64
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-157. GPCDAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO95	R/W	0h	Data Register for this pin Reset type: SYSRSn
30	GPIO94	R/W	0h	Data Register for this pin Reset type: SYSRSn
29	GPIO93	R/W	0h	Data Register for this pin Reset type: SYSRSn
28	GPIO92	R/W	0h	Data Register for this pin Reset type: SYSRSn
27	GPIO91	R/W	0h	Data Register for this pin Reset type: SYSRSn
26	GPIO90	R/W	0h	Data Register for this pin Reset type: SYSRSn
25	GPIO89	R/W	0h	Data Register for this pin Reset type: SYSRSn
24	GPIO88	R/W	0h	Data Register for this pin Reset type: SYSRSn
23	GPIO87	R/W	0h	Data Register for this pin Reset type: SYSRSn
22	GPIO86	R/W	0h	Data Register for this pin Reset type: SYSRSn
21	GPIO85	R/W	0h	Data Register for this pin Reset type: SYSRSn

Table 15-157. GPCDAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	GPIO84	R/W	0h	Data Register for this pin Reset type: SYSRSn
19	GPIO83	R/W	0h	Data Register for this pin Reset type: SYSRSn
18	GPIO82	R/W	0h	Data Register for this pin Reset type: SYSRSn
17	GPIO81	R/W	0h	Data Register for this pin Reset type: SYSRSn
16	GPIO80	R/W	0h	Data Register for this pin Reset type: SYSRSn
15	GPIO79	R/W	0h	Data Register for this pin Reset type: SYSRSn
14	GPIO78	R/W	0h	Data Register for this pin Reset type: SYSRSn
13	GPIO77	R/W	0h	Data Register for this pin Reset type: SYSRSn
12	GPIO76	R/W	0h	Data Register for this pin Reset type: SYSRSn
11	GPIO75	R/W	0h	Data Register for this pin Reset type: SYSRSn
10	GPIO74	R/W	0h	Data Register for this pin Reset type: SYSRSn
9	GPIO73	R/W	0h	Data Register for this pin Reset type: SYSRSn
8	GPIO72	R/W	0h	Data Register for this pin Reset type: SYSRSn
7	GPIO71	R/W	0h	Data Register for this pin Reset type: SYSRSn
6	GPIO70	R/W	0h	Data Register for this pin Reset type: SYSRSn
5	GPIO69	R/W	0h	Data Register for this pin Reset type: SYSRSn
4	GPIO68	R/W	0h	Data Register for this pin Reset type: SYSRSn
3	GPIO67	R/W	0h	Data Register for this pin Reset type: SYSRSn
2	GPIO66	R/W	0h	Data Register for this pin Reset type: SYSRSn
1	GPIO65	R/W	0h	Data Register for this pin Reset type: SYSRSn
0	GPIO64	R/W	0h	Data Register for this pin Reset type: SYSRSn

15.10.6.10 GPCSET Register (Offset = 24h) [Reset = 0h]

GPCSET is shown in Figure 15-141 and described in Table 15-158.

Return to the Summary Table.

GPIO C Data Set Register (GPIO64 to 95)
Writing a 1 will force GPIO output data latch to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-141. GPCSET Register

31	30	29	28	27	26	25	24
GPIO95	GPIO94	GPIO93	GPIO92	GPIO91	GPIO90	GPIO89	GPIO88
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO87	GPIO86	GPIO85	GPIO84	GPIO83	GPIO82	GPIO81	GPIO80
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO79	GPIO78	GPIO77	GPIO76	GPIO75	GPIO74	GPIO73	GPIO72
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO71	GPIO70	GPIO69	GPIO68	GPIO67	GPIO66	GPIO65	GPIO64
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-158. GPCSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO95	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
30	GPIO94	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
29	GPIO93	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
28	GPIO92	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
27	GPIO91	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
26	GPIO90	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
25	GPIO89	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
24	GPIO88	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
23	GPIO87	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
22	GPIO86	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
21	GPIO85	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
20	GPIO84	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
19	GPIO83	R/W	0h	Output Set bit for this pin Reset type: SYSRSn

Table 15-158. GPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO82	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
17	GPIO81	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
16	GPIO80	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
15	GPIO79	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
14	GPIO78	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
13	GPIO77	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
12	GPIO76	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
11	GPIO75	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
10	GPIO74	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
9	GPIO73	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
8	GPIO72	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
7	GPIO71	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
6	GPIO70	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
5	GPIO69	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
4	GPIO68	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
3	GPIO67	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
2	GPIO66	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
1	GPIO65	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
0	GPIO64	R/W	0h	Output Set bit for this pin Reset type: SYSRSn

15.10.6.11 GPCCLEAR Register (Offset = 28h) [Reset = 0h]

GPCCLEAR is shown in Figure 15-142 and described in Table 15-159.

Return to the Summary Table.

GPIO C Data Clear Register (GPIO64 to 95)
Writing a 1 will force GPIO0 output data latch to 0.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-142. GPCCLEAR Register

31	30	29	28	27	26	25	24
GPIO95	GPIO94	GPIO93	GPIO92	GPIO91	GPIO90	GPIO89	GPIO88
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO87	GPIO86	GPIO85	GPIO84	GPIO83	GPIO82	GPIO81	GPIO80
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO79	GPIO78	GPIO77	GPIO76	GPIO75	GPIO74	GPIO73	GPIO72
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO71	GPIO70	GPIO69	GPIO68	GPIO67	GPIO66	GPIO65	GPIO64
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-159. GPCCLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO95	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
30	GPIO94	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
29	GPIO93	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
28	GPIO92	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
27	GPIO91	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
26	GPIO90	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
25	GPIO89	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
24	GPIO88	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
23	GPIO87	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
22	GPIO86	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
21	GPIO85	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
20	GPIO84	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
19	GPIO83	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn

Table 15-159. GPCCLEAR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO82	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
17	GPIO81	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
16	GPIO80	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
15	GPIO79	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
14	GPIO78	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
13	GPIO77	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
12	GPIO76	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
11	GPIO75	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
10	GPIO74	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
9	GPIO73	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
8	GPIO72	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
7	GPIO71	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
6	GPIO70	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
5	GPIO69	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
4	GPIO68	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
3	GPIO67	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
2	GPIO66	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
1	GPIO65	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
0	GPIO64	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn

15.10.6.12 GPCTOGGLE Register (Offset = 2Ch) [Reset = 0h]

GPCTOGGLE is shown in Figure 15-143 and described in Table 15-160.

Return to the Summary Table.

GPIO C Data Toggle Register (GPIO64 to 95)
Writing a 1 will toggle GPIO0 output data latch 1 to 0 or 0 to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-143. GPCTOGGLE Register

31	30	29	28	27	26	25	24
GPIO95	GPIO94	GPIO93	GPIO92	GPIO91	GPIO90	GPIO89	GPIO88
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO87	GPIO86	GPIO85	GPIO84	GPIO83	GPIO82	GPIO81	GPIO80
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO79	GPIO78	GPIO77	GPIO76	GPIO75	GPIO74	GPIO73	GPIO72
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO71	GPIO70	GPIO69	GPIO68	GPIO67	GPIO66	GPIO65	GPIO64
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-160. GPCTOGGLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO95	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
30	GPIO94	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
29	GPIO93	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
28	GPIO92	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
27	GPIO91	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
26	GPIO90	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
25	GPIO89	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
24	GPIO88	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
23	GPIO87	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
22	GPIO86	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
21	GPIO85	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
20	GPIO84	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
19	GPIO83	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn

Table 15-160. GPCTOGGLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO82	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
17	GPIO81	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
16	GPIO80	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
15	GPIO79	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
14	GPIO78	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
13	GPIO77	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
12	GPIO76	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
11	GPIO75	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
10	GPIO74	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
9	GPIO73	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
8	GPIO72	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
7	GPIO71	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
6	GPIO70	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
5	GPIO69	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
4	GPIO68	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
3	GPIO67	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
2	GPIO66	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
1	GPIO65	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
0	GPIO64	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn

15.10.6.13 GPDDAT Register (Offset = 30h) [Reset = 0h]

GPDDAT is shown in Figure 15-144 and described in Table 15-161.

Return to the Summary Table.

GPIO D Data Register (GPIO96 to 127)
Reading this register indicates the current status of the GPIO pin, irrespective of which mode the pin is in. Writing to this register will set the GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero.
DESIGNER NOTE:
[1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register.

Figure 15-144. GPDDAT Register

31	30	29	28	27	26	25	24
GPIO127	GPIO126	GPIO125	GPIO124	GPIO123	GPIO122	GPIO121	GPIO120
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO119	GPIO118	GPIO117	GPIO116	GPIO115	GPIO114	GPIO113	GPIO112
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO111	GPIO110	GPIO109	GPIO108	GPIO107	GPIO106	GPIO105	GPIO104
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO103	GPIO102	GPIO101	GPIO100	GPIO99	GPIO98	GPIO97	GPIO96
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-161. GPDDAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO127	R/W	0h	Data Register for this pin Reset type: SYSRSn
30	GPIO126	R/W	0h	Data Register for this pin Reset type: SYSRSn
29	GPIO125	R/W	0h	Data Register for this pin Reset type: SYSRSn
28	GPIO124	R/W	0h	Data Register for this pin Reset type: SYSRSn
27	GPIO123	R/W	0h	Data Register for this pin Reset type: SYSRSn
26	GPIO122	R/W	0h	Data Register for this pin Reset type: SYSRSn
25	GPIO121	R/W	0h	Data Register for this pin Reset type: SYSRSn
24	GPIO120	R/W	0h	Data Register for this pin Reset type: SYSRSn
23	GPIO119	R/W	0h	Data Register for this pin Reset type: SYSRSn
22	GPIO118	R/W	0h	Data Register for this pin Reset type: SYSRSn
21	GPIO117	R/W	0h	Data Register for this pin Reset type: SYSRSn

Table 15-161. GPDDAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	GPIO116	R/W	0h	Data Register for this pin Reset type: SYSRSn
19	GPIO115	R/W	0h	Data Register for this pin Reset type: SYSRSn
18	GPIO114	R/W	0h	Data Register for this pin Reset type: SYSRSn
17	GPIO113	R/W	0h	Data Register for this pin Reset type: SYSRSn
16	GPIO112	R/W	0h	Data Register for this pin Reset type: SYSRSn
15	GPIO111	R/W	0h	Data Register for this pin Reset type: SYSRSn
14	GPIO110	R/W	0h	Data Register for this pin Reset type: SYSRSn
13	GPIO109	R/W	0h	Data Register for this pin Reset type: SYSRSn
12	GPIO108	R/W	0h	Data Register for this pin Reset type: SYSRSn
11	GPIO107	R/W	0h	Data Register for this pin Reset type: SYSRSn
10	GPIO106	R/W	0h	Data Register for this pin Reset type: SYSRSn
9	GPIO105	R/W	0h	Data Register for this pin Reset type: SYSRSn
8	GPIO104	R/W	0h	Data Register for this pin Reset type: SYSRSn
7	GPIO103	R/W	0h	Data Register for this pin Reset type: SYSRSn
6	GPIO102	R/W	0h	Data Register for this pin Reset type: SYSRSn
5	GPIO101	R/W	0h	Data Register for this pin Reset type: SYSRSn
4	GPIO100	R/W	0h	Data Register for this pin Reset type: SYSRSn
3	GPIO99	R/W	0h	Data Register for this pin Reset type: SYSRSn
2	GPIO98	R/W	0h	Data Register for this pin Reset type: SYSRSn
1	GPIO97	R/W	0h	Data Register for this pin Reset type: SYSRSn
0	GPIO96	R/W	0h	Data Register for this pin Reset type: SYSRSn

15.10.6.14 GPDSET Register (Offset = 34h) [Reset = 0h]

GPDSET is shown in Figure 15-145 and described in Table 15-162.

Return to the Summary Table.

GPIO D Data Set Register (GPIO96 to 127)
Writing a 1 will force GPIO output data latch to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-145. GPDSET Register

31	30	29	28	27	26	25	24
GPIO127	GPIO126	GPIO125	GPIO124	GPIO123	GPIO122	GPIO121	GPIO120
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO119	GPIO118	GPIO117	GPIO116	GPIO115	GPIO114	GPIO113	GPIO112
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO111	GPIO110	GPIO109	GPIO108	GPIO107	GPIO106	GPIO105	GPIO104
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO103	GPIO102	GPIO101	GPIO100	GPIO99	GPIO98	GPIO97	GPIO96
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-162. GPDSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO127	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
30	GPIO126	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
29	GPIO125	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
28	GPIO124	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
27	GPIO123	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
26	GPIO122	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
25	GPIO121	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
24	GPIO120	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
23	GPIO119	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
22	GPIO118	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
21	GPIO117	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
20	GPIO116	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
19	GPIO115	R/W	0h	Output Set bit for this pin Reset type: SYSRSn

Table 15-162. GPDSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO114	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
17	GPIO113	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
16	GPIO112	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
15	GPIO111	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
14	GPIO110	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
13	GPIO109	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
12	GPIO108	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
11	GPIO107	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
10	GPIO106	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
9	GPIO105	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
8	GPIO104	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
7	GPIO103	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
6	GPIO102	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
5	GPIO101	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
4	GPIO100	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
3	GPIO99	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
2	GPIO98	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
1	GPIO97	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
0	GPIO96	R/W	0h	Output Set bit for this pin Reset type: SYSRSn

15.10.6.15 GPDCLEAR Register (Offset = 38h) [Reset = 0h]

GPDCLEAR is shown in Figure 15-146 and described in Table 15-163.

Return to the Summary Table.

GPIO D Data Clear Register (GPIO96 to 127)
Writing a 1 will force GPIO0 output data latch to 0.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-146. GPDCLEAR Register

31	30	29	28	27	26	25	24
GPIO127	GPIO126	GPIO125	GPIO124	GPIO123	GPIO122	GPIO121	GPIO120
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO119	GPIO118	GPIO117	GPIO116	GPIO115	GPIO114	GPIO113	GPIO112
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO111	GPIO110	GPIO109	GPIO108	GPIO107	GPIO106	GPIO105	GPIO104
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO103	GPIO102	GPIO101	GPIO100	GPIO99	GPIO98	GPIO97	GPIO96
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-163. GPDCLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO127	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
30	GPIO126	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
29	GPIO125	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
28	GPIO124	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
27	GPIO123	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
26	GPIO122	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
25	GPIO121	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
24	GPIO120	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
23	GPIO119	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
22	GPIO118	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
21	GPIO117	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
20	GPIO116	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
19	GPIO115	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn

Table 15-163. GPDCLEAR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO114	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
17	GPIO113	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
16	GPIO112	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
15	GPIO111	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
14	GPIO110	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
13	GPIO109	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
12	GPIO108	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
11	GPIO107	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
10	GPIO106	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
9	GPIO105	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
8	GPIO104	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
7	GPIO103	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
6	GPIO102	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
5	GPIO101	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
4	GPIO100	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
3	GPIO99	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
2	GPIO98	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
1	GPIO97	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
0	GPIO96	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn

15.10.6.16 GPDTOGGLE Register (Offset = 3Ch) [Reset = 0h]

GPDTOGGLE is shown in Figure 15-147 and described in Table 15-164.

Return to the Summary Table.

GPIO D Data Toggle Register (GPIO96 to 127)
Writing a 1 will toggle GPIO0 output data latch 1 to 0 or 0 to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-147. GPDTOGGLE Register

31	30	29	28	27	26	25	24
GPIO127	GPIO126	GPIO125	GPIO124	GPIO123	GPIO122	GPIO121	GPIO120
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO119	GPIO118	GPIO117	GPIO116	GPIO115	GPIO114	GPIO113	GPIO112
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO111	GPIO110	GPIO109	GPIO108	GPIO107	GPIO106	GPIO105	GPIO104
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO103	GPIO102	GPIO101	GPIO100	GPIO99	GPIO98	GPIO97	GPIO96
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-164. GPDTOGGLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO127	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
30	GPIO126	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
29	GPIO125	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
28	GPIO124	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
27	GPIO123	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
26	GPIO122	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
25	GPIO121	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
24	GPIO120	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
23	GPIO119	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
22	GPIO118	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
21	GPIO117	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
20	GPIO116	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
19	GPIO115	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn

Table 15-164. GPDTOGGLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO114	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
17	GPIO113	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
16	GPIO112	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
15	GPIO111	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
14	GPIO110	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
13	GPIO109	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
12	GPIO108	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
11	GPIO107	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
10	GPIO106	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
9	GPIO105	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
8	GPIO104	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
7	GPIO103	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
6	GPIO102	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
5	GPIO101	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
4	GPIO100	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
3	GPIO99	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
2	GPIO98	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
1	GPIO97	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
0	GPIO96	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn

15.10.6.17 GPEDAT Register (Offset = 40h) [Reset = 0h]

GPEDAT is shown in Figure 15-148 and described in Table 15-165.

Return to the Summary Table.

GPIO E Data Register (GPIO128 to 159)
Reading this register indicates the current status of the GPIO pin, irrespective of which mode the pin is in. Writing to this register will set the GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero.
DESIGNER NOTE:
[1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the

Figure 15-148. GPEDAT Register

31	30	29	28	27	26	25	24
GPIO159	GPIO158	GPIO157	GPIO156	GPIO155	GPIO154	GPIO153	GPIO152
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO151	GPIO150	GPIO149	GPIO148	GPIO147	GPIO146	GPIO145	GPIO144
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO143	GPIO142	GPIO141	GPIO140	GPIO139	GPIO138	GPIO137	GPIO136
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO135	GPIO134	GPIO133	GPIO132	GPIO131	GPIO130	GPIO129	GPIO128
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-165. GPEDAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO159	R/W	0h	Data Register for this pin Reset type: SYSRSn
30	GPIO158	R/W	0h	Data Register for this pin Reset type: SYSRSn
29	GPIO157	R/W	0h	Data Register for this pin Reset type: SYSRSn
28	GPIO156	R/W	0h	Data Register for this pin Reset type: SYSRSn
27	GPIO155	R/W	0h	Data Register for this pin Reset type: SYSRSn
26	GPIO154	R/W	0h	Data Register for this pin Reset type: SYSRSn
25	GPIO153	R/W	0h	Data Register for this pin Reset type: SYSRSn
24	GPIO152	R/W	0h	Data Register for this pin Reset type: SYSRSn
23	GPIO151	R/W	0h	Data Register for this pin Reset type: SYSRSn
22	GPIO150	R/W	0h	Data Register for this pin Reset type: SYSRSn
21	GPIO149	R/W	0h	Data Register for this pin Reset type: SYSRSn

Table 15-165. GPEDAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	GPIO148	R/W	0h	Data Register for this pin Reset type: SYSRSn
19	GPIO147	R/W	0h	Data Register for this pin Reset type: SYSRSn
18	GPIO146	R/W	0h	Data Register for this pin Reset type: SYSRSn
17	GPIO145	R/W	0h	Data Register for this pin Reset type: SYSRSn
16	GPIO144	R/W	0h	Data Register for this pin Reset type: SYSRSn
15	GPIO143	R/W	0h	Data Register for this pin Reset type: SYSRSn
14	GPIO142	R/W	0h	Data Register for this pin Reset type: SYSRSn
13	GPIO141	R/W	0h	Data Register for this pin Reset type: SYSRSn
12	GPIO140	R/W	0h	Data Register for this pin Reset type: SYSRSn
11	GPIO139	R/W	0h	Data Register for this pin Reset type: SYSRSn
10	GPIO138	R/W	0h	Data Register for this pin Reset type: SYSRSn
9	GPIO137	R/W	0h	Data Register for this pin Reset type: SYSRSn
8	GPIO136	R/W	0h	Data Register for this pin Reset type: SYSRSn
7	GPIO135	R/W	0h	Data Register for this pin Reset type: SYSRSn
6	GPIO134	R/W	0h	Data Register for this pin Reset type: SYSRSn
5	GPIO133	R/W	0h	Data Register for this pin Reset type: SYSRSn
4	GPIO132	R/W	0h	Data Register for this pin Reset type: SYSRSn
3	GPIO131	R/W	0h	Data Register for this pin Reset type: SYSRSn
2	GPIO130	R/W	0h	Data Register for this pin Reset type: SYSRSn
1	GPIO129	R/W	0h	Data Register for this pin Reset type: SYSRSn
0	GPIO128	R/W	0h	Data Register for this pin Reset type: SYSRSn

15.10.6.18 GPESET Register (Offset = 44h) [Reset = 0h]

GPESET is shown in Figure 15-149 and described in Table 15-166.

Return to the Summary Table.

GPIO E Data Set Register (GPIO128 to 159)
Writing a 1 will force GPIO output data latch to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-149. GPESET Register

31	30	29	28	27	26	25	24
GPIO159	GPIO158	GPIO157	GPIO156	GPIO155	GPIO154	GPIO153	GPIO152
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO151	GPIO150	GPIO149	GPIO148	GPIO147	GPIO146	GPIO145	GPIO144
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO143	GPIO142	GPIO141	GPIO140	GPIO139	GPIO138	GPIO137	GPIO136
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO135	GPIO134	GPIO133	GPIO132	GPIO131	GPIO130	GPIO129	GPIO128
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-166. GPESET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO159	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
30	GPIO158	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
29	GPIO157	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
28	GPIO156	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
27	GPIO155	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
26	GPIO154	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
25	GPIO153	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
24	GPIO152	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
23	GPIO151	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
22	GPIO150	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
21	GPIO149	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
20	GPIO148	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
19	GPIO147	R/W	0h	Output Set bit for this pin Reset type: SYSRSn

Table 15-166. GPESET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO146	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
17	GPIO145	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
16	GPIO144	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
15	GPIO143	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
14	GPIO142	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
13	GPIO141	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
12	GPIO140	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
11	GPIO139	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
10	GPIO138	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
9	GPIO137	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
8	GPIO136	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
7	GPIO135	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
6	GPIO134	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
5	GPIO133	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
4	GPIO132	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
3	GPIO131	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
2	GPIO130	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
1	GPIO129	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
0	GPIO128	R/W	0h	Output Set bit for this pin Reset type: SYSRSn

15.10.6.19 GPECLEAR Register (Offset = 48h) [Reset = 0h]

GPECLEAR is shown in Figure 15-150 and described in Table 15-167.

Return to the Summary Table.

GPIO E Data Clear Register (GPIO128 to 159)
Writing a 1 will force GPIO0 output data latch to 0.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-150. GPECLEAR Register

31	30	29	28	27	26	25	24
GPIO159	GPIO158	GPIO157	GPIO156	GPIO155	GPIO154	GPIO153	GPIO152
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO151	GPIO150	GPIO149	GPIO148	GPIO147	GPIO146	GPIO145	GPIO144
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO143	GPIO142	GPIO141	GPIO140	GPIO139	GPIO138	GPIO137	GPIO136
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO135	GPIO134	GPIO133	GPIO132	GPIO131	GPIO130	GPIO129	GPIO128
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-167. GPECLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO159	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
30	GPIO158	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
29	GPIO157	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
28	GPIO156	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
27	GPIO155	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
26	GPIO154	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
25	GPIO153	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
24	GPIO152	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
23	GPIO151	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
22	GPIO150	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
21	GPIO149	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
20	GPIO148	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
19	GPIO147	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn

Table 15-167. GPECLEAR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO146	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
17	GPIO145	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
16	GPIO144	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
15	GPIO143	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
14	GPIO142	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
13	GPIO141	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
12	GPIO140	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
11	GPIO139	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
10	GPIO138	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
9	GPIO137	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
8	GPIO136	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
7	GPIO135	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
6	GPIO134	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
5	GPIO133	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
4	GPIO132	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
3	GPIO131	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
2	GPIO130	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
1	GPIO129	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
0	GPIO128	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn

15.10.6.20 GPETOGGLE Register (Offset = 4Ch) [Reset = 0h]

GPETOGGLE is shown in Figure 15-151 and described in Table 15-168.

Return to the Summary Table.

GPIO E Data Toggle Register (GPIO128 to 159)
Writing a 1 will toggle GPIO0 output data latch 1 to 0 or 0 to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-151. GPETOGGLE Register

31	30	29	28	27	26	25	24
GPIO159	GPIO158	GPIO157	GPIO156	GPIO155	GPIO154	GPIO153	GPIO152
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO151	GPIO150	GPIO149	GPIO148	GPIO147	GPIO146	GPIO145	GPIO144
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO143	GPIO142	GPIO141	GPIO140	GPIO139	GPIO138	GPIO137	GPIO136
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO135	GPIO134	GPIO133	GPIO132	GPIO131	GPIO130	GPIO129	GPIO128
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-168. GPETOGGLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO159	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
30	GPIO158	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
29	GPIO157	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
28	GPIO156	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
27	GPIO155	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
26	GPIO154	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
25	GPIO153	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
24	GPIO152	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
23	GPIO151	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
22	GPIO150	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
21	GPIO149	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
20	GPIO148	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
19	GPIO147	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn

Table 15-168. GPETOGGLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO146	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
17	GPIO145	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
16	GPIO144	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
15	GPIO143	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
14	GPIO142	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
13	GPIO141	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
12	GPIO140	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
11	GPIO139	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
10	GPIO138	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
9	GPIO137	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
8	GPIO136	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
7	GPIO135	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
6	GPIO134	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
5	GPIO133	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
4	GPIO132	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
3	GPIO131	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
2	GPIO130	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
1	GPIO129	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
0	GPIO128	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn

15.10.6.21 GPFDAT Register (Offset = 50h) [Reset = 0h]

GPFDAT is shown in Figure 15-152 and described in Table 15-169.

Return to the Summary Table.

GPIO F Data Register (GPIO160 to 191)
Reading this register indicates the current status of the GPIO pin, irrespective of which mode the pin is in. Writing to this register will set the GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero.
DESIGNER NOTE:
[1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the

Figure 15-152. GPFDAT Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	GPIO168
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO167	GPIO166	GPIO165	GPIO164	GPIO163	GPIO162	GPIO161	GPIO160
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-169. GPFDAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	RESERVED	R/W	0h	Reserved
28	RESERVED	R/W	0h	Reserved
27	RESERVED	R/W	0h	Reserved
26	RESERVED	R/W	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23	RESERVED	R/W	0h	Reserved
22	RESERVED	R/W	0h	Reserved
21	RESERVED	R/W	0h	Reserved
20	RESERVED	R/W	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved

Table 15-169. GPFDAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved
8	GPIO168	R/W	0h	Data Register for this pin Reset type: SYSRSn
7	GPIO167	R/W	0h	Data Register for this pin Reset type: SYSRSn
6	GPIO166	R/W	0h	Data Register for this pin Reset type: SYSRSn
5	GPIO165	R/W	0h	Data Register for this pin Reset type: SYSRSn
4	GPIO164	R/W	0h	Data Register for this pin Reset type: SYSRSn
3	GPIO163	R/W	0h	Data Register for this pin Reset type: SYSRSn
2	GPIO162	R/W	0h	Data Register for this pin Reset type: SYSRSn
1	GPIO161	R/W	0h	Data Register for this pin Reset type: SYSRSn
0	GPIO160	R/W	0h	Data Register for this pin Reset type: SYSRSn

15.10.6.22 GPFSET Register (Offset = 54h) [Reset = 0h]

GPFSET is shown in Figure 15-153 and described in Table 15-170.

Return to the Summary Table.

GPIO F Data Set Register (GPIO160 to 191)
Writing a 1 will force GPIO output data latch to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-153. GPFSET Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	GPIO168
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO167	GPIO166	GPIO165	GPIO164	GPIO163	GPIO162	GPIO161	GPIO160
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-170. GPFSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	RESERVED	R/W	0h	Reserved
28	RESERVED	R/W	0h	Reserved
27	RESERVED	R/W	0h	Reserved
26	RESERVED	R/W	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23	RESERVED	R/W	0h	Reserved
22	RESERVED	R/W	0h	Reserved
21	RESERVED	R/W	0h	Reserved
20	RESERVED	R/W	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved

Table 15-170. GPFSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	GPIO168	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
7	GPIO167	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
6	GPIO166	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
5	GPIO165	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
4	GPIO164	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
3	GPIO163	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
2	GPIO162	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
1	GPIO161	R/W	0h	Output Set bit for this pin Reset type: SYSRSn
0	GPIO160	R/W	0h	Output Set bit for this pin Reset type: SYSRSn

15.10.6.23 GPFCLEAR Register (Offset = 58h) [Reset = 0h]

GPFCLEAR is shown in Figure 15-154 and described in Table 15-171.

Return to the Summary Table.

GPIO F Data Clear Register (GPIO160 to 191)
Writing a 1 will force GPIO0 output data latch to 0.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-154. GPFCLEAR Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	GPIO168
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO167	GPIO166	GPIO165	GPIO164	GPIO163	GPIO162	GPIO161	GPIO160
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-171. GPFCLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	RESERVED	R/W	0h	Reserved
28	RESERVED	R/W	0h	Reserved
27	RESERVED	R/W	0h	Reserved
26	RESERVED	R/W	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23	RESERVED	R/W	0h	Reserved
22	RESERVED	R/W	0h	Reserved
21	RESERVED	R/W	0h	Reserved
20	RESERVED	R/W	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved

Table 15-171. GPFCLEAR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	GPIO168	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
7	GPIO167	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
6	GPIO166	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
5	GPIO165	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
4	GPIO164	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
3	GPIO163	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
2	GPIO162	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
1	GPIO161	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn
0	GPIO160	R/W	0h	Output Clear bit for this pin Reset type: SYSRSn

15.10.6.24 GPFTOGGLE Register (Offset = 5Ch) [Reset = 0h]

GPFTOGGLE is shown in Figure 15-155 and described in Table 15-172.

Return to the Summary Table.

GPIO F Data Toggle Register (GPIO160 to 191)
Writing a 1 will toggle GPIO0 output data latch 1 to 0 or 0 to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 15-155. GPFTOGGLE Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	GPIO168
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO167	GPIO166	GPIO165	GPIO164	GPIO163	GPIO162	GPIO161	GPIO160
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-172. GPFTOGGLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	RESERVED	R/W	0h	Reserved
28	RESERVED	R/W	0h	Reserved
27	RESERVED	R/W	0h	Reserved
26	RESERVED	R/W	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23	RESERVED	R/W	0h	Reserved
22	RESERVED	R/W	0h	Reserved
21	RESERVED	R/W	0h	Reserved
20	RESERVED	R/W	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved

Table 15-172. GPFTOGGLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	GPIO168	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
7	GPIO167	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
6	GPIO166	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
5	GPIO165	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
4	GPIO164	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
3	GPIO163	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
2	GPIO162	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
1	GPIO161	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
0	GPIO160	R/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn

15.10.7 CM_GPIO_DATA_READ_REGS Registers

Table 15-173 lists the memory-mapped registers for the CM_GPIO_DATA_READ_REGS registers. All register offset addresses not listed in Table 15-173 should be considered as reserved locations and the register contents should not be modified.

Table 15-173. CM_GPIO_DATA_READ_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	GPADAT_R	GPIO A Data Read Register		Go
4h	GPBDAT_R	GPIO B Data Read Register		Go
8h	GPCDAT_R	GPIO C Data Read Register		Go
Ch	GPDDAT_R	GPIO D Data Read Register		Go
10h	GPEDAT_R	GPIO E Data Read Register		Go
14h	GPFDAT_R	GPIO F Data Read Register		Go
18h	GPGDAT_R	GPIO G Data Read Register		Go
1Ch	GPHDAT_R	GPIO H Data Read Register		Go

Complex bit access types are encoded to fit into small table cells. Table 15-174 shows the codes that are used for access types in this section.

Table 15-174. CM_GPIO_DATA_READ_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

15.10.7.1 GPADAT_R Register (Offset = 0h) [Reset = 0h]

GPADAT_R is shown in Figure 15-156 and described in Table 15-175.

Return to the Summary Table.

GPIO A Data Read Register.
Returns the contents of GPADAT register on a read, write to this register has no effect

Figure 15-156. GPADAT_R Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 15-175. GPADAT_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	A read from this register returns the contents of GPADAT register, writes have no impact Reset type: SYSRSn

15.10.7.2 GPBDAT_R Register (Offset = 4h) [Reset = 0h]

GPBDAT_R is shown in Figure 15-157 and described in Table 15-176.

Return to the Summary Table.

GPIO B Data Read Register.
Returns the contents of GPBDAT register on a read, write to this register has no effect

Figure 15-157. GPBDAT_R Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 15-176. GPBDAT_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	A read from this register returns the contents of GPBDAT register, writes have no impact Reset type: SYSRSn

15.10.7.3 GPCDAT_R Register (Offset = 8h) [Reset = 0h]

GPCDAT_R is shown in Figure 15-158 and described in Table 15-177.

Return to the Summary Table.

GPIO C Data Read Register.
Returns the contents of GPCDAT register on a read, write to this register has no effect

Figure 15-158. GPCDAT_R Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 15-177. GPCDAT_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	A read from this register returns the contents of GPCDAT register, writes have no impact Reset type: SYSRSn

15.10.7.4 GPDDAT_R Register (Offset = Ch) [Reset = 0h]

GPDDAT_R is shown in Figure 15-159 and described in Table 15-178.

Return to the Summary Table.

GPIO D Data Read Register.
Returns the contents of GPDDAT register on a read, write to this register has no effect

Figure 15-159. GPDDAT_R Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 15-178. GPDDAT_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	A read from this register returns the contents of GPDDAT register, writes have no impact Reset type: SYSRSn

15.10.7.5 GPEDAT_R Register (Offset = 10h) [Reset = 0h]

GPEDAT_R is shown in Figure 15-160 and described in Table 15-179.

Return to the Summary Table.

GPIO E Data Read Register.
Returns the contents of GPEDAT register on a read, write to this register has no effect

Figure 15-160. GPEDAT_R Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 15-179. GPEDAT_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	A read from this register returns the contents of GPEDAT register, writes have no impact Reset type: SYSRSn

15.10.7.6 GPFDAT_R Register (Offset = 14h) [Reset = 0h]

GPFDAT_R is shown in Figure 15-161 and described in Table 15-180.

Return to the Summary Table.

GPIO F Data Read Register.
Returns the contents of GPFDAT register on a read, write to this register has no effect

Figure 15-161. GPFDAT_R Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 15-180. GPFDAT_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	A read from this register returns the contents of GPFDAT register, writes have no impact Reset type: SYSRSn

15.10.7.7 GPGDAT_R Register (Offset = 18h) [Reset = 0h]

GPGDAT_R is shown in Figure 15-162 and described in Table 15-181.

Return to the Summary Table.

GPIO G Data Read Register.
Returns the contents of GPGDAT register on a read, write to this register has no effect

Figure 15-162. GPGDAT_R Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 15-181. GPGDAT_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	A read from this register returns the contents of GPGDAT register, writes have no impact Reset type: SYSRSn

15.10.7.8 GPHDAT_R Register (Offset = 1Ch) [Reset = 0h]

GPHDAT_R is shown in Figure 15-163 and described in Table 15-182.

Return to the Summary Table.

GPIO H Data Read Register.
Returns the contents of GPHDAT register on a read, write to this register has no effect

Figure 15-163. GPHDAT_R Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 15-182. GPHDAT_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	A read from this register returns the contents of GPHDAT register, writes have no impact Reset type: SYSRSn

15.10.8 GPIO Registers to Driverlib Functions

Table 15-183. GPIO Registers to Driverlib Functions

File	Driverlib Function
GPACTRL	
gpio.c	GPIO_setQualificationPeriod
GPAQSEL1	
gpio.c	GPIO_setQualificationMode
gpio.c	GPIO_getQualificationMode
GPAQSEL2	
-	See GPAQSEL1
GPAMUX1	
gpio.c	GPIO_setPinConfig
GPAMUX2	
-	See GPAMUX1
GPADIR	
gpio.c	GPIO_setDirectionMode
gpio.c	GPIO_getDirectionMode
GPAPUD	
gpio.c	GPIO_setPadConfig
gpio.c	GPIO_getPadConfig
GPAINV	
gpio.c	GPIO_setPadConfig
gpio.c	GPIO_getPadConfig
GPAODR	
gpio.c	GPIO_setPadConfig
gpio.c	GPIO_getPadConfig
GPAGMUX1	
gpio.c	GPIO_setPinConfig
GPAGMUX2	
-	See GPAGMUX1
GPACSEL1	

Table 15-183. GPIO Registers to Driverlib Functions (continued)

File	Driverlib Function
gpio.c	GPIO_setControllerCore
GPACSEL2	
-	See GPACSEL1
GPACSEL3	
-	See GPACSEL1
GPACSEL4	
-	See GPACSEL1
GPALOCK	
gpio.h	GPIO_lockPortConfig
gpio.h	GPIO_unlockPortConfig
GPACR	
gpio.h	GPIO_commitPortConfig
GPBCTRL	
-	See GPACTRL
GPBQSEL1	
-	See GPAQSEL1
GPBQSEL2	
-	See GPAQSEL1
GPBMUX1	
-	See GPAMUX1
GPBMUX2	
-	See GPAMUX1
GPBDIR	
-	See GPADIR
GPBPUD	
-	See GPAPUD
GPBINV	
-	See GPAINV
GPBODR	
-	See GPAODR
GPBAMSEL	
gpio.c	GPIO_setAnalogMode
GPBGMUX1	
-	See GPAGMUX1
GPBGMUX2	
-	See GPAGMUX1
GPBCSEL1	
-	See GPACSEL1
GPBCSEL2	
-	See GPACSEL1
GPBCSEL3	
-	See GPACSEL1
GPBCSEL4	
-	See GPACSEL1
GPBLOCK	

Table 15-183. GPIO Registers to Driverlib Functions (continued)

File	Driverlib Function
-	See GPALOCK
GPBCR	
-	See GPACR
GPCCTRL	
-	See GPACTRL
GPCQSEL1	
-	See GPAQSEL1
GPCQSEL2	
-	See GPAQSEL1
GPCMUX1	
-	See GPAMUX1
GPCMUX2	
-	See GPAMUX1
GPCDIR	
-	See GPADIR
GPCPUD	
-	See GPAPUD
GPCINV	
-	See GPAINV
GPCODR	
-	See GPAODR
GPCGMUX1	
-	See GPAGMUX1
GPCGMUX2	
-	See GPAGMUX1
GPCCSEL1	
-	See GPACSEL1
GPCCSEL2	
-	See GPACSEL1
GPCCSEL3	
-	See GPACSEL1
GPCCSEL4	
-	See GPACSEL1
GPCLOCK	
-	See GPALOCK
GPCCR	
-	See GPACR
GPDCTRL	
-	See GPACTRL
GPDQSEL1	
-	See GPAQSEL1
GPDQSEL2	
-	See GPAQSEL1
GPDMUX1	
-	See GPAMUX1

Table 15-183. GPIO Registers to Driverlib Functions (continued)

File	Driverlib Function
GPDMUX2	
-	See GPAMUX1
GPDDIR	
-	See GPADIR
GPDPUD	
-	See GPAPUD
GPDINV	
-	See GPAINV
GPDODR	
-	See GPAODR
GPDGMUX1	
-	See GPAGMUX1
GPDGMUX2	
-	See GPAGMUX1
GPDCSEL1	
-	See GPACSEL1
GPDCSEL2	
-	See GPACSEL1
GPDCSEL3	
-	See GPACSEL1
GPDCSEL4	
-	See GPACSEL1
GPDLOCK	
-	See GPALOCK
GPDCR	
-	See GPACR
GPECTRL	
-	See GPACTRL
GPEQSEL1	
-	See GPAQSEL1
GPEQSEL2	
-	See GPAQSEL1
GPEMUX1	
-	See GPAMUX1
GPEMUX2	
-	See GPAMUX1
GPEDIR	
-	See GPADIR
GPEPUD	
-	See GPAPUD
GPEINV	
-	See GPAINV
GPEODR	
-	See GPAODR
GPEGMUX1	

Table 15-183. GPIO Registers to Driverlib Functions (continued)

File	Driverlib Function
-	See GPAGMUX1
GPEGMUX2	
-	See GPAGMUX1
GPECSEL1	
-	See GPACSEL1
GPECSEL2	
-	See GPACSEL1
GPECSEL3	
-	See GPACSEL1
GPECSEL4	
-	See GPACSEL1
GPELOCK	
-	See GPALOCK
GPECR	
-	See GPACR
GPFCTRL	
-	See GPACTRL
GPFQSEL1	
-	See GPAQSEL1
GPFMUX1	
-	See GPAMUX1
GPFDIR	
-	See GPADIR
GPFPUD	
-	See GPAPUD
GPFINV	
-	See GPAINV
GPFODR	
-	See GPAODR
GPFGMUX1	
-	See GPAGMUX1
GPFCSEL1	
-	See GPACSEL1
GPFCSEL2	
-	See GPACSEL1
GPFLOCK	
-	See GPALOCK
GPFCR	
-	See GPACR
GPADAT	
gpio.h	GPIO_readPin
gpio.h	GPIO_readPortData
gpio.h	GPIO_writePortData
GPASET	
gpio.h	GPIO_writePin

Table 15-183. GPIO Registers to Driverlib Functions (continued)

File	Driverlib Function
gpio.h	GPIO_setPortPins
GPACLEAR	
gpio.h	GPIO_writePin
gpio.h	GPIO_clearPortPins
GPATOGGLE	
gpio.h	GPIO_togglePin
gpio.h	GPIO_togglePortPins
GPBDAT	
-	See GPADAT
GPBSET	
-	See GPASET
GPBCLEAR	
-	See GPACLEAR
GPBTOGGLE	
-	See GPATOGGLE
GPCDAT	
-	See GPADAT
GPCSET	
-	See GPASET
GPCCLEAR	
-	See GPACLEAR
GPCTOGGLE	
-	See GPATOGGLE
GPDDAT	
-	See GPADAT
GPDSET	
-	See GPASET
GPDCLEAR	
-	See GPACLEAR
GPDTOGGLE	
-	See GPATOGGLE
GPEDAT	
-	See GPADAT
GPESET	
-	See GPASET
GPECLEAR	
-	See GPACLEAR
GPETOGGLE	
-	See GPATOGGLE
GPFDAT	
-	See GPADAT
GPFSET	
-	See GPASET
GPFCLEAR	
-	See GPACLEAR

Table 15-183. GPIO Registers to Driverlib Functions (continued)

File	Driverlib Function
GPFTOGGLE	
-	See GPATOGGLE
GPADAT_R	
-	
GPBDAT_R	
-	
GPCDAT_R	
-	
GPDDAT_R	
-	
GPEDAT_R	
-	
GPFDAT_R	
-	

Chapter 16

Interprocessor Communication (IPC)

The Interprocessor Communications (IPC) module allows communication between the two CPU subsystems.

16.1 Introduction

This section details the IPC features that each CPU can use to request and share information. The IPC features are:

- Message RAMs
- IPC flags and interrupts
- IPC command registers
- Flash pump semaphore
- Clock configuration semaphore
- Free-running counter

All IPC features are independent of each other, and most do not require any specific data format.

There are also two registers for boot mode and status communication. Please refer to the boot ROM chapter for more information on these registers.

This device has three cores (one M4 core, two C28x (CPU1, CPU2) cores). So, we have three different IPC modules

- CPU1_TO_CPU2 IPC architecture. See Figure 16-1.
- CPUx_TO_CM IPC architecture (where x = 1, 2). See Figure 16-2.

16.2 Message RAMs

There are two dedicated 2kB blocks of message RAM. Each CPU and the DMA have read and write access to one RAM and read-only access to the other RAM, as shown in Table 16-1, Table 16-2, and Table 16-3.

Reading or writing a message RAM does not trigger any events on the remote CPU.

Table 16-1. CPU1-CM IPC Message RAM Read / Write Access

	CPU1	CPU1.DMA	M4	M4-µDMA	Ethernet
CPU1TOCMMSGRAM	Read / Write	Read / Write	Read / Debug writes are allowed	Read	Read
CMTOCPU1MSGRAM	Read / Debug writes are allowed	Read	Read / Write	Read / Write	Read / Write

Table 16-2. CPU2-CM IPC Message RAM Read / Write Access

	CPU2	CPU2.DMA	M4	M4-µDMA	Ethernet
CPU2TOCMMSGRAM	Read / Write	Read / Write	Read / Debug writes are allowed	Read	Read
CMTOCPU2MSGRAM	Read / Debug writes are allowed	Read	Read / Write	Read / Write	Read / Write

Table 16-3. CPU1-CPU2 IPC Message RAM Read / Write Access

	CPU1	CPU1.DMA	CPU2	CPU2.DMA
CPU1TOCPU2MSGRAM	Read / Write	Read / Write	Read / Debug writes are allowed	Read
CPU2TOCPU1MSGRAM	Read / Debug writes are allowed	Read	Read / Write	Read / Write

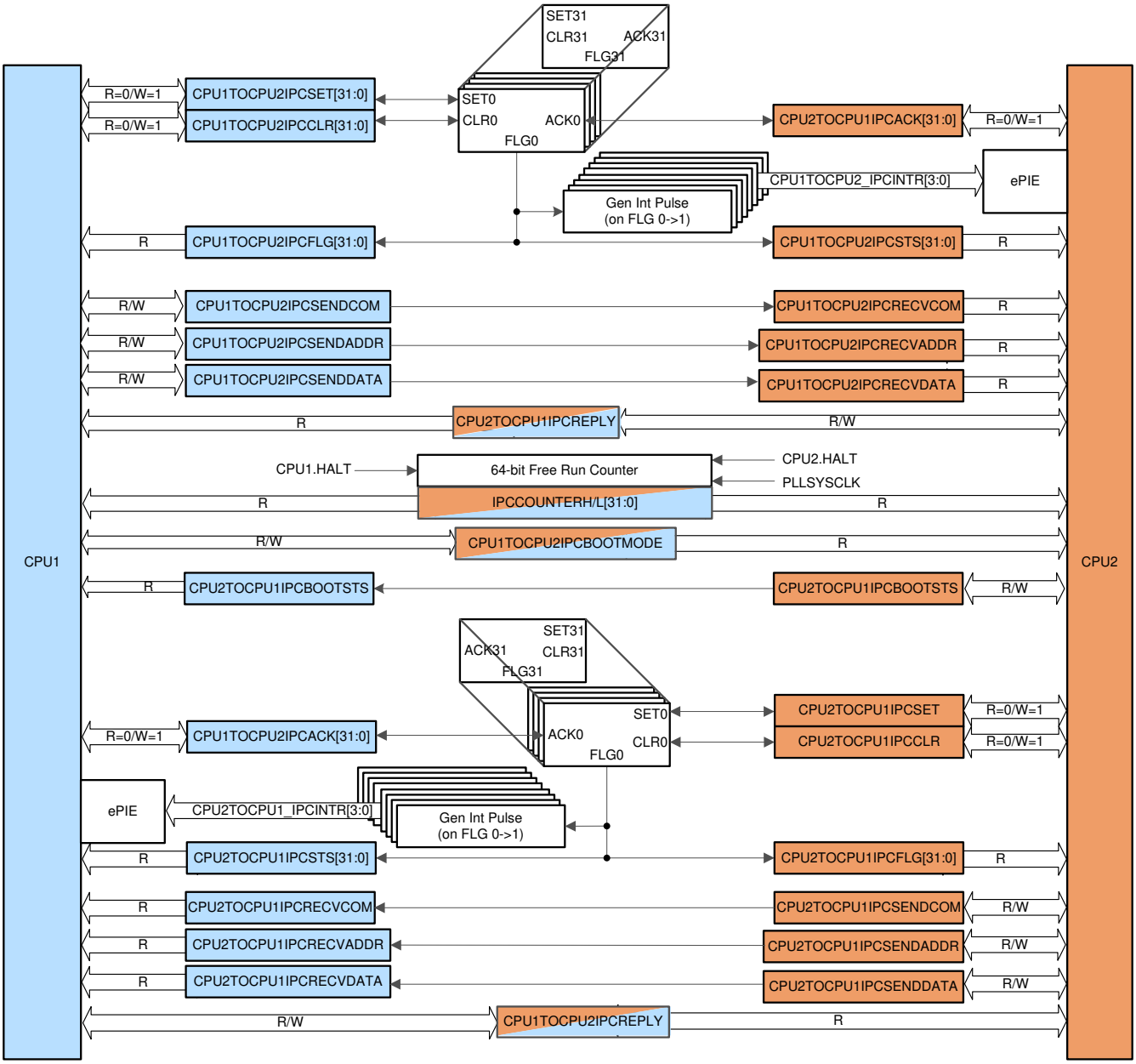


Figure 16-1. CPU1_TO_CPU2 IPC Module

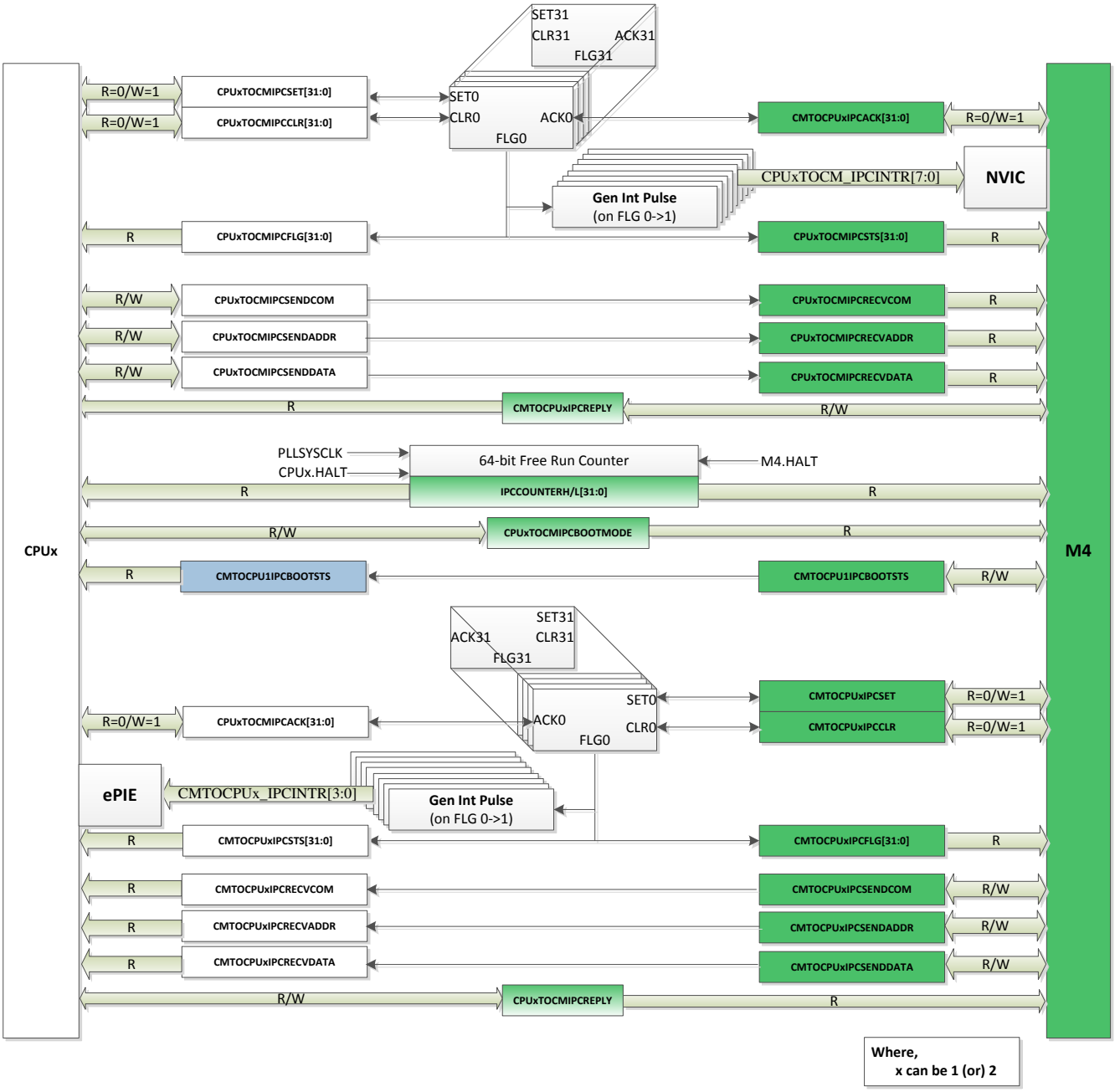


Figure 16-2. CPUx_to_CM IPC Module

16.3 IPC Flags and Interrupts

There are 32 IPC event signals in each direction between the CPU pairs. These signals can be used for flag-based event polling. With the C28x core, four of them (IPC0 - IPC3) can be configured to generate IPC interrupts on the remote CPU. With the Arm® Cortex®-M4 core, eight of them (IPC0-IPC7) can be configured to generate IPC interrupt on the remote CPU.

16.4 IPC Command Registers

The IPC command registers provide a simple and flexible way for the CPUs to exchange more complex messages. Each CPU has eight dedicated registers; four for sending messages and four for receiving messages. The register names were chosen to support a simple command/response protocol, but can be used for any purpose. Only the read/write permissions are determined by hardware; the data format is entirely software-defined.

For sending messages, each CPU has three writable registers and one read-only register. Those same registers are accessible on the remote CPU as three read-only registers and one writable register. Table 16-4 shows the command registers.

Table 16-4. IPC Command Registers

Local Register Name	Local CPU	Remote CPU	Remote Register Name
IPCSENDCOM	R/W	R	IPCRECVCOM
IPCSENDADDR	R/W	R	IPCRECVADDR
IPCSENDDATA	R/W	R	IPCRECVDATA
IPCREPLY	R	R/W	IPCREPLY

16.5 Free-Running Counter

A 64-bit free-running counter is present in the device and can be used to timestamp IPC events between processors. The counter is clocked by PLLSYSCLK and reset by SYSRSn. The counter is implemented as two 32-bit registers, IPCCOUNTERH and IPCCOUNTERL. When IPCCOUNTERL is read, the value of IPCCOUNTERH is saved. A subsequent read to IPCCOUNTERH returns this saved value. Therefore, the user must always read IPCCOUNTERL first then read IPCCOUNTERH next. This design prevents race conditions due to IPCCOUNTERL overflowing between reads of the two registers.

The free-running counter stops only when emulation is suspended (when debugger hits a breakpoint) on all CPUs. If any core is executing, the counter runs.

16.6 IPC Communication Protocol

This section describes the hardware support options for IPC communication between the two CPUs. These options can be used independently or in combination. All flag definitions and data formats are entirely user-defined.

- The flag system supports event-based communication via interrupts and register polling.
  - CPUx can raise an IPC event by writing to any of the 32 bits of its IPCSET register. This sets the corresponding bits in the CPUx IPCFLG register and CPUy IPCSTS register.
  - CPUy can signal its response to the event by setting the appropriate bit in its IPCACK register. This clears the corresponding bits in the CPUx IPCFLG register and the CPUy IPCSTS register.
  - If CPUx needs to cancel an event, it can set the appropriate bit in its IPCCLR register. This has the same effect as CPUy writing to IPCACK.
  - For the CPU1_to_CPU2 IPC module, flags 0–3 (set via IPCSET[3:0]) fire interrupts to the remote CPU. The remote CPU must configure its ePIE module properly in order to receive an IPC interrupt. Flags 4–31 (set via IPCSET[31:4]) do not produce interrupts. Multiple flags can be set, acknowledged, and cleared simultaneously.
  - For CPU1_to_CM (or) CPU2_to_CM IPC module, flags 0-7 (set IPCSET[7:0]) fire interrupts to the remote CPU. The remote CPU must configure its NVIC / ePIE module properly in order to receive an IPC interrupt. Flags 8–31 (set via IPCSET[31:8]) do not produce interrupts. Multiple flags can be set, acknowledged, and cleared simultaneously.
- The command registers support sending several distinct pieces of information. They are named COM, ADDR, DATA, and REPLY for convenience only and can hold whatever data the application needs.
  - CPUx can write data to its IPCSENDCOM, IPCSENDADDR, and IPCSENDDATA registers. CPUy receives these in its IPCRECVCOM, IPCRECVADDR, and IPCRECVDATA registers.
  - CPUy can respond by writing to its IPCREPLY register. CPUx receives this data in its IPCREPLY register.
- There is an additional pair of command-like registers offered for boot-time IPC or any other convenient use — IPCBOOTMODE and IPCBOOTSTS. Both CPUs can read these registers. CPUx can only write to IPCBOOTMODE, and CPUy can only write to IPCBOOTSTS.
- There are two shared memories for passing large amounts of data between the CPUs. Each CPU has a writable memory for sending data and a read-only memory for receiving data.
- Here is an example of how to use these features together. CPUx needs some data from CPUy's LS RAM. The data is at CPUy address 0x9400 and is 0x80 16-bit words long. The protocol could be implemented like this:
  - CPUx writes 0x1 to IPCSENDCOM, defined in software to mean "copy data from address". It writes the address (0x9400) to IPCSENDADDR and the data length (0x80) to IPCSENDDATA.
  - CPUx writes to IPCSET[3] and IPCSET[16]. Here, IPC flag 3 is configured to send an interrupt and IPCSET[16] is defined in software to indicate an incoming command. CPUx begins polling for IPCFLG[3] to go low.
  - CPUy receives the interrupt. In the interrupt handler, it checks IPCSTS, finds that flag 16 is set, and runs a command processor.
  - CPUy reads the command (0x1) from IPCRECVCOM, the address (0x9400) from IPCRECVADDR, and the data length (0x80) from IPCRECVDATA. CPUy then copies the LS RAM data to an empty space in its writable shared memory starting at offset 0x210.
  - CPUy writes the shared memory address (0x210) to its IPCLOCALREPLY register. It then writes to IPCACK[16] and IPCACK[3] to clear the flags and indicate completion of the command. CPUy's work is done.
  - CPUx sees IPCFLG[3] go low. It reads IPCREMOTEREPLY to get the shared memory offset of the copied data (0x210).

16.7 Software

16.7.1 IPC Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/ipc

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

16.7.1.1 IPC basic message passing example with interrupt - C28X_CM

FILE: ipc_ex1_basic_c28x1.c

This example demonstrates how to configure IPC and pass information from C28x to CM core without message queues. It is recommended to run the C28x1 core first, followed by the CM core.

External Connections
- None.

Watch Variables
- pass

16.7.1.2 IPC basic message passing example with interrupt - C28X_CM

FILE: ipc_ex1_basic_cm.c

This example demonstrates how to configure IPC and pass information from C28x to CM core without message queues It is recommended to run the C28x1 core first, followed by the CM core.

External Connections
- None.

Watch Variables
- None.

16.7.1.3 IPC basic message passing example with interrupt - C28X_DUAL

FILE: ipc_ex1_basic_c28x1.c

This example demonstrates how to configure IPC and pass information from C28x1 to C28x2 core without message queues. It is recommended to run the C28x1 core first, followed by the C28x2 core.

External Connections
- None.

Watch Variables
- pass

16.7.1.4 IPC basic message passing example with interrupt - C28X_DUAL

FILE: ipc_ex1_basic_c28x2.c

This example demonstrates how to configure IPC and pass information from C28x1 to C28x2 core without message queues It is recommended to run the C28x1 core first, followed by the C28x2 core.

External Connections
- None.

Watch Variables
- None.

16.7.1.5 IPC message passing example with interrupt and message queue - C28X_CM

FILE: ipc_ex2_msgqueue_c28x1.c

This example demonstrates how to configure IPC and pass information from C28x to CM core with message queues. It is recommended to run the C28x1 core first, followed by the CM core.

External Connections
- None.

Watch Variables
- pass

16.7.1.6 IPC message passing example with interrupt and message queue - C28X_CM

FILE: ipc_ex2_msgqueue_cm.c

This example demonstrates how to configure IPC and pass information from C28x to CM core with message queues. It is recommended to run the C28x1 core first, followed by the CM core.

External Connections
- None.

Watch Variables
- None.

16.7.1.7 IPC message passing example with interrupt and message queue - C28X_DUAL

FILE: ipc_ex2_msgqueue_c28x1.c

This example demonstrates how to configure IPC and pass information from C28x1 to C28x2 core with message queues. It is recommended to run the C28x1 core first, followed by the C28x2 core.

External Connections
- None.

Watch Variables
- pass

16.7.1.8 IPC message passing example with interrupt and message queue - C28X_DUAL

FILE: ipc_ex2_msgqueue_c28x2.c

This example demonstrates how to configure IPC and pass information from C28x1 to C28x2 core with message queues. It is recommended to run the C28x1 core first, followed by the C28x2 core.

External Connections
- None.

Watch Variables
- None.

16.7.1.9 IPC basic message passing example with interrupt - C28X_DUAL

FILE: ipc_ex3_basic_sysconfig_c28x1.c

This example demonstrates how to configure IPC and pass information from C28x1 to C28x2 core without message queues. It is recommended to run the C28x1 core first, followed by the C28x2 core.

External Connections
- None.

Watch Variables
- pass

16.7.1.10 IPC basic message passing example with interrupt - C28X_DUAL

FILE: ipc_ex3_basic_sysconfig_c28x2.c

This example demonstrates how to configure IPC and pass information from C28x1 to C28x2 core without message queues. It is recommended to run the C28x1 core first, followed by the C28x2 core.

External Connections

- None.

Watch Variables

- None.

16.7.1.11 IPC message passing example with interrupt and message queue - C28X_DUAL

FILE: ipc_ex4_msgqueue_sysconfig_c28x1.c

This example demonstrates how to configure IPC and pass information from C28x1 to C28x2 core with message queues. It is recommended to run the C28x1 core first, followed by the C28x2 core.

External Connections

- None.

Watch Variables

- pass

16.7.1.12 IPC message passing example with interrupt and message queue - C28X_DUAL

FILE: ipc_ex4_msgqueue_sysconfig_c28x2.c

This example demonstrates how to configure IPC and pass information from C28x1 to C28x2 core with message queues. It is recommended to run the C28x1 core first, followed by the C28x2 core.

External Connections

- None.

Watch Variables

- None.

16.8 IPC Registers

This section describes the Interprocessor Communication Registers.

16.8.1 IPC Base Address Table (C28)

Table 16-5. IPC Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
Cpu1toCmIpcRegs	CPU1TOCM_IPC_REGS_CPU1VIEW	IPC_CPUXTOCM_BASE	0x0005_CE40	YES	-	-	-	YES
Cpu1toCpu2IpcRegs	CPU1TOCPU2_IPC_REGS_CPU1VIEW	IPC_CPUXTOCPUX_BASE	0x0005_CE00	YES	-	-	-	YES
Cpu2toCmIpcRegs	CPU2TOCM_IPC_REGS_CPU2VIEW	-	0x0005_CE40	-	YES	-	-	YES
Cpu2toCpu1IpcRegs	CPU1TOCPU2_IPC_REGS_CPU2VIEW	-	0x0005_CE00	-	YES	-	-	YES

16.8.2 CM IPC Base Address Table (CM)

Table 16-6. CM IPC Base Address Table (CM)

DriverLib Name	Base Address	µDMA Access	Ethernet DMA Access
IPC_CMTOCPU1_BASE	0x400F_D000	-	-
IPC_CMTOCPU2_BASE	0x400F_D080	-	-

16.8.3 CPU1TOCPU2_IPC_REGS_CPU1VIEW Registers

Table 16-7 lists the memory-mapped registers for the CPU1TOCPU2_IPC_REGS_CPU1VIEW registers. All register offset addresses not listed in Table 16-7 should be considered as reserved locations and the register contents should not be modified.

Table 16-7. CPU1TOCPU2_IPC_REGS_CPU1VIEW Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CPU1TOCPU2IPCACK	CPU1TOCPU2IPCACK Register		Go
2h	CPU2TOCPU1IPCSTS	CPU2TOCPU1IPCSTS Register		Go
4h	CPU1TOCPU2IPCSET	CPU1TOCPU2IPCSET Register		Go
6h	CPU1TOCPU2IPCCLR	CPU1TOCPU2IPCCLR Register		Go
8h	CPU1TOCPU2IPCFLG	CPU1TOCPU2IPCFLG Register		Go
Ch	IPCCOUNTERL	IPCCOUNTERL Register		Go
Eh	IPCCOUNTERH	IPCCOUNTERH Register		Go
10h	CPU1TOCPU2IPCSENDCOM	CPU1TOCPU2IPCSENDCOM Register		Go
12h	CPU1TOCPU2IPCSENDADDR	CPU1TOCPU2IPCSENDADDR Register		Go
14h	CPU1TOCPU2IPCSENDDATA	CPU1TOCPU2IPCSENDDATA Register		Go
16h	CPU2TOCPU1IPCREPLY	CPU2TOCPU1IPCREPLY Register		Go
18h	CPU2TOCPU1IPCRECVCOM	CPU2TOCPU1IPCRECVCOM Register		Go
1Ah	CPU2TOCPU1IPCRECVADDR	CPU2TOCPU1IPCRECVADDR Register		Go
1Ch	CPU2TOCPU1IPCRECVDATA	CPU2TOCPU1IPCRECVDATA Register		Go
1Eh	CPU1TOCPU2IPCREPLY	CPU1TOCPU2IPCREPLY Register		Go
20h	CPU2TOCPU1IPCBOOTSTS	CPU2TOCPU1IPCBOOTSTS Register		Go
22h	CPU1TOCPU2IPCBOOTMODE	CPU1TOCPU2IPCBOOTMODE Register		Go
24h	PUMPREQUEST	PUMPREQUEST Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 16-8 shows the codes that are used for access types in this section.

Table 16-8. CPU1TOCPU2_IPC_REGS_CPU1VIEW Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 16-8. CPU1TOCPU2_IPC_REGS_CPU1VIEW Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

16.8.3.1 CPU1TOCPU2IPCACK Register (Offset = 0h) [Reset = 0h]

CPU1TOCPU2IPCACK is shown in Figure 16-3 and described in Table 16-9.

Return to the Summary Table.

CPU1TOCPU2IPCACK Register

Figure 16-3. CPU1TOCPU2IPCACK Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-9. CPU1TOCPU2IPCACK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC31 bit. Reset type: CPU1.SYSRSn
30	IPC30	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC30 bit. Reset type: CPU1.SYSRSn
29	IPC29	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC29 bit. Reset type: CPU1.SYSRSn
28	IPC28	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC28 bit. Reset type: CPU1.SYSRSn
27	IPC27	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC27 bit. Reset type: CPU1.SYSRSn
26	IPC26	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC26 bit. Reset type: CPU1.SYSRSn
25	IPC25	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC25 bit. Reset type: CPU1.SYSRSn
24	IPC24	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC24 bit. Reset type: CPU1.SYSRSn
23	IPC23	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC23 bit. Reset type: CPU1.SYSRSn
22	IPC22	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC22 bit. Reset type: CPU1.SYSRSn
21	IPC21	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC21 bit. Reset type: CPU1.SYSRSn
20	IPC20	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC20 bit. Reset type: CPU1.SYSRSn
19	IPC19	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC19 bit. Reset type: CPU1.SYSRSn
18	IPC18	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC18 bit. Reset type: CPU1.SYSRSn
17	IPC17	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC17 bit. Reset type: CPU1.SYSRSn

Table 16-9. CPU1TOCPU2IPCACK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	IPC16	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC16 bit. Reset type: CPU1.SYSRSn
15	IPC15	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC15 bit. Reset type: CPU1.SYSRSn
14	IPC14	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC14 bit. Reset type: CPU1.SYSRSn
13	IPC13	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC13 bit. Reset type: CPU1.SYSRSn
12	IPC12	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC12 bit. Reset type: CPU1.SYSRSn
11	IPC11	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC11 bit. Reset type: CPU1.SYSRSn
10	IPC10	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC10 bit. Reset type: CPU1.SYSRSn
9	IPC9	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC9 bit. Reset type: CPU1.SYSRSn
8	IPC8	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC8 bit. Reset type: CPU1.SYSRSn
7	IPC7	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC7 bit. Reset type: CPU1.SYSRSn
6	IPC6	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC6 bit. Reset type: CPU1.SYSRSn
5	IPC5	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC5 bit. Reset type: CPU1.SYSRSn
4	IPC4	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC4 bit. Reset type: CPU1.SYSRSn
3	IPC3	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC3 bit. Reset type: CPU1.SYSRSn
2	IPC2	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC2 bit. Reset type: CPU1.SYSRSn
1	IPC1	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC1 bit. Reset type: CPU1.SYSRSn
0	IPC0	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCPU1IPCFLG.IPC0 bit. Reset type: CPU1.SYSRSn

16.8.3.2 CPU2TOCPU1IPCSTS Register (Offset = 2h) [Reset = 0h]

CPU2TOCPU1IPCSTS is shown in Figure 16-4 and described in Table 16-10.

Return to the Summary Table.

Status of CPU1TOCPU2IPCFLG register

Figure 16-4. CPU2TOCPU1IPCSTS Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 16-10. CPU2TOCPU1IPCSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R	0h	Indicates to CPU1 if the IPC31 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC31 bit. 0: No IPC31 event was set by CPU2 1: An IPC31 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
30	IPC30	R	0h	Indicates to CPU1 if the IPC30 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC30 bit. 0: No IPC30 event was set by CPU2 1: An IPC30 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
29	IPC29	R	0h	Indicates to CPU1 if the IPC29 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC29 bit. 0: No IPC29 event was set by CPU2 1: An IPC29 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
28	IPC28	R	0h	Indicates to CPU1 if the IPC28 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC28 bit. 0: No IPC28 event was set by CPU2 1: An IPC28 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
27	IPC27	R	0h	Indicates to CPU1 if the IPC27 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC27 bit. 0: No IPC27 event was set by CPU2 1: An IPC27 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-10. CPU2TOCPU1IPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	IPC26	R	0h	Indicates to CPU1 if the IPC26 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC26 bit. 0: No IPC26 event was set by CPU2 1: An IPC26 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
25	IPC25	R	0h	Indicates to CPU1 if the IPC25 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC25 bit. 0: No IPC25 event was set by CPU2 1: An IPC25 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
24	IPC24	R	0h	Indicates to CPU1 if the IPC24 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC24 bit. 0: No IPC24 event was set by CPU2 1: An IPC24 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
23	IPC23	R	0h	Indicates to CPU1 if the IPC23 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC23 bit. 0: No IPC23 event was set by CPU2 1: An IPC23 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
22	IPC22	R	0h	Indicates to CPU1 if the IPC22 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC22 bit. 0: No IPC22 event was set by CPU2 1: An IPC22 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
21	IPC21	R	0h	Indicates to CPU1 if the IPC21 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC21 bit. 0: No IPC21 event was set by CPU2 1: An IPC21 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
20	IPC20	R	0h	Indicates to CPU1 if the IPC20 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC20 bit. 0: No IPC20 event was set by CPU2 1: An IPC20 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
19	IPC19	R	0h	Indicates to CPU1 if the IPC19 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC19 bit. 0: No IPC19 event was set by CPU2 1: An IPC19 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-10. CPU2TOCPU1IPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	IPC18	R	0h	Indicates to CPU1 if the IPC18 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC18 bit. 0: No IPC18 event was set by CPU2 1: An IPC18 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
17	IPC17	R	0h	Indicates to CPU1 if the IPC17 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC17 bit. 0: No IPC17 event was set by CPU2 1: An IPC17 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
16	IPC16	R	0h	Indicates to CPU1 if the IPC16 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC16 bit. 0: No IPC16 event was set by CPU2 1: An IPC16 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
15	IPC15	R	0h	Indicates to CPU1 if the IPC15 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC15 bit. 0: No IPC15 event was set by CPU2 1: An IPC15 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
14	IPC14	R	0h	Indicates to CPU1 if the IPC14 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC14 bit. 0: No IPC14 event was set by CPU2 1: An IPC14 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
13	IPC13	R	0h	Indicates to CPU1 if the IPC13 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC13 bit. 0: No IPC13 event was set by CPU2 1: An IPC13 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
12	IPC12	R	0h	Indicates to CPU1 if the IPC12 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC12 bit. 0: No IPC12 event was set by CPU2 1: An IPC12 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
11	IPC11	R	0h	Indicates to CPU1 if the IPC11 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC11 bit. 0: No IPC11 event was set by CPU2 1: An IPC11 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-10. CPU2TOCPU1IPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	IPC10	R	0h	Indicates to CPU1 if the IPC10 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC10 bit. 0: No IPC10 event was set by CPU2 1: An IPC10 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
9	IPC9	R	0h	Indicates to CPU1 if the IPC9 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC9 bit. 0: No IPC9 event was set by CPU2 1: An IPC9 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
8	IPC8	R	0h	Indicates to CPU1 if the IPC8 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC8 bit. 0: No IPC8 event was set by CPU2 1: An IPC8 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
7	IPC7	R	0h	Indicates to CPU1 if the IPC7 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC7 bit. 0: No IPC7 event was set by CPU2 1: An IPC7 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
6	IPC6	R	0h	Indicates to CPU1 if the IPC6 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC6 bit. 0: No IPC6 event was set by CPU2 1: An IPC6 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
5	IPC5	R	0h	Indicates to CPU1 if the IPC5 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC5 bit. 0: No IPC5 event was set by CPU2 1: An IPC5 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
4	IPC4	R	0h	Indicates to CPU1 if the IPC4 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC4 bit. 0: No IPC4 event was set by CPU2 1: An IPC4 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
3	IPC3	R	0h	Indicates to CPU1 if the IPC3 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC3 bit. 0: No IPC3 event was set by CPU2 1: An IPC3 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-10. CPU2TOCPU1IPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	IPC2	R	0h	Indicates to CPU1 if the IPC2 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC2 bit. 0: No IPC2 event was set by CPU2 1: An IPC2 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
1	IPC1	R	0h	Indicates to CPU1 if the IPC1 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC1 bit. 0: No IPC1 event was set by CPU2 1: An IPC1 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
0	IPC0	R	0h	Indicates to CPU1 if the IPC0 event flag was set by CPU2. Reflects the state of CPU2TOCPU1IPCFLG.IPC0 bit. 0: No IPC0 event was set by CPU2 1: An IPC0 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

16.8.3.3 CPU1TOCPU2IPCSET Register (Offset = 4h) [Reset = 0h]

CPU1TOCPU2IPCSET is shown in Figure 16-5 and described in Table 16-11.

Return to the Summary Table.

Set CPU1TOCPU2IPCFLG register

Figure 16-5. CPU1TOCPU2IPCSET Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-11. CPU1TOCPU2IPCSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC31 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
30	IPC30	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC30 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
29	IPC29	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC29 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
28	IPC28	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC28 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
27	IPC27	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC27 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-11. CPU1TOCPU2IPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	IPC26	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC26 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
25	IPC25	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC25 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
24	IPC24	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC24 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
23	IPC23	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC23 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
22	IPC22	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC22 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
21	IPC21	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC21 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
20	IPC20	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC20 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
19	IPC19	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC19 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
18	IPC18	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC18 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
17	IPC17	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC17 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-11. CPU1TOCPU2IPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	IPC16	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC16 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
15	IPC15	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC15 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
14	IPC14	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC14 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
13	IPC13	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC13 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
12	IPC12	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC12 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
11	IPC11	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC11 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
10	IPC10	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC10 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
9	IPC9	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC9 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
8	IPC8	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC8 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
7	IPC7	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC7 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-11. CPU1TOCPU2IPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	IPC6	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC6 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
5	IPC5	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC5 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
4	IPC4	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC4 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
3	IPC3	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC3 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
2	IPC2	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC2 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
1	IPC1	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC1 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
0	IPC0	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCPU2IPCFLG.IPC0 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

16.8.3.4 CPU1TOCPU2IPCCLR Register (Offset = 6h) [Reset = 0h]

CPU1TOCPU2IPCCLR is shown in Figure 16-6 and described in Table 16-12.

Return to the Summary Table.

Clear CPU1TOCPU2IPCFLG register

Figure 16-6. CPU1TOCPU2IPCCLR Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-12. CPU1TOCPU2IPCCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC31 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
30	IPC30	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC30 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
29	IPC29	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC29 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
28	IPC28	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC28 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
27	IPC27	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC27 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-12. CPU1TOCPU2IPCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	IPC26	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC26 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
25	IPC25	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC25 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
24	IPC24	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC24 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
23	IPC23	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC23 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
22	IPC22	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC22 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
21	IPC21	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC21 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
20	IPC20	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC20 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
19	IPC19	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC19 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
18	IPC18	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC18 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
17	IPC17	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC17 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-12. CPU1TOCPU2IPCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	IPC16	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC16 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
15	IPC15	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC15 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
14	IPC14	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC14 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
13	IPC13	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC13 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
12	IPC12	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC12 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
11	IPC11	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC11 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
10	IPC10	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC10 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
9	IPC9	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC9 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
8	IPC8	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC8 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
7	IPC7	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC7 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-12. CPU1TOCPU2IPCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	IPC6	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC6 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
5	IPC5	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC5 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
4	IPC4	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC4 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
3	IPC3	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC3 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
2	IPC2	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC2 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
1	IPC1	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC1 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
0	IPC0	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCPU2IPCFLG.IPC0 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

16.8.3.5 CPU1TOCPU2IPCFLG Register (Offset = 8h) [Reset = 0h]

CPU1TOCPU2IPCFLG is shown in Figure 16-7 and described in Table 16-13.

Return to the Summary Table.

CPU1TOCPU2IPCFLG Register

Figure 16-7. CPU1TOCPU2IPCFLG Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 16-13. CPU1TOCPU2IPCFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R	0h	0: No IPC31 event request to CPU2 1: IPC31 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
30	IPC30	R	0h	0: No IPC30 event request to CPU2 1: IPC30 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
29	IPC29	R	0h	0: No IPC29 event request to CPU2 1: IPC29 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
28	IPC28	R	0h	0: No IPC28 event request to CPU2 1: IPC28 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
27	IPC27	R	0h	0: No IPC27 event request to CPU2 1: IPC27 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
26	IPC26	R	0h	0: No IPC26 event request to CPU2 1: IPC26 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-13. CPU1TOCPU2IPCFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	IPC25	R	0h	0: No IPC25 event request to CPU2 1: IPC25 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
24	IPC24	R	0h	0: No IPC24 event request to CPU2 1: IPC24 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
23	IPC23	R	0h	0: No IPC23 event request to CPU2 1: IPC23 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
22	IPC22	R	0h	0: No IPC22 event request to CPU2 1: IPC22 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
21	IPC21	R	0h	0: No IPC21 event request to CPU2 1: IPC21 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
20	IPC20	R	0h	0: No IPC20 event request to CPU2 1: IPC20 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
19	IPC19	R	0h	0: No IPC19 event request to CPU2 1: IPC19 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
18	IPC18	R	0h	0: No IPC18 event request to CPU2 1: IPC18 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
17	IPC17	R	0h	0: No IPC17 event request to CPU2 1: IPC17 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
16	IPC16	R	0h	0: No IPC16 event request to CPU2 1: IPC16 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
15	IPC15	R	0h	0: No IPC15 event request to CPU2 1: IPC15 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
14	IPC14	R	0h	0: No IPC14 event request to CPU2 1: IPC14 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-13. CPU1TOCPU2IPCFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	IPC13	R	0h	0: No IPC13 event request to CPU2 1: IPC13 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
12	IPC12	R	0h	0: No IPC12 event request to CPU2 1: IPC12 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
11	IPC11	R	0h	0: No IPC11 event request to CPU2 1: IPC11 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
10	IPC10	R	0h	0: No IPC10 event request to CPU2 1: IPC10 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
9	IPC9	R	0h	0: No IPC9 event request to CPU2 1: IPC9 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
8	IPC8	R	0h	0: No IPC8 event request to CPU2 1: IPC8 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
7	IPC7	R	0h	0: No IPC7 event request to CPU2 1: IPC7 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
6	IPC6	R	0h	0: No IPC6 event request to CPU2 1: IPC6 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
5	IPC5	R	0h	0: No IPC5 event request to CPU2 1: IPC5 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
4	IPC4	R	0h	0: No IPC4 event request to CPU2 1: IPC4 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
3	IPC3	R	0h	0: No IPC3 event request to CPU2 1: IPC3 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
2	IPC2	R	0h	0: No IPC2 event request to CPU2 1: IPC2 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-13. CPU1TOCPU2IPCFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	IPC1	R	0h	0: No IPC1 event request to CPU2 1: IPC1 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
0	IPC0	R	0h	0: No IPC0 event request to CPU2 1: IPC0 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

16.8.3.6 IPCCOUNTERL Register (Offset = Ch) [Reset = 0h]

IPCCOUNTERL is shown in Figure 16-8 and described in Table 16-14.

Return to the Summary Table.

IPC Counter Low Register

Figure 16-8. IPCCOUNTERL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT																															
R-0h																															

Table 16-14. IPCCOUNTERL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNT	R	0h	This is the lower 32-bits of free running 64 bit timestamp counter clocked by the PLLSYSCLK. Reset type: CPU1.SYSRSn

16.8.3.7 IPCCOUNTERH Register (Offset = Eh) [Reset = 0h]

IPCCOUNTERH is shown in Figure 16-9 and described in Table 16-15.

Return to the Summary Table.

IPC Counter High Register

Figure 16-9. IPCCOUNTERH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT																															
R-0h																															

Table 16-15. IPCCOUNTERH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNT	R	0h	This is the upper 32-bits of free running 64 bit timestamp counter clocked by the PLLSYSCLK. Reset type: CPU1.SYSRSn

16.8.3.8 CPU1TOCPU2IPCSENDCOM Register (Offset = 10h) [Reset = 0h]

CPU1TOCPU2IPCSENDCOM is shown in Figure 16-10 and described in Table 16-16.

Return to the Summary Table.

CPU1 to CPU2 IPC Command

Figure 16-10. CPU1TOCPU2IPCSENDCOM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMMAND																															
R/W-0h																															

Table 16-16. CPU1TOCPU2IPCSENDCOM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COMMAND	R/W	0h	This is a general purpose register used to send software-defined commands to CPU2 from CPU1. Reset type: CPU1.SYSRSn

16.8.3.9 CPU1TOCPU2IPCSENDADDR Register (Offset = 12h) [Reset = 0h]

CPU1TOCPU2IPCSENDADDR is shown in Figure 16-11 and described in Table 16-17.

Return to the Summary Table.

CPU1 to CPU2 IPC Address

Figure 16-11. CPU1TOCPU2IPCSENDADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS																															
R/W-0h																															

Table 16-17. CPU1TOCPU2IPCSENDADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRESS	R/W	0h	This is a general purpose register used to send software-defined address to CPU2 from CPU1. Reset type: CPU1.SYSRSn

16.8.3.10 CPU1TOCPU2IPCSENDDATA Register (Offset = 14h) [Reset = 0h]

CPU1TOCPU2IPCSENDDATA is shown in Figure 16-12 and described in Table 16-18.

Return to the Summary Table.

CPU1 to CPU2 IPC Data

Figure 16-12. CPU1TOCPU2IPCSENDDATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDATA																															
R/W-0h																															

Table 16-18. CPU1TOCPU2IPCSENDDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WDATA	R/W	0h	This is a general purpose register used to send software-defined data to CPU2 from CPU1. Reset type: CPU1.SYSRSn

16.8.3.11 CPU2TOCPU1IPCREPLY Register (Offset = 16h) [Reset = 0h]

CPU2TOCPU1IPCREPLY is shown in Figure 16-13 and described in Table 16-19.

Return to the Summary Table.

Reply from CPU2 to CPU1TOCPU2IPCSENDCOM command request

Figure 16-13. CPU2TOCPU1IPCREPLY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDATA																															
R/W-0h																															

Table 16-19. CPU2TOCPU1IPCREPLY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RDATA	R/W	0h	This is a general purpose register used to send a reply to CPU1 to CPU2 command from CPU2. Note: This register is not writable from CPU1. Reset type: CPU2.SYSRSn

16.8.3.12 CPU2TOCPU1IPCRECVCOM Register (Offset = 18h) [Reset = 0h]

CPU2TOCPU1IPCRECVCOM is shown in Figure 16-14 and described in Table 16-20.

Return to the Summary Table.

Refelects the value in CPU2TOCPU1IPCSENDCOM Register

Figure 16-14. CPU2TOCPU1IPCRECVCOM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMMAND																															
R-0h																															

Table 16-20. CPU2TOCPU1IPCRECVCOM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COMMAND	R	0h	Refelects the state of CPU2TOCPU1IPCSENDCOM register Reset type: CPU2.SYSRSn

16.8.3.13 CPU2TOCPU1IPCRECVADDR Register (Offset = 1Ah) [Reset = 0h]

CPU2TOCPU1IPCRECVADDR is shown in Figure 16-15 and described in Table 16-21.

Return to the Summary Table.

Refelects the value in CPU2TOCPU1IPCSENDADDR Register

Figure 16-15. CPU2TOCPU1IPCRECVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS																															
R-0h																															

Table 16-21. CPU2TOCPU1IPCRECVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRESS	R	0h	Refelects the state of CPU2TOCPU1IPCSENDADDR register Reset type: CPU2.SYSRSn

16.8.3.14 CPU2TOCPU1IPCRECVDATA Register (Offset = 1Ch) [Reset = 0h]

CPU2TOCPU1IPCRECVDATA is shown in Figure 16-16 and described in Table 16-22.

Return to the Summary Table.

Refelects the value in CPU2TOCPU1IPCSENDDATA Register

Figure 16-16. CPU2TOCPU1IPCRECVDATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDATA																															
R-0h																															

Table 16-22. CPU2TOCPU1IPCRECVDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WDATA	R	0h	Refelects the state of CPU2TOCPU1IPCSENDDATA register Reset type: CPU2.SYSRSn

16.8.3.15 CPU1TOCPU2IPCREPLY Register (Offset = 1Eh) [Reset = 0h]

CPU1TOCPU2IPCREPLY is shown in Figure 16-17 and described in Table 16-23.

Return to the Summary Table.

Reply from CPU1 to CPU2TOCPU1IPCSENDCOM command

Figure 16-17. CPU1TOCPU2IPCREPLY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDATA																															
R/W-0h																															

Table 16-23. CPU1TOCPU2IPCREPLY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RDATA	R/W	0h	This is a general purpose register used to send a reply to CPU2 to CPU1 command from CPU1. Note: This register is not writable from CPU2. Reset type: CPU1.SYSRSn

16.8.3.16 CPU2TOCPU1IPCBOOTSTS Register (Offset = 20h) [Reset = 0h]

CPU2TOCPU1IPCBOOTSTS is shown in Figure 16-18 and described in Table 16-24.

Return to the Summary Table.

CPU2 to CPU1 BOOT Status

Figure 16-18. CPU2TOCPU1IPCBOOTSTS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOOTSTS																															
R/W-0h																															

Table 16-24. CPU2TOCPU1IPCBOOTSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BOOTSTS	R/W	0h	This register is used by CPU2 to pass the boot Status to CPU1. The data format is software-defined. It can only be written by CPU2. Reset type: CPU2.SYSRSn

16.8.3.17 CPU1TOCPU2IPCBOOTMODE Register (Offset = 22h) [Reset = 0h]

CPU1TOCPU2IPCBOOTMODE is shown in Figure 16-19 and described in Table 16-25.

Return to the Summary Table.

CPU1 to CPU2 BOOT Mode setting

Figure 16-19. CPU1TOCPU2IPCBOOTMODE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOOTMODE																															
R/W-0h																															

Table 16-25. CPU1TOCPU2IPCBOOTMODE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BOOTMODE	R/W	0h	This register is used by CPU1 to pass a boot mode information to CPU2. The data format is software-defined. It can only be written by CPU1. Reset type: CPU1.SYSRSn

16.8.3.18 PUMPREQUEST Register (Offset = 24h) [Reset = 0h]

PUMPREQUEST is shown in Figure 16-20 and described in Table 16-26.

Return to the Summary Table.

Flash programming semaphore PUMP request register

Figure 16-20. PUMPREQUEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
R-0/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														SEM	
R-0h														R/W-0h	

Table 16-26. PUMPREQUEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	In order to write to the semaphore bits, 0x5a5a must be written to these key bits at the same time. Otherwise, writes are ignored. The key is cleared immediately after writing, so it must be written again for every semaphore change. Reset type: CPU1.SYSRSn
15-2	RESERVED	R	0h	Reserved
1-0	SEM	R/W	0h	These bits decide which CPU has control of the flash pump, which allows write access to the flash memory. The possible values are: 00: Read-only state. CPU1 has control of the pump, but CPU2 and CM may seize control at any time. 01: CPU2 has exclusive control of the pump and of these semaphore bits. CPU2 can relinquish control by setting the bits back to 00. 10: CPU1 has exclusive control of the pump and of these semaphore bits. CPU1 can relinquish control by setting the bits back to 00. 11: CM has exclusive control of the pump and of these semaphore bits. CM can relinquish control by setting the bits back to 00. Going from 01->10/11 or 10->01/11 or 11->01/10 is not allowed. The semaphore bits [1:0] must be written along with the correct key in bits [31:16]. Note: This field will be reset by the respective CPU resets depending on who owns the PUMP. For example if CPU2 is owning the pump, then CPU2SYSRSN would reset this field. Reset type: CPU1.SYSRSn, CPU2.SYSRSn, CM.RESETn

16.8.4 CPU1TOCPU2_IPC_REGS_CPU2VIEW Registers

Table 16-27 lists the memory-mapped registers for the CPU1TOCPU2_IPC_REGS_CPU2VIEW registers. All register offset addresses not listed in Table 16-27 should be considered as reserved locations and the register contents should not be modified.

Table 16-27. CPU1TOCPU2_IPC_REGS_CPU2VIEW Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CPU2TOCPU1IPCACK	CPU2TOCPU1IPCACK Register		Go
2h	CPU1TOCPU2IPCSTS	CPU1TOCPU2IPCSTS Register		Go
4h	CPU2TOCPU1IPCSET	CPU2TOCPU1IPCSET Register		Go
6h	CPU2TOCPU1IPCCLR	CPU2TOCPU1IPCCLR Register		Go
8h	CPU2TOCPU1IPCFLG	CPU2TOCPU1IPCFLG Register		Go
Ch	IPCCOUNTERL	IPCCOUNTERL Register		Go
Eh	IPCCOUNTERH	IPCCOUNTERH Register		Go
10h	CPU1TOCPU2IPCRECVCOM	CPU1TOCPU2IPCRECVCOM Register		Go
12h	CPU1TOCPU2IPCRECVADDR	CPU1TOCPU2IPCRECVADDR Register		Go
14h	CPU1TOCPU2IPCRECVDATA	CPU1TOCPU2IPCRECVDATA Register		Go
16h	CPU2TOCPU1IPCREPLY	CPU2TOCPU1IPCREPLY Register		Go
18h	CPU2TOCPU1IPCSENDCOM	CPU2TOCPU1IPCSENDCOM Register		Go
1Ah	CPU2TOCPU1IPCSENDADDR	CPU2TOCPU1IPCSENDADDR Register		Go
1Ch	CPU2TOCPU1IPCSENDDATA	CPU2TOCPU1IPCSENDDATA Register		Go
1Eh	CPU1TOCPU2IPCREPLY	CPU1TOCPU2IPCREPLY Register		Go
20h	CPU2TOCPU1IPCBOOTSTS	CPU2TOCPU1IPCBOOTSTS Register		Go
22h	CPU1TOCPU2IPCBOOTMODE	CPU1TOCPU2IPCBOOTMODE Register		Go
24h	PUMPREQUEST	PUMPREQUEST Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 16-28 shows the codes that are used for access types in this section.

Table 16-28. CPU1TOCPU2_IPC_REGS_CPU2VIEW Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 16-28. CPU1TOCPU2_IPC_REGS_CPU2VIEW Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

16.8.4.1 CPU2TOCPU1IPCACK Register (Offset = 0h) [Reset = 0h]

CPU2TOCPU1IPCACK is shown in Figure 16-21 and described in Table 16-29.

Return to the Summary Table.

CPU2TOCPU1IPCACK Register

Figure 16-21. CPU2TOCPU1IPCACK Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-29. CPU2TOCPU1IPCACK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC31 bit. Reset type: CPU2.SYSRSn
30	IPC30	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC30 bit. Reset type: CPU2.SYSRSn
29	IPC29	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC29 bit. Reset type: CPU2.SYSRSn
28	IPC28	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC28 bit. Reset type: CPU2.SYSRSn
27	IPC27	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC27 bit. Reset type: CPU2.SYSRSn
26	IPC26	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC26 bit. Reset type: CPU2.SYSRSn
25	IPC25	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC25 bit. Reset type: CPU2.SYSRSn
24	IPC24	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC24 bit. Reset type: CPU2.SYSRSn
23	IPC23	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC23 bit. Reset type: CPU2.SYSRSn
22	IPC22	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC22 bit. Reset type: CPU2.SYSRSn
21	IPC21	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC21 bit. Reset type: CPU2.SYSRSn
20	IPC20	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC20 bit. Reset type: CPU2.SYSRSn
19	IPC19	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC19 bit. Reset type: CPU2.SYSRSn
18	IPC18	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC18 bit. Reset type: CPU2.SYSRSn
17	IPC17	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC17 bit. Reset type: CPU2.SYSRSn

Table 16-29. CPU2TOCPU1IPCACK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	IPC16	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC16 bit. Reset type: CPU2.SYSRSn
15	IPC15	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC15 bit. Reset type: CPU2.SYSRSn
14	IPC14	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC14 bit. Reset type: CPU2.SYSRSn
13	IPC13	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC13 bit. Reset type: CPU2.SYSRSn
12	IPC12	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC12 bit. Reset type: CPU2.SYSRSn
11	IPC11	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC11 bit. Reset type: CPU2.SYSRSn
10	IPC10	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC10 bit. Reset type: CPU2.SYSRSn
9	IPC9	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC9 bit. Reset type: CPU2.SYSRSn
8	IPC8	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC8 bit. Reset type: CPU2.SYSRSn
7	IPC7	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC7 bit. Reset type: CPU2.SYSRSn
6	IPC6	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC6 bit. Reset type: CPU2.SYSRSn
5	IPC5	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC5 bit. Reset type: CPU2.SYSRSn
4	IPC4	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC4 bit. Reset type: CPU2.SYSRSn
3	IPC3	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC3 bit. Reset type: CPU2.SYSRSn
2	IPC2	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC2 bit. Reset type: CPU2.SYSRSn
1	IPC1	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC1 bit. Reset type: CPU2.SYSRSn
0	IPC0	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCPU2IPCFLG.IPC0 bit. Reset type: CPU2.SYSRSn

16.8.4.2 CPU1TOCPU2IPCSTS Register (Offset = 2h) [Reset = 0h]

CPU1TOCPU2IPCSTS is shown in Figure 16-22 and described in Table 16-30.

Return to the Summary Table.

Status of CPU2TOCPU1IPCFLG register

Figure 16-22. CPU1TOCPU2IPCSTS Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 16-30. CPU1TOCPU2IPCSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R	0h	Indicates to CPU2 if the IPC31 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC31 bit. 0: No IPC31 event was set by CPU1 1: An IPC31 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
30	IPC30	R	0h	Indicates to CPU2 if the IPC30 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC30 bit. 0: No IPC30 event was set by CPU1 1: An IPC30 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
29	IPC29	R	0h	Indicates to CPU2 if the IPC29 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC29 bit. 0: No IPC29 event was set by CPU1 1: An IPC29 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
28	IPC28	R	0h	Indicates to CPU2 if the IPC28 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC28 bit. 0: No IPC28 event was set by CPU1 1: An IPC28 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
27	IPC27	R	0h	Indicates to CPU2 if the IPC27 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC27 bit. 0: No IPC27 event was set by CPU1 1: An IPC27 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-30. CPU1TOCPU2IPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	IPC26	R	0h	Indicates to CPU2 if the IPC26 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC26 bit. 0: No IPC26 event was set by CPU1 1: An IPC26 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
25	IPC25	R	0h	Indicates to CPU2 if the IPC25 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC25 bit. 0: No IPC25 event was set by CPU1 1: An IPC25 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
24	IPC24	R	0h	Indicates to CPU2 if the IPC24 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC24 bit. 0: No IPC24 event was set by CPU1 1: An IPC24 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
23	IPC23	R	0h	Indicates to CPU2 if the IPC23 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC23 bit. 0: No IPC23 event was set by CPU1 1: An IPC23 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
22	IPC22	R	0h	Indicates to CPU2 if the IPC22 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC22 bit. 0: No IPC22 event was set by CPU1 1: An IPC22 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
21	IPC21	R	0h	Indicates to CPU2 if the IPC21 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC21 bit. 0: No IPC21 event was set by CPU1 1: An IPC21 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
20	IPC20	R	0h	Indicates to CPU2 if the IPC20 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC20 bit. 0: No IPC20 event was set by CPU1 1: An IPC20 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
19	IPC19	R	0h	Indicates to CPU2 if the IPC19 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC19 bit. 0: No IPC19 event was set by CPU1 1: An IPC19 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-30. CPU1TOCPU2IPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	IPC18	R	0h	Indicates to CPU2 if the IPC18 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC18 bit. 0: No IPC18 event was set by CPU1 1: An IPC18 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
17	IPC17	R	0h	Indicates to CPU2 if the IPC17 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC17 bit. 0: No IPC17 event was set by CPU1 1: An IPC17 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
16	IPC16	R	0h	Indicates to CPU2 if the IPC16 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC16 bit. 0: No IPC16 event was set by CPU1 1: An IPC16 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
15	IPC15	R	0h	Indicates to CPU2 if the IPC15 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC15 bit. 0: No IPC15 event was set by CPU1 1: An IPC15 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
14	IPC14	R	0h	Indicates to CPU2 if the IPC14 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC14 bit. 0: No IPC14 event was set by CPU1 1: An IPC14 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
13	IPC13	R	0h	Indicates to CPU2 if the IPC13 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC13 bit. 0: No IPC13 event was set by CPU1 1: An IPC13 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
12	IPC12	R	0h	Indicates to CPU2 if the IPC12 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC12 bit. 0: No IPC12 event was set by CPU1 1: An IPC12 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
11	IPC11	R	0h	Indicates to CPU2 if the IPC11 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC11 bit. 0: No IPC11 event was set by CPU1 1: An IPC11 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-30. CPU1TOCPU2IPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	IPC10	R	0h	Indicates to CPU2 if the IPC10 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC10 bit. 0: No IPC10 event was set by CPU1 1: An IPC10 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
9	IPC9	R	0h	Indicates to CPU2 if the IPC9 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC9 bit. 0: No IPC9 event was set by CPU1 1: An IPC9 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
8	IPC8	R	0h	Indicates to CPU2 if the IPC8 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC8 bit. 0: No IPC8 event was set by CPU1 1: An IPC8 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
7	IPC7	R	0h	Indicates to CPU2 if the IPC7 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC7 bit. 0: No IPC7 event was set by CPU1 1: An IPC7 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
6	IPC6	R	0h	Indicates to CPU2 if the IPC6 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC6 bit. 0: No IPC6 event was set by CPU1 1: An IPC6 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
5	IPC5	R	0h	Indicates to CPU2 if the IPC5 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC5 bit. 0: No IPC5 event was set by CPU1 1: An IPC5 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
4	IPC4	R	0h	Indicates to CPU2 if the IPC4 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC4 bit. 0: No IPC4 event was set by CPU1 1: An IPC4 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
3	IPC3	R	0h	Indicates to CPU2 if the IPC3 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC3 bit. 0: No IPC3 event was set by CPU1 1: An IPC3 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-30. CPU1TOCPU2IPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	IPC2	R	0h	Indicates to CPU2 if the IPC2 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC2 bit. 0: No IPC2 event was set by CPU1 1: An IPC2 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
1	IPC1	R	0h	Indicates to CPU2 if the IPC1 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC1 bit. 0: No IPC1 event was set by CPU1 1: An IPC1 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
0	IPC0	R	0h	Indicates to CPU2 if the IPC0 event flag was set by CPU1. Reflects the state of CPU1TOCPU2IPCFLG.IPC0 bit. 0: No IPC0 event was set by CPU1 1: An IPC0 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

16.8.4.3 CPU2TOCPU1IPCSET Register (Offset = 4h) [Reset = 0h]

CPU2TOCPU1IPCSET is shown in Figure 16-23 and described in Table 16-31.

Return to the Summary Table.

Set CPU2TOCPU1IPCFLG register

Figure 16-23. CPU2TOCPU1IPCSET Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-31. CPU2TOCPU1IPCSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC31 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
30	IPC30	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC30 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
29	IPC29	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC29 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
28	IPC28	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC28 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
27	IPC27	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC27 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-31. CPU2TOCPU1IPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	IPC26	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC26 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
25	IPC25	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC25 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
24	IPC24	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC24 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
23	IPC23	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC23 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
22	IPC22	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC22 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
21	IPC21	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC21 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
20	IPC20	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC20 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
19	IPC19	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC19 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
18	IPC18	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC18 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
17	IPC17	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC17 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-31. CPU2TOCPU1IPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	IPC16	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC16 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
15	IPC15	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC15 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
14	IPC14	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC14 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
13	IPC13	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC13 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
12	IPC12	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC12 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
11	IPC11	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC11 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
10	IPC10	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC10 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
9	IPC9	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC9 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
8	IPC8	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC8 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
7	IPC7	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC7 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-31. CPU2TOCPU1IPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	IPC6	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC6 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
5	IPC5	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC5 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
4	IPC4	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC4 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
3	IPC3	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC3 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
2	IPC2	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC2 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
1	IPC1	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC1 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
0	IPC0	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCPU1IPCFLG.IPC0 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

16.8.4.4 CPU2TOCPU1IPCCLR Register (Offset = 6h) [Reset = 0h]

CPU2TOCPU1IPCCLR is shown in Figure 16-24 and described in Table 16-32.

Return to the Summary Table.

Clear CPU2TOCPU1IPCFLG register

Figure 16-24. CPU2TOCPU1IPCCLR Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-32. CPU2TOCPU1IPCCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC31 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
30	IPC30	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC30 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
29	IPC29	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC29 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
28	IPC28	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC28 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
27	IPC27	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC27 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-32. CPU2TOCPU1IPCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	IPC26	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC26 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
25	IPC25	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC25 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
24	IPC24	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC24 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
23	IPC23	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC23 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
22	IPC22	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC22 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
21	IPC21	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC21 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
20	IPC20	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC20 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
19	IPC19	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC19 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
18	IPC18	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC18 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
17	IPC17	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC17 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-32. CPU2TOCPU1IPCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	IPC16	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC16 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
15	IPC15	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC15 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
14	IPC14	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC14 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
13	IPC13	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC13 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
12	IPC12	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC12 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
11	IPC11	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC11 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
10	IPC10	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC10 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
9	IPC9	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC9 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
8	IPC8	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC8 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
7	IPC7	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC7 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-32. CPU2TOCPU1IPCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	IPC6	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC6 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
5	IPC5	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC5 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
4	IPC4	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC4 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
3	IPC3	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC3 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
2	IPC2	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC2 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
1	IPC1	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC1 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
0	IPC0	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCPU1IPCFLG.IPC0 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

16.8.4.5 CPU2TOCPU1IPCFLG Register (Offset = 8h) [Reset = 0h]

CPU2TOCPU1IPCFLG is shown in Figure 16-25 and described in Table 16-33.

Return to the Summary Table.

CPU2TOCPU1IPCFLG Register

Figure 16-25. CPU2TOCPU1IPCFLG Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 16-33. CPU2TOCPU1IPCFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R	0h	0: No IPC31 event request to CPU1 1: IPC31 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
30	IPC30	R	0h	0: No IPC30 event request to CPU1 1: IPC30 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
29	IPC29	R	0h	0: No IPC29 event request to CPU1 1: IPC29 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
28	IPC28	R	0h	0: No IPC28 event request to CPU1 1: IPC28 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
27	IPC27	R	0h	0: No IPC27 event request to CPU1 1: IPC27 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
26	IPC26	R	0h	0: No IPC26 event request to CPU1 1: IPC26 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-33. CPU2TOCPU1IPCFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	IPC25	R	0h	0: No IPC25 event request to CPU1 1: IPC25 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
24	IPC24	R	0h	0: No IPC24 event request to CPU1 1: IPC24 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
23	IPC23	R	0h	0: No IPC23 event request to CPU1 1: IPC23 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
22	IPC22	R	0h	0: No IPC22 event request to CPU1 1: IPC22 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
21	IPC21	R	0h	0: No IPC21 event request to CPU1 1: IPC21 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
20	IPC20	R	0h	0: No IPC20 event request to CPU1 1: IPC20 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
19	IPC19	R	0h	0: No IPC19 event request to CPU1 1: IPC19 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
18	IPC18	R	0h	0: No IPC18 event request to CPU1 1: IPC18 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
17	IPC17	R	0h	0: No IPC17 event request to CPU1 1: IPC17 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
16	IPC16	R	0h	0: No IPC16 event request to CPU1 1: IPC16 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
15	IPC15	R	0h	0: No IPC15 event request to CPU1 1: IPC15 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
14	IPC14	R	0h	0: No IPC14 event request to CPU1 1: IPC14 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-33. CPU2TOCPU1IPCFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	IPC13	R	0h	0: No IPC13 event request to CPU1 1: IPC13 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
12	IPC12	R	0h	0: No IPC12 event request to CPU1 1: IPC12 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
11	IPC11	R	0h	0: No IPC11 event request to CPU1 1: IPC11 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
10	IPC10	R	0h	0: No IPC10 event request to CPU1 1: IPC10 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
9	IPC9	R	0h	0: No IPC9 event request to CPU1 1: IPC9 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
8	IPC8	R	0h	0: No IPC8 event request to CPU1 1: IPC8 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
7	IPC7	R	0h	0: No IPC7 event request to CPU1 1: IPC7 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
6	IPC6	R	0h	0: No IPC6 event request to CPU1 1: IPC6 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
5	IPC5	R	0h	0: No IPC5 event request to CPU1 1: IPC5 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
4	IPC4	R	0h	0: No IPC4 event request to CPU1 1: IPC4 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
3	IPC3	R	0h	0: No IPC3 event request to CPU1 1: IPC3 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
2	IPC2	R	0h	0: No IPC2 event request to CPU1 1: IPC2 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-33. CPU2TOCPU1IPCFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	IPC1	R	0h	0: No IPC1 event request to CPU1 1: IPC1 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
0	IPC0	R	0h	0: No IPC0 event request to CPU1 1: IPC0 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

16.8.4.6 IPCCOUNTERL Register (Offset = Ch) [Reset = 0h]

IPCCOUNTERL is shown in Figure 16-26 and described in Table 16-34.

Return to the Summary Table.

IPC Counter Low Register

Figure 16-26. IPCCOUNTERL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT																															
R-0h																															

Table 16-34. IPCCOUNTERL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNT	R	0h	This is the lower 32-bits of free running 64 bit timestamp counter clocked by the PLLSYSCLK. Reset type: CPU1.SYSRSn

16.8.4.7 IPCCOUNTERH Register (Offset = Eh) [Reset = 0h]

IPCCOUNTERH is shown in Figure 16-27 and described in Table 16-35.

Return to the Summary Table.

IPC Counter High Register

Figure 16-27. IPCCOUNTERH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT																															
R-0h																															

Table 16-35. IPCCOUNTERH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNT	R	0h	This is the upper 32-bits of free running 64 bit timestamp counter clocked by the PLLSYSCLK. Reset type: CPU1.SYSRSn

16.8.4.8 CPU1TOCPU2IPCRECVCOM Register (Offset = 10h) [Reset = 0h]

CPU1TOCPU2IPCRECVCOM is shown in Figure 16-28 and described in Table 16-36.

Return to the Summary Table.

Refelects the value in CPU1TOCPU2IPCSENDCOM Register

Figure 16-28. CPU1TOCPU2IPCRECVCOM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMMAND																															
R-0h																															

Table 16-36. CPU1TOCPU2IPCRECVCOM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COMMAND	R	0h	Refelects the state of CPU1TOCPU2IPCSENDCOM register Reset type: CPU1.SYSRSn

16.8.4.9 CPU1TOCPU2IPCRECVADDR Register (Offset = 12h) [Reset = 0h]

CPU1TOCPU2IPCRECVADDR is shown in Figure 16-29 and described in Table 16-37.

Return to the Summary Table.

Refelects the value in CPU1TOCPU2IPCSENADDR Register

Figure 16-29. CPU1TOCPU2IPCRECVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS																															
R-0h																															

Table 16-37. CPU1TOCPU2IPCRECVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRESS	R	0h	Refelects the state of CPU1TOCPU2IPCSENDADDR register Reset type: CPU1.SYSRSn

16.8.4.10 CPU1TOCPU2IPCRECVDATA Register (Offset = 14h) [Reset = 0h]

CPU1TOCPU2IPCRECVDATA is shown in Figure 16-30 and described in Table 16-38.

Return to the Summary Table.

Refelects the value in CPU1TOCPU2IPCSENDDATA Register

Figure 16-30. CPU1TOCPU2IPCRECVDATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDATA																															
R-0h																															

Table 16-38. CPU1TOCPU2IPCRECVDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WDATA	R	0h	Refelects the state of CPU1TOCPU2IPCSENDDATA register Reset type: CPU1.SYSRSn

16.8.4.11 CPU2TOCPU1IPCREPLY Register (Offset = 16h) [Reset = 0h]

CPU2TOCPU1IPCREPLY is shown in Figure 16-31 and described in Table 16-39.

Return to the Summary Table.

Reply from CPU2 to CPU1TOCPU2IPCSENDCOM command

Figure 16-31. CPU2TOCPU1IPCREPLY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDATA																															
R/W-0h																															

Table 16-39. CPU2TOCPU1IPCREPLY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RDATA	R/W	0h	This is a general purpose register used to send a reply to CPU1 to CPU2 command from CPU2. Note: This register is not writable from CPU1. Reset type: CPU2.SYSRSn

16.8.4.12 CPU2TOCPU1IPCSENDCOM Register (Offset = 18h) [Reset = 0h]

CPU2TOCPU1IPCSENDCOM is shown in Figure 16-32 and described in Table 16-40.

Return to the Summary Table.

CPU2 to CPU1 IPC Command

Figure 16-32. CPU2TOCPU1IPCSENDCOM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMMAND																															
R/W-0h																															

Table 16-40. CPU2TOCPU1IPCSENDCOM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COMMAND	R/W	0h	This is a general purpose register used to send software-defined commands to CPU1 from CPU2. Reset type: CPU2.SYSRSn

16.8.4.13 CPU2TOCPU1IPCSENDADDR Register (Offset = 1Ah) [Reset = 0h]

CPU2TOCPU1IPCSENDADDR is shown in Figure 16-33 and described in Table 16-41.

Return to the Summary Table.

CPU2 to CPU1 IPC Address

Figure 16-33. CPU2TOCPU1IPCSENDADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS																															
R/W-0h																															

Table 16-41. CPU2TOCPU1IPCSENDADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRESS	R/W	0h	This is a general purpose register used to send software-defined address to CPU1 from CPU2. Reset type: CPU2.SYSRSn

16.8.4.14 CPU2TOCPU1IPCSENDDATA Register (Offset = 1Ch) [Reset = 0h]

CPU2TOCPU1IPCSENDDATA is shown in Figure 16-34 and described in Table 16-42.

Return to the Summary Table.

CPU2 to CPU1 IPC Data

Figure 16-34. CPU2TOCPU1IPCSENDDATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDATA																															
R/W-0h																															

Table 16-42. CPU2TOCPU1IPCSENDDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WDATA	R/W	0h	This is a general purpose register used to send software-defined data to CPU1 from CPU2. Reset type: CPU2.SYSRSn

16.8.4.15 CPU1TOCPU2IPCREPLY Register (Offset = 1Eh) [Reset = 0h]

CPU1TOCPU2IPCREPLY is shown in Figure 16-35 and described in Table 16-43.

Return to the Summary Table.

Reply from CPU1 to CPU2TOCPU1IPCSENDCOM command request

Figure 16-35. CPU1TOCPU2IPCREPLY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDATA																															
R/W-0h																															

Table 16-43. CPU1TOCPU2IPCREPLY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RDATA	R/W	0h	This is a general purpose register used to send a reply to CPU2 to CPU1 command from CPU1. Note: This register is not writable from CPU2. Reset type: CPU1.SYSRSn

16.8.4.16 CPU2TOCPU1IPCBOOTSTS Register (Offset = 20h) [Reset = 0h]

CPU2TOCPU1IPCBOOTSTS is shown in Figure 16-36 and described in Table 16-44.

Return to the Summary Table.

CPU2 to CPU1 BOOT Status

Figure 16-36. CPU2TOCPU1IPCBOOTSTS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOOTSTS																															
R/W-0h																															

Table 16-44. CPU2TOCPU1IPCBOOTSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BOOTSTS	R/W	0h	This register is used by CPU2 to pass the boot Status to CPU1. The data format is software-defined. It can only be written by CPU2. Reset type: CPU2.SYSRSn

16.8.4.17 CPU1TOCPU2IPCBOOTMODE Register (Offset = 22h) [Reset = 0h]

CPU1TOCPU2IPCBOOTMODE is shown in Figure 16-37 and described in Table 16-45.

Return to the Summary Table.

CPU1 to CPU2 BOOT Mode setting

Figure 16-37. CPU1TOCPU2IPCBOOTMODE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOOTMODE																															
R/W-0h																															

Table 16-45. CPU1TOCPU2IPCBOOTMODE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BOOTMODE	R/W	0h	This register is used by CPU1 to pass a boot mode information to CPU2. The data format is software-defined. It can only be written by CPU1. Reset type: CPU1.SYSRSn

16.8.4.18 PUMPREQUEST Register (Offset = 24h) [Reset = 0h]

PUMPREQUEST is shown in Figure 16-38 and described in Table 16-46.

Return to the Summary Table.

Flash programming semaphore PUMP request register

Figure 16-38. PUMPREQUEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
R-0/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														SEM	
R-0h														R/W-0h	

Table 16-46. PUMPREQUEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	In order to write to the semaphore bits, 0x5a5a must be written to these key bits at the same time. Otherwise, writes are ignored. The key is cleared immediately after writing, so it must be written again for every semaphore change. Reset type: CPU1.SYSRSn
15-2	RESERVED	R	0h	Reserved
1-0	SEM	R/W	0h	These bits decide which CPU has control of the flash pump, which allows write access to the flash memory. The possible values are: 00: Read-only state. CPU1 has control of the pump, but CPU2 and CM may seize control at any time. 01: CPU2 has exclusive control of the pump and of these semaphore bits. CPU2 can relinquish control by setting the bits back to 00. 10: CPU1 has exclusive control of the pump and of these semaphore bits. CPU1 can relinquish control by setting the bits back to 00. 11: CM has exclusive control of the pump and of these semaphore bits. CM can relinquish control by setting the bits back to 00. Going from 01->10/11 or 10->01/11 or 11->01/10 is not allowed. The semaphore bits [1:0] must be written along with the correct key in bits [31:16]. Note: This field will be reset by the respective CPU resets depending on who owns the PUMP. For example if CPU2 is owning the pump, then CPU2SYSRSN would reset this field. Reset type: CPU1.SYSRSn, CPU2.SYSRSn, CM.RESETn

16.8.5 CPU1TOCM_IPC_REGS_CPU1VIEW Registers

Table 16-47 lists the memory-mapped registers for the CPU1TOCM_IPC_REGS_CPU1VIEW registers. All register offset addresses not listed in Table 16-47 should be considered as reserved locations and the register contents should not be modified.

Table 16-47. CPU1TOCM_IPC_REGS_CPU1VIEW Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CPU1TOCMIPCACK	CPU1TOCMIPCACK Register		Go
2h	CMTOCPU1IPCSTS	CMTOCPU1IPCSTS Register		Go
4h	CPU1TOCMIPCSET	CPU1TOCMIPCSET Register		Go
6h	CPU1TOCMIPCCLR	CPU1TOCMIPCCLR Register		Go
8h	CPU1TOCMIPCFLG	CPU1TOCMIPCFLG Register		Go
Ch	IPCCOUNTERL	IPCCOUNTERL Register		Go
Eh	IPCCOUNTERH	IPCCOUNTERH Register		Go
10h	CPU1TOCMIPCSENDCOM	CPU1TOCMIPCSENDCOM Register		Go
12h	CPU1TOCMIPCSENDADDR	CPU1TOCMIPCSENDADDR Register		Go
14h	CPU1TOCMIPCSENDDATA	CPU1TOCMIPCSENDDATA Register		Go
16h	CMTOCPU1IPCREPLY	CMTOCPU1IPCREPLY Register		Go
18h	CMTOCPU1IPCRECVCOM	CMTOCPU1IPCRECVCOM Register		Go
1Ah	CMTOCPU1IPCRECVADDR	CMTOCPU1IPCRECVADDR Register		Go
1Ch	CMTOCPU1IPCRECVDATA	CMTOCPU1IPCRECVDATA Register		Go
1Eh	CPU1TOCMIPCREPLY	CPU1TOCMIPCREPLY Register		Go
20h	CMTOCPU1IPCBOOTSTS	CMTOCPU1IPCBOOTSTS Register		Go
22h	CPU1TOCMIPCBOOTMODE	CPU1TOCMIPCBOOTMODE Register		Go

Complex bit access types are encoded to fit into small table cells. Table 16-48 shows the codes that are used for access types in this section.

Table 16-48. CPU1TOCM_IPC_REGS_CPU1VIEW Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 16-48. CPU1TOCM_IPC_REGS_CPU1VIEW Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

16.8.5.1 CPU1TOCMIPCACK Register (Offset = 0h) [Reset = 0h]

CPU1TOCMIPCACK is shown in Figure 16-39 and described in Table 16-49.

Return to the Summary Table.

CPU1TOCMIPCACK Register

Figure 16-39. CPU1TOCMIPCACK Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-49. CPU1TOCMIPCACK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC31 bit. Reset type: CPU1.SYSRSn
30	IPC30	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC30 bit. Reset type: CPU1.SYSRSn
29	IPC29	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC29 bit. Reset type: CPU1.SYSRSn
28	IPC28	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC28 bit. Reset type: CPU1.SYSRSn
27	IPC27	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC27 bit. Reset type: CPU1.SYSRSn
26	IPC26	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC26 bit. Reset type: CPU1.SYSRSn
25	IPC25	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC25 bit. Reset type: CPU1.SYSRSn
24	IPC24	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC24 bit. Reset type: CPU1.SYSRSn
23	IPC23	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC23 bit. Reset type: CPU1.SYSRSn
22	IPC22	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC22 bit. Reset type: CPU1.SYSRSn
21	IPC21	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC21 bit. Reset type: CPU1.SYSRSn
20	IPC20	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC20 bit. Reset type: CPU1.SYSRSn
19	IPC19	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC19 bit. Reset type: CPU1.SYSRSn
18	IPC18	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC18 bit. Reset type: CPU1.SYSRSn
17	IPC17	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC17 bit. Reset type: CPU1.SYSRSn

Table 16-49. CPU1TOCMIPCACK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	IPC16	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC16 bit. Reset type: CPU1.SYSRSn
15	IPC15	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC15 bit. Reset type: CPU1.SYSRSn
14	IPC14	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC14 bit. Reset type: CPU1.SYSRSn
13	IPC13	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC13 bit. Reset type: CPU1.SYSRSn
12	IPC12	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC12 bit. Reset type: CPU1.SYSRSn
11	IPC11	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC11 bit. Reset type: CPU1.SYSRSn
10	IPC10	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC10 bit. Reset type: CPU1.SYSRSn
9	IPC9	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC9 bit. Reset type: CPU1.SYSRSn
8	IPC8	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC8 bit. Reset type: CPU1.SYSRSn
7	IPC7	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC7 bit. Reset type: CPU1.SYSRSn
6	IPC6	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC6 bit. Reset type: CPU1.SYSRSn
5	IPC5	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC5 bit. Reset type: CPU1.SYSRSn
4	IPC4	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC4 bit. Reset type: CPU1.SYSRSn
3	IPC3	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC3 bit. Reset type: CPU1.SYSRSn
2	IPC2	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC2 bit. Reset type: CPU1.SYSRSn
1	IPC1	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC1 bit. Reset type: CPU1.SYSRSn
0	IPC0	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU1IPCFLG.IPC0 bit. Reset type: CPU1.SYSRSn

16.8.5.2 CMTOCPU1IPCSTS Register (Offset = 2h) [Reset = 0h]

CMTOCPU1IPCSTS is shown in Figure 16-40 and described in Table 16-50.

Return to the Summary Table.

Status of CPU1TOCMIPCFLG register

Figure 16-40. CMTOCPU1IPCSTS Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 16-50. CMTOCPU1IPCSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R	0h	Indicates to CPU1 if the IPC31 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC31 bit. 0: No IPC31 event was set by CM 1: An IPC31 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
30	IPC30	R	0h	Indicates to CPU1 if the IPC30 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC30 bit. 0: No IPC30 event was set by CM 1: An IPC30 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
29	IPC29	R	0h	Indicates to CPU1 if the IPC29 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC29 bit. 0: No IPC29 event was set by CM 1: An IPC29 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
28	IPC28	R	0h	Indicates to CPU1 if the IPC28 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC28 bit. 0: No IPC28 event was set by CM 1: An IPC28 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
27	IPC27	R	0h	Indicates to CPU1 if the IPC27 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC27 bit. 0: No IPC27 event was set by CM 1: An IPC27 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-50. CMTOCPU1IPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	IPC26	R	0h	Indicates to CPU1 if the IPC26 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC26 bit. 0: No IPC26 event was set by CM 1: An IPC26 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
25	IPC25	R	0h	Indicates to CPU1 if the IPC25 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC25 bit. 0: No IPC25 event was set by CM 1: An IPC25 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
24	IPC24	R	0h	Indicates to CPU1 if the IPC24 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC24 bit. 0: No IPC24 event was set by CM 1: An IPC24 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
23	IPC23	R	0h	Indicates to CPU1 if the IPC23 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC23 bit. 0: No IPC23 event was set by CM 1: An IPC23 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
22	IPC22	R	0h	Indicates to CPU1 if the IPC22 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC22 bit. 0: No IPC22 event was set by CM 1: An IPC22 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
21	IPC21	R	0h	Indicates to CPU1 if the IPC21 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC21 bit. 0: No IPC21 event was set by CM 1: An IPC21 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
20	IPC20	R	0h	Indicates to CPU1 if the IPC20 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC20 bit. 0: No IPC20 event was set by CM 1: An IPC20 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
19	IPC19	R	0h	Indicates to CPU1 if the IPC19 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC19 bit. 0: No IPC19 event was set by CM 1: An IPC19 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-50. CMTOCPU1IPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	IPC18	R	0h	Indicates to CPU1 if the IPC18 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC18 bit. 0: No IPC18 event was set by CM 1: An IPC18 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
17	IPC17	R	0h	Indicates to CPU1 if the IPC17 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC17 bit. 0: No IPC17 event was set by CM 1: An IPC17 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
16	IPC16	R	0h	Indicates to CPU1 if the IPC16 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC16 bit. 0: No IPC16 event was set by CM 1: An IPC16 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
15	IPC15	R	0h	Indicates to CPU1 if the IPC15 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC15 bit. 0: No IPC15 event was set by CM 1: An IPC15 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
14	IPC14	R	0h	Indicates to CPU1 if the IPC14 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC14 bit. 0: No IPC14 event was set by CM 1: An IPC14 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
13	IPC13	R	0h	Indicates to CPU1 if the IPC13 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC13 bit. 0: No IPC13 event was set by CM 1: An IPC13 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
12	IPC12	R	0h	Indicates to CPU1 if the IPC12 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC12 bit. 0: No IPC12 event was set by CM 1: An IPC12 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
11	IPC11	R	0h	Indicates to CPU1 if the IPC11 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC11 bit. 0: No IPC11 event was set by CM 1: An IPC11 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-50. CMTOCPU1IPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	IPC10	R	0h	Indicates to CPU1 if the IPC10 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC10 bit. 0: No IPC10 event was set by CM 1: An IPC10 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
9	IPC9	R	0h	Indicates to CPU1 if the IPC9 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC9 bit. 0: No IPC9 event was set by CM 1: An IPC9 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
8	IPC8	R	0h	Indicates to CPU1 if the IPC8 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC8 bit. 0: No IPC8 event was set by CM 1: An IPC8 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
7	IPC7	R	0h	Indicates to CPU1 if the IPC7 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC7 bit. 0: No IPC7 event was set by CM 1: An IPC7 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
6	IPC6	R	0h	Indicates to CPU1 if the IPC6 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC6 bit. 0: No IPC6 event was set by CM 1: An IPC6 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
5	IPC5	R	0h	Indicates to CPU1 if the IPC5 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC5 bit. 0: No IPC5 event was set by CM 1: An IPC5 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
4	IPC4	R	0h	Indicates to CPU1 if the IPC4 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC4 bit. 0: No IPC4 event was set by CM 1: An IPC4 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
3	IPC3	R	0h	Indicates to CPU1 if the IPC3 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC3 bit. 0: No IPC3 event was set by CM 1: An IPC3 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-50. CMTOCPU1IPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	IPC2	R	0h	Indicates to CPU1 if the IPC2 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC2 bit. 0: No IPC2 event was set by CM 1: An IPC2 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
1	IPC1	R	0h	Indicates to CPU1 if the IPC1 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC1 bit. 0: No IPC1 event was set by CM 1: An IPC1 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
0	IPC0	R	0h	Indicates to CPU1 if the IPC0 event flag was set by CM. Reflects the state of CMTOCPU1IPCFLG.IPC0 bit. 0: No IPC0 event was set by CM 1: An IPC0 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

16.8.5.3 CPU1TOCMIPCSET Register (Offset = 4h) [Reset = 0h]

CPU1TOCMIPCSET is shown in Figure 16-41 and described in Table 16-51.

Return to the Summary Table.

Set CPU1TOCMIPCFLG register

Figure 16-41. CPU1TOCMIPCSET Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-51. CPU1TOCMIPCSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC31 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
30	IPC30	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC30 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
29	IPC29	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC29 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
28	IPC28	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC28 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
27	IPC27	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC27 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-51. CPU1TOCMIPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	IPC26	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC26 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
25	IPC25	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC25 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
24	IPC24	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC24 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
23	IPC23	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC23 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
22	IPC22	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC22 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
21	IPC21	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC21 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
20	IPC20	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC20 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
19	IPC19	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC19 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
18	IPC18	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC18 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
17	IPC17	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC17 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-51. CPU1TOCMIPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	IPC16	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC16 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
15	IPC15	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC15 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
14	IPC14	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC14 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
13	IPC13	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC13 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
12	IPC12	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC12 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
11	IPC11	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC11 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
10	IPC10	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC10 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
9	IPC9	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC9 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
8	IPC8	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC8 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
7	IPC7	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC7 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-51. CPU1TOCMIPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	IPC6	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC6 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
5	IPC5	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC5 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
4	IPC4	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC4 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
3	IPC3	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC3 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
2	IPC2	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC2 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
1	IPC1	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC1 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
0	IPC0	R-0/W1S	0h	Writing 1 to this bit sets the CPU1TOCMIPCFLG.IPC0 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

16.8.5.4 CPU1TOCMIPCCLR Register (Offset = 6h) [Reset = 0h]

CPU1TOCMIPCCLR is shown in Figure 16-42 and described in Table 16-52.

Return to the Summary Table.

Clear CPU1TOCMIPCFLG register

Figure 16-42. CPU1TOCMIPCCLR Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-52. CPU1TOCMIPCCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC31 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
30	IPC30	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC30 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
29	IPC29	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC29 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
28	IPC28	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC28 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
27	IPC27	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC27 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-52. CPU1TOCMIPCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	IPC26	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC26 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
25	IPC25	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC25 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
24	IPC24	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC24 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
23	IPC23	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC23 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
22	IPC22	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC22 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
21	IPC21	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC21 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
20	IPC20	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC20 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
19	IPC19	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC19 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
18	IPC18	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC18 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
17	IPC17	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC17 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-52. CPU1TOCMIPCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	IPC16	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC16 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
15	IPC15	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC15 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
14	IPC14	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC14 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
13	IPC13	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC13 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
12	IPC12	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC12 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
11	IPC11	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC11 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
10	IPC10	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC10 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
9	IPC9	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC9 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
8	IPC8	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC8 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
7	IPC7	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC7 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-52. CPU1TOCMIPCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	IPC6	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC6 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
5	IPC5	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC5 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
4	IPC4	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC4 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
3	IPC3	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC3 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
2	IPC2	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC2 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
1	IPC1	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC1 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
0	IPC0	R-0/W1S	0h	Writing 1 to this bit cleare the CPU1TOCMIPCFLG.IPC0 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

16.8.5.5 CPU1TOCMIPCFLG Register (Offset = 8h) [Reset = 0h]

CPU1TOCMIPCFLG is shown in Figure 16-43 and described in Table 16-53.

Return to the Summary Table.

CPU1TOCMIPCFLG Register

Figure 16-43. CPU1TOCMIPCFLG Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 16-53. CPU1TOCMIPCFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R	0h	0: No IPC31 event request to CM 1: IPC31 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
30	IPC30	R	0h	0: No IPC30 event request to CM 1: IPC30 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
29	IPC29	R	0h	0: No IPC29 event request to CM 1: IPC29 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
28	IPC28	R	0h	0: No IPC28 event request to CM 1: IPC28 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
27	IPC27	R	0h	0: No IPC27 event request to CM 1: IPC27 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
26	IPC26	R	0h	0: No IPC26 event request to CM 1: IPC26 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-53. CPU1TOCMIPCFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	IPC25	R	0h	0: No IPC25 event request to CM 1: IPC25 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
24	IPC24	R	0h	0: No IPC24 event request to CM 1: IPC24 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
23	IPC23	R	0h	0: No IPC23 event request to CM 1: IPC23 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
22	IPC22	R	0h	0: No IPC22 event request to CM 1: IPC22 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
21	IPC21	R	0h	0: No IPC21 event request to CM 1: IPC21 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
20	IPC20	R	0h	0: No IPC20 event request to CM 1: IPC20 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
19	IPC19	R	0h	0: No IPC19 event request to CM 1: IPC19 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
18	IPC18	R	0h	0: No IPC18 event request to CM 1: IPC18 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
17	IPC17	R	0h	0: No IPC17 event request to CM 1: IPC17 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
16	IPC16	R	0h	0: No IPC16 event request to CM 1: IPC16 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
15	IPC15	R	0h	0: No IPC15 event request to CM 1: IPC15 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
14	IPC14	R	0h	0: No IPC14 event request to CM 1: IPC14 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-53. CPU1TOCMIPCFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	IPC13	R	0h	0: No IPC13 event request to CM 1: IPC13 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
12	IPC12	R	0h	0: No IPC12 event request to CM 1: IPC12 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
11	IPC11	R	0h	0: No IPC11 event request to CM 1: IPC11 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
10	IPC10	R	0h	0: No IPC10 event request to CM 1: IPC10 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
9	IPC9	R	0h	0: No IPC9 event request to CM 1: IPC9 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
8	IPC8	R	0h	0: No IPC8 event request to CM 1: IPC8 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
7	IPC7	R	0h	0: No IPC7 event request to CM 1: IPC7 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
6	IPC6	R	0h	0: No IPC6 event request to CM 1: IPC6 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
5	IPC5	R	0h	0: No IPC5 event request to CM 1: IPC5 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
4	IPC4	R	0h	0: No IPC4 event request to CM 1: IPC4 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
3	IPC3	R	0h	0: No IPC3 event request to CM 1: IPC3 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
2	IPC2	R	0h	0: No IPC2 event request to CM 1: IPC2 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-53. CPU1TOCMIPCFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	IPC1	R	0h	0: No IPC1 event request to CM 1: IPC1 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
0	IPC0	R	0h	0: No IPC0 event request to CM 1: IPC0 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

16.8.5.6 IPCCOUNTERL Register (Offset = Ch) [Reset = 0h]

IPCCOUNTERL is shown in Figure 16-44 and described in Table 16-54.

Return to the Summary Table.

IPC Counter Low Register

Figure 16-44. IPCCOUNTERL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT																															
R-0h																															

Table 16-54. IPCCOUNTERL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNT	R	0h	This is the lower 32-bits of free running 64 bit timestamp counter clocked by the PLLSYSCLK. Reset type: CPU1.SYSRSn

16.8.5.7 IPCCOUNTERH Register (Offset = Eh) [Reset = 0h]

IPCCOUNTERH is shown in Figure 16-45 and described in Table 16-55.

Return to the Summary Table.

IPC Counter High Register

Figure 16-45. IPCCOUNTERH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT																															
R-0h																															

Table 16-55. IPCCOUNTERH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNT	R	0h	This is the upper 32-bits of free running 64 bit timestamp counter clocked by the PLLSYSCLK. Reset type: CPU1.SYSRSn

16.8.5.8 CPU1TOCMIPCSENDCOM Register (Offset = 10h) [Reset = 0h]

CPU1TOCMIPCSENDCOM is shown in Figure 16-46 and described in Table 16-56.

Return to the Summary Table.

CPU1 to CM IPC Command

Figure 16-46. CPU1TOCMIPCSENDCOM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMMAND																															
R/W-0h																															

Table 16-56. CPU1TOCMIPCSENDCOM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COMMAND	R/W	0h	This is a general purpose register used to send software-defined commands to CM from CPU1. Reset type: CPU1.SYSRSn

16.8.5.9 CPU1TOCMIPCSENDADDR Register (Offset = 12h) [Reset = 0h]

CPU1TOCMIPCSENDADDR is shown in Figure 16-47 and described in Table 16-57.

Return to the Summary Table.

CPU1 to CM IPC Address

Figure 16-47. CPU1TOCMIPCSENDADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS																															
R/W-0h																															

Table 16-57. CPU1TOCMIPCSENDADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRESS	R/W	0h	This is a general purpose register used to send software-defined address to CM from CPU1. Reset type: CPU1.SYSRSn

16.8.5.10 CPU1TOCMIPCSENDDATA Register (Offset = 14h) [Reset = 0h]

CPU1TOCMIPCSENDDATA is shown in Figure 16-48 and described in Table 16-58.

Return to the Summary Table.

CPU1 to CM IPC Data

Figure 16-48. CPU1TOCMIPCSENDDATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDATA																															
R/W-0h																															

Table 16-58. CPU1TOCMIPCSENDDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WDATA	R/W	0h	This is a general purpose register used to send software-defined data to CM from CPU1. Reset type: CPU1.SYSRSn

16.8.5.11 CMTOCPU1IPCREPLY Register (Offset = 16h) [Reset = 0h]

CMTOCPU1IPCREPLY is shown in Figure 16-49 and described in Table 16-59.

Return to the Summary Table.

Reply from CM to CPU1TOCMIPCSENDCOM command request

Figure 16-49. CMTOCPU1IPCREPLY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDATA																															
R/W-0h																															

Table 16-59. CMTOCPU1IPCREPLY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RDATA	R/W	0h	This is a general purpose register used to send a reply to CPU1 to CM command from CM. Note: This register is not writable from CPU1. Reset type: CM.RESETn

16.8.5.12 CMTOCPU1IPCRECVCOM Register (Offset = 18h) [Reset = 0h]

CMTOCPU1IPCRECVCOM is shown in Figure 16-50 and described in Table 16-60.

Return to the Summary Table.

Refelects the value in CMTOCPU1IPCSENDCOM Register

Figure 16-50. CMTOCPU1IPCRECVCOM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMMAND																															
R-0h																															

Table 16-60. CMTOCPU1IPCRECVCOM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COMMAND	R	0h	Refelects the state of CMTOCPU1IPCSENDCOM register Reset type: CM.RESETn

16.8.5.13 CMTOCPU1IPCRECVADDR Register (Offset = 1Ah) [Reset = 0h]

CMTOCPU1IPCRECVADDR is shown in Figure 16-51 and described in Table 16-61.

Return to the Summary Table.

Refelects the value in CMTOCPU1IPCSENDADDR Register

Figure 16-51. CMTOCPU1IPCRECVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS																															
R-0h																															

Table 16-61. CMTOCPU1IPCRECVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRESS	R	0h	Refelects the state of CMTOCPU1IPCSENDADDR register Reset type: CM.RESETn

16.8.5.14 CMTOCPU1IPCRECVDATA Register (Offset = 1Ch) [Reset = 0h]

CMTOCPU1IPCRECVDATA is shown in Figure 16-52 and described in Table 16-62.

Return to the Summary Table.

Refelects the value in CMTOCPU1IPCSENDDATA Register

Figure 16-52. CMTOCPU1IPCRECVDATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDATA																															
R-0h																															

Table 16-62. CMTOCPU1IPCRECVDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WDATA	R	0h	Refelects the state of CMTOCPU1IPCSENDDATA register Reset type: CM.RESETn

16.8.5.15 CPU1TOCMIPCREPLY Register (Offset = 1Eh) [Reset = 0h]

CPU1TOCMIPCREPLY is shown in Figure 16-53 and described in Table 16-63.

Return to the Summary Table.

Reply from CPU1 to CMTOCPU1IPCSENDCOM command

Figure 16-53. CPU1TOCMIPCREPLY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDATA																															
R/W-0h																															

Table 16-63. CPU1TOCMIPCREPLY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RDATA	R/W	0h	This is a general purpose register used to send a reply to CM to CPU1 command from CPU1. Note: This register is not writable from CM. Reset type: CPU1.SYSRSn

16.8.5.16 CMTOCPU1IPCBOOTSTS Register (Offset = 20h) [Reset = 0h]

CMTOCPU1IPCBOOTSTS is shown in Figure 16-54 and described in Table 16-64.

Return to the Summary Table.

CM to CPU1 BOOT Status

Figure 16-54. CMTOCPU1IPCBOOTSTS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOOTSTS																															
R/W-0h																															

Table 16-64. CMTOCPU1IPCBOOTSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BOOTSTS	R/W	0h	This register is used by CM to pass the boot Status to CPU1. The data format is software-defined. It can only be written by CM. Reset type: CM.RESETn

16.8.5.17 CPU1TOCMIPCBOOTMODE Register (Offset = 22h) [Reset = 0h]

CPU1TOCMIPCBOOTMODE is shown in Figure 16-55 and described in Table 16-65.

Return to the Summary Table.

CPU1 to CM BOOT Mode setting

Figure 16-55. CPU1TOCMIPCBOOTMODE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOOTMODE																															
R/W-0h																															

Table 16-65. CPU1TOCMIPCBOOTMODE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BOOTMODE	R/W	0h	This register is used by CPU1 to pass a boot mode information to CM. The data format is software-defined. It can only be written by CPU1. Reset type: CPU1.SYSRSn

16.8.6 CPU1TOCM_IPC_REGS_CMVIEW Registers

Table 16-66 lists the memory-mapped registers for the CPU1TOCM_IPC_REGS_CMVIEW registers. All register offset addresses not listed in Table 16-66 should be considered as reserved locations and the register contents should not be modified.

Table 16-66. CPU1TOCM_IPC_REGS_CMVIEW Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CMTOCPU1IPCACK	CMTOCPU1IPCACK Register		Go
2h	CPU1TOCMIPCSTS	CPU1TOCMIPCSTS Register		Go
4h	CMTOCPU1IPCSET	CMTOCPU1IPCSET Register		Go
6h	CMTOCPU1IPCCLR	CMTOCPU1IPCCLR Register		Go
8h	CMTOCPU1IPCFLG	CMTOCPU1IPCFLG Register		Go
Ch	IPCCOUNTERL	IPCCOUNTERL Register		Go
Eh	IPCCOUNTERH	IPCCOUNTERH Register		Go
10h	CPU1TOCMIPCRECVCOM	CPU1TOCMIPCRECVCOM Register		Go
12h	CPU1TOCMIPCRECVADDR	CPU1TOCMIPCRECVADDR Register		Go
14h	CPU1TOCMIPCRECVDATA	CPU1TOCMIPCRECVDATA Register		Go
16h	CMTOCPU1IPCREPLY	CMTOCPU1IPCREPLY Register		Go
18h	CMTOCPU1IPCSENDCOM	CMTOCPU1IPCSENDCOM Register		Go
1Ah	CMTOCPU1IPCSENDADDR	CMTOCPU1IPCSENDADDR Register		Go
1Ch	CMTOCPU1IPCSENDDATA	CMTOCPU1IPCSENDDATA Register		Go
1Eh	CPU1TOCMIPCREPLY	CPU1TOCMIPCREPLY Register		Go
20h	CMTOCPU1IPCBOOTSTS	CMTOCPU1IPCBOOTSTS Register		Go
22h	CPU1TOCMIPCBOOTMODE	CPU1TOCMIPCBOOTMODE Register		Go
24h	PUMPREQUEST	PUMPREQUEST Register		Go

Complex bit access types are encoded to fit into small table cells. Table 16-67 shows the codes that are used for access types in this section.

Table 16-67. CPU1TOCM_IPC_REGS_CMVIEW Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 16-67. CPU1TOCM_IPC_REGS_CMVIEW Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

16.8.6.1 CMTOCPU1IPCACK Register (Offset = 0h) [Reset = 0h]

CMTOCPU1IPCACK is shown in Figure 16-56 and described in Table 16-68.

Return to the Summary Table.

CMTOCPU1IPCACK Register

Figure 16-56. CMTOCPU1IPCACK Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-68. CMTOCPU1IPCACK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC31 bit. Reset type: CM.RESETn
30	IPC30	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC30 bit. Reset type: CM.RESETn
29	IPC29	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC29 bit. Reset type: CM.RESETn
28	IPC28	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC28 bit. Reset type: CM.RESETn
27	IPC27	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC27 bit. Reset type: CM.RESETn
26	IPC26	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC26 bit. Reset type: CM.RESETn
25	IPC25	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC25 bit. Reset type: CM.RESETn
24	IPC24	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC24 bit. Reset type: CM.RESETn
23	IPC23	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC23 bit. Reset type: CM.RESETn
22	IPC22	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC22 bit. Reset type: CM.RESETn
21	IPC21	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC21 bit. Reset type: CM.RESETn
20	IPC20	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC20 bit. Reset type: CM.RESETn
19	IPC19	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC19 bit. Reset type: CM.RESETn
18	IPC18	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC18 bit. Reset type: CM.RESETn
17	IPC17	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC17 bit. Reset type: CM.RESETn

Table 16-68. CMTOCPU1IPCACK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	IPC16	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC16 bit. Reset type: CM.RESETn
15	IPC15	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC15 bit. Reset type: CM.RESETn
14	IPC14	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC14 bit. Reset type: CM.RESETn
13	IPC13	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC13 bit. Reset type: CM.RESETn
12	IPC12	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC12 bit. Reset type: CM.RESETn
11	IPC11	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC11 bit. Reset type: CM.RESETn
10	IPC10	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC10 bit. Reset type: CM.RESETn
9	IPC9	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC9 bit. Reset type: CM.RESETn
8	IPC8	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC8 bit. Reset type: CM.RESETn
7	IPC7	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC7 bit. Reset type: CM.RESETn
6	IPC6	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC6 bit. Reset type: CM.RESETn
5	IPC5	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC5 bit. Reset type: CM.RESETn
4	IPC4	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC4 bit. Reset type: CM.RESETn
3	IPC3	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC3 bit. Reset type: CM.RESETn
2	IPC2	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC2 bit. Reset type: CM.RESETn
1	IPC1	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC1 bit. Reset type: CM.RESETn
0	IPC0	R-0/W1S	0h	Writing 1 to this bit, will clear CPU1TOCMIPCFLG.IPC0 bit. Reset type: CM.RESETn

16.8.6.2 CPU1TOCMIPCSTS Register (Offset = 2h) [Reset = 0h]

CPU1TOCMIPCSTS is shown in Figure 16-57 and described in Table 16-69.

Return to the Summary Table.

Status of CMTOCPU1IPCFLG register

Figure 16-57. CPU1TOCMIPCSTS Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 16-69. CPU1TOCMIPCSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R	0h	Indicates to CM if the IPC31 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC31 bit. 0: No IPC31 event was set by CPU1 1: An IPC31 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
30	IPC30	R	0h	Indicates to CM if the IPC30 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC30 bit. 0: No IPC30 event was set by CPU1 1: An IPC30 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
29	IPC29	R	0h	Indicates to CM if the IPC29 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC29 bit. 0: No IPC29 event was set by CPU1 1: An IPC29 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
28	IPC28	R	0h	Indicates to CM if the IPC28 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC28 bit. 0: No IPC28 event was set by CPU1 1: An IPC28 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
27	IPC27	R	0h	Indicates to CM if the IPC27 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC27 bit. 0: No IPC27 event was set by CPU1 1: An IPC27 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-69. CPU1TOCMIPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	IPC26	R	0h	Indicates to CM if the IPC26 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC26 bit. 0: No IPC26 event was set by CPU1 1: An IPC26 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
25	IPC25	R	0h	Indicates to CM if the IPC25 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC25 bit. 0: No IPC25 event was set by CPU1 1: An IPC25 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
24	IPC24	R	0h	Indicates to CM if the IPC24 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC24 bit. 0: No IPC24 event was set by CPU1 1: An IPC24 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
23	IPC23	R	0h	Indicates to CM if the IPC23 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC23 bit. 0: No IPC23 event was set by CPU1 1: An IPC23 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
22	IPC22	R	0h	Indicates to CM if the IPC22 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC22 bit. 0: No IPC22 event was set by CPU1 1: An IPC22 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
21	IPC21	R	0h	Indicates to CM if the IPC21 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC21 bit. 0: No IPC21 event was set by CPU1 1: An IPC21 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
20	IPC20	R	0h	Indicates to CM if the IPC20 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC20 bit. 0: No IPC20 event was set by CPU1 1: An IPC20 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
19	IPC19	R	0h	Indicates to CM if the IPC19 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC19 bit. 0: No IPC19 event was set by CPU1 1: An IPC19 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-69. CPU1TOCMIPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	IPC18	R	0h	Indicates to CM if the IPC18 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC18 bit. 0: No IPC18 event was set by CPU1 1: An IPC18 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
17	IPC17	R	0h	Indicates to CM if the IPC17 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC17 bit. 0: No IPC17 event was set by CPU1 1: An IPC17 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
16	IPC16	R	0h	Indicates to CM if the IPC16 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC16 bit. 0: No IPC16 event was set by CPU1 1: An IPC16 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
15	IPC15	R	0h	Indicates to CM if the IPC15 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC15 bit. 0: No IPC15 event was set by CPU1 1: An IPC15 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
14	IPC14	R	0h	Indicates to CM if the IPC14 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC14 bit. 0: No IPC14 event was set by CPU1 1: An IPC14 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
13	IPC13	R	0h	Indicates to CM if the IPC13 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC13 bit. 0: No IPC13 event was set by CPU1 1: An IPC13 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
12	IPC12	R	0h	Indicates to CM if the IPC12 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC12 bit. 0: No IPC12 event was set by CPU1 1: An IPC12 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
11	IPC11	R	0h	Indicates to CM if the IPC11 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC11 bit. 0: No IPC11 event was set by CPU1 1: An IPC11 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-69. CPU1TOCMIPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	IPC10	R	0h	Indicates to CM if the IPC10 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC10 bit. 0: No IPC10 event was set by CPU1 1: An IPC10 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
9	IPC9	R	0h	Indicates to CM if the IPC9 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC9 bit. 0: No IPC9 event was set by CPU1 1: An IPC9 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
8	IPC8	R	0h	Indicates to CM if the IPC8 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC8 bit. 0: No IPC8 event was set by CPU1 1: An IPC8 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
7	IPC7	R	0h	Indicates to CM if the IPC7 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC7 bit. 0: No IPC7 event was set by CPU1 1: An IPC7 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
6	IPC6	R	0h	Indicates to CM if the IPC6 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC6 bit. 0: No IPC6 event was set by CPU1 1: An IPC6 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
5	IPC5	R	0h	Indicates to CM if the IPC5 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC5 bit. 0: No IPC5 event was set by CPU1 1: An IPC5 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
4	IPC4	R	0h	Indicates to CM if the IPC4 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC4 bit. 0: No IPC4 event was set by CPU1 1: An IPC4 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
3	IPC3	R	0h	Indicates to CM if the IPC3 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC3 bit. 0: No IPC3 event was set by CPU1 1: An IPC3 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

Table 16-69. CPU1TOCMIPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	IPC2	R	0h	Indicates to CM if the IPC2 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC2 bit. 0: No IPC2 event was set by CPU1 1: An IPC2 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
1	IPC1	R	0h	Indicates to CM if the IPC1 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC1 bit. 0: No IPC1 event was set by CPU1 1: An IPC1 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn
0	IPC0	R	0h	Indicates to CM if the IPC0 event flag was set by CPU1. Reflects the state of CPU1TOCMIPCFLG.IPC0 bit. 0: No IPC0 event was set by CPU1 1: An IPC0 event was set by CPU1 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU1.SYSRSn

16.8.6.3 CMTOCPU1IPCSET Register (Offset = 4h) [Reset = 0h]

CMTOCPU1IPCSET is shown in Figure 16-58 and described in Table 16-70.

Return to the Summary Table.

Set CMTOCPU1IPCFLG register

Figure 16-58. CMTOCPU1IPCSET Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-70. CMTOCPU1IPCSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC31 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
30	IPC30	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC30 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
29	IPC29	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC29 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
28	IPC28	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC28 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
27	IPC27	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC27 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-70. CMTOCPU1IPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	IPC26	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC26 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
25	IPC25	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC25 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
24	IPC24	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC24 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
23	IPC23	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC23 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
22	IPC22	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC22 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
21	IPC21	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC21 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
20	IPC20	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC20 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
19	IPC19	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC19 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
18	IPC18	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC18 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
17	IPC17	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC17 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-70. CMTOCPU1IPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	IPC16	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC16 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
15	IPC15	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC15 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
14	IPC14	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC14 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
13	IPC13	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC13 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
12	IPC12	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC12 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
11	IPC11	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC11 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
10	IPC10	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC10 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
9	IPC9	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC9 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
8	IPC8	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC8 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
7	IPC7	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC7 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-70. CMTOCPU1IPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	IPC6	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC6 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
5	IPC5	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC5 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
4	IPC4	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC4 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
3	IPC3	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC3 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
2	IPC2	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC2 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
1	IPC1	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC1 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
0	IPC0	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU1IPCFLG.IPC0 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

16.8.6.4 CMTOCPU1IPCCLR Register (Offset = 6h) [Reset = 0h]

CMTOCPU1IPCCLR is shown in Figure 16-59 and described in Table 16-71.

Return to the Summary Table.

Clear CMTOCPU1IPCFLG register

Figure 16-59. CMTOCPU1IPCCLR Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-71. CMTOCPU1IPCCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC31 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
30	IPC30	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC30 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
29	IPC29	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC29 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
28	IPC28	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC28 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
27	IPC27	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC27 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-71. CMTOCPU1IPCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	IPC26	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC26 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
25	IPC25	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC25 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
24	IPC24	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC24 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
23	IPC23	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC23 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
22	IPC22	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC22 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
21	IPC21	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC21 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
20	IPC20	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC20 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
19	IPC19	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC19 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
18	IPC18	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC18 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
17	IPC17	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC17 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-71. CMTOCPU1IPCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	IPC16	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC16 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
15	IPC15	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC15 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
14	IPC14	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC14 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
13	IPC13	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC13 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
12	IPC12	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC12 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
11	IPC11	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC11 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
10	IPC10	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC10 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
9	IPC9	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC9 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
8	IPC8	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC8 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
7	IPC7	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC7 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-71. CMTOCPU1IPCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	IPC6	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC6 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
5	IPC5	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC5 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
4	IPC4	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC4 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
3	IPC3	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC3 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
2	IPC2	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC2 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
1	IPC1	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC1 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
0	IPC0	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU1IPCFLG.IPC0 event flag for CPU1. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

16.8.6.5 CMTOCPU1IPCFLG Register (Offset = 8h) [Reset = 0h]

CMTOCPU1IPCFLG is shown in Figure 16-60 and described in Table 16-72.

Return to the Summary Table.

CMTOCPU1IPCFLG Register

Figure 16-60. CMTOCPU1IPCFLG Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 16-72. CMTOCPU1IPCFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R	0h	0: No IPC31 event request to CPU1 1: IPC31 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
30	IPC30	R	0h	0: No IPC30 event request to CPU1 1: IPC30 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
29	IPC29	R	0h	0: No IPC29 event request to CPU1 1: IPC29 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
28	IPC28	R	0h	0: No IPC28 event request to CPU1 1: IPC28 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
27	IPC27	R	0h	0: No IPC27 event request to CPU1 1: IPC27 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
26	IPC26	R	0h	0: No IPC26 event request to CPU1 1: IPC26 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-72. CMTOCPU1IPCFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	IPC25	R	0h	0: No IPC25 event request to CPU1 1: IPC25 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
24	IPC24	R	0h	0: No IPC24 event request to CPU1 1: IPC24 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
23	IPC23	R	0h	0: No IPC23 event request to CPU1 1: IPC23 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
22	IPC22	R	0h	0: No IPC22 event request to CPU1 1: IPC22 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
21	IPC21	R	0h	0: No IPC21 event request to CPU1 1: IPC21 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
20	IPC20	R	0h	0: No IPC20 event request to CPU1 1: IPC20 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
19	IPC19	R	0h	0: No IPC19 event request to CPU1 1: IPC19 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
18	IPC18	R	0h	0: No IPC18 event request to CPU1 1: IPC18 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
17	IPC17	R	0h	0: No IPC17 event request to CPU1 1: IPC17 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
16	IPC16	R	0h	0: No IPC16 event request to CPU1 1: IPC16 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
15	IPC15	R	0h	0: No IPC15 event request to CPU1 1: IPC15 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
14	IPC14	R	0h	0: No IPC14 event request to CPU1 1: IPC14 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-72. CMTOCPU1IPCFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	IPC13	R	0h	0: No IPC13 event request to CPU1 1: IPC13 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
12	IPC12	R	0h	0: No IPC12 event request to CPU1 1: IPC12 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
11	IPC11	R	0h	0: No IPC11 event request to CPU1 1: IPC11 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
10	IPC10	R	0h	0: No IPC10 event request to CPU1 1: IPC10 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
9	IPC9	R	0h	0: No IPC9 event request to CPU1 1: IPC9 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
8	IPC8	R	0h	0: No IPC8 event request to CPU1 1: IPC8 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
7	IPC7	R	0h	0: No IPC7 event request to CPU1 1: IPC7 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
6	IPC6	R	0h	0: No IPC6 event request to CPU1 1: IPC6 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
5	IPC5	R	0h	0: No IPC5 event request to CPU1 1: IPC5 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
4	IPC4	R	0h	0: No IPC4 event request to CPU1 1: IPC4 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
3	IPC3	R	0h	0: No IPC3 event request to CPU1 1: IPC3 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
2	IPC2	R	0h	0: No IPC2 event request to CPU1 1: IPC2 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-72. CMTOCPU1IPCFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	IPC1	R	0h	0: No IPC1 event request to CPU1 1: IPC1 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
0	IPC0	R	0h	0: No IPC0 event request to CPU1 1: IPC0 event request to CPU1 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

16.8.6.6 IPCCOUNTERL Register (Offset = Ch) [Reset = 0h]

IPCCOUNTERL is shown in Figure 16-61 and described in Table 16-73.

Return to the Summary Table.

IPC Counter Low Register

Figure 16-61. IPCCOUNTERL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT																															
R-0h																															

Table 16-73. IPCCOUNTERL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNT	R	0h	This is the lower 32-bits of free running 64 bit timestamp counter clocked by the PLLSYSCLK. Reset type: CPU1.SYSRSn

16.8.6.7 IPCCOUNTERH Register (Offset = Eh) [Reset = 0h]

IPCCOUNTERH is shown in Figure 16-62 and described in Table 16-74.

Return to the Summary Table.

IPC Counter High Register

Figure 16-62. IPCCOUNTERH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT																															
R-0h																															

Table 16-74. IPCCOUNTERH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNT	R	0h	This is the upper 32-bits of free running 64 bit timestamp counter clocked by the PLLSYSCLK. Reset type: CPU1.SYSRSn

16.8.6.8 CPU1TOCMIPCRECVCOM Register (Offset = 10h) [Reset = 0h]

CPU1TOCMIPCRECVCOM is shown in Figure 16-63 and described in Table 16-75.

Return to the Summary Table.

Refelects the value in CPU1TOCMIPCSENDCOM Register

Figure 16-63. CPU1TOCMIPCRECVCOM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMMAND																															
R-0h																															

Table 16-75. CPU1TOCMIPCRECVCOM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COMMAND	R	0h	Refelects the state of CPU1TOCMIPCSENDCOM register Reset type: CPU1.SYSRSn

16.8.6.9 CPU1TOCMIPCRECVADDR Register (Offset = 12h) [Reset = 0h]

CPU1TOCMIPCRECVADDR is shown in Figure 16-64 and described in Table 16-76.

Return to the Summary Table.

Refelects the value in CPU1TOCMIPCSENADDR Register

Figure 16-64. CPU1TOCMIPCRECVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS																															
R-0h																															

Table 16-76. CPU1TOCMIPCRECVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRESS	R	0h	Refelects the state of CPU1TOCMIPCSENDADDR register Reset type: CPU1.SYSRSn

16.8.6.10 CPU1TOCMIPCRECVDATA Register (Offset = 14h) [Reset = 0h]

CPU1TOCMIPCRECVDATA is shown in Figure 16-65 and described in Table 16-77.

Return to the Summary Table.

Refelects the value in CPU1TOCMIPCSENDDATA Register

Figure 16-65. CPU1TOCMIPCRECVDATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDATA																															
R-0h																															

Table 16-77. CPU1TOCMIPCRECVDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WDATA	R	0h	Refelects the state of CPU1TOCMIPCSENDDATA register Reset type: CPU1.SYSRSn

16.8.6.11 CMTOCPU1IPCREPLY Register (Offset = 16h) [Reset = 0h]

CMTOCPU1IPCREPLY is shown in Figure 16-66 and described in Table 16-78.

Return to the Summary Table.

Reply from CM to CPU1TOCMIPCSENDCOM command

Figure 16-66. CMTOCPU1IPCREPLY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDATA																															
R/W-0h																															

Table 16-78. CMTOCPU1IPCREPLY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RDATA	R/W	0h	This is a general purpose register used to send a reply to CPU1 to CM command from CM. Note: This register is not writable from CPU1. Reset type: CM.RESETn

16.8.6.12 CMTOCPU1IPCSENDCOM Register (Offset = 18h) [Reset = 0h]

CMTOCPU1IPCSENDCOM is shown in Figure 16-67 and described in Table 16-79.

Return to the Summary Table.

CM to CPU1 IPC Command

Figure 16-67. CMTOCPU1IPCSENDCOM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMMAND																															
R/W-0h																															

Table 16-79. CMTOCPU1IPCSENDCOM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COMMAND	R/W	0h	This is a general purpose register used to send software-defined commands to CPU1 from CM. Reset type: CM.RESETn

16.8.6.13 CMTOCPU1IPCSENDADDR Register (Offset = 1Ah) [Reset = 0h]

CMTOCPU1IPCSENDADDR is shown in Figure 16-68 and described in Table 16-80.

Return to the Summary Table.

CM to CPU1 IPC Address

Figure 16-68. CMTOCPU1IPCSENDADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS																															
R/W-0h																															

Table 16-80. CMTOCPU1IPCSENDADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRESS	R/W	0h	This is a general purpose register used to send software-defined address to CPU1 from CM. Reset type: CM.RESETn

16.8.6.14 CMTOCPU1IPCSENDDATA Register (Offset = 1Ch) [Reset = 0h]

CMTOCPU1IPCSENDDATA is shown in Figure 16-69 and described in Table 16-81.

Return to the Summary Table.

CM to CPU1 IPC Data

Figure 16-69. CMTOCPU1IPCSENDDATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDATA																															
R/W-0h																															

Table 16-81. CMTOCPU1IPCSENDDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WDATA	R/W	0h	This is a general purpose register used to send software-defined data to CPU1 from CM. Reset type: CM.RESETn

16.8.6.15 CPU1TOCMIPCREPLY Register (Offset = 1Eh) [Reset = 0h]

CPU1TOCMIPCREPLY is shown in Figure 16-70 and described in Table 16-82.

Return to the Summary Table.

Reply from CPU1 to CMTOCPU1IPCSENDCOM command request

Figure 16-70. CPU1TOCMIPCREPLY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDATA																															
R/W-0h																															

Table 16-82. CPU1TOCMIPCREPLY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RDATA	R/W	0h	This is a general purpose register used to send a reply to CM to CPU1 command from CPU1. Note: This register is not writable from CM. Reset type: CPU1.SYSRSn

16.8.6.16 CMTOCPU1IPCBOOTSTS Register (Offset = 20h) [Reset = 0h]

CMTOCPU1IPCBOOTSTS is shown in Figure 16-71 and described in Table 16-83.

Return to the Summary Table.

CM to CPU1 BOOT Status

Figure 16-71. CMTOCPU1IPCBOOTSTS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOOTSTS																															
R/W-0h																															

Table 16-83. CMTOCPU1IPCBOOTSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BOOTSTS	R/W	0h	This register is used by CM to pass the boot Status to CPU1. The data format is software-defined. It can only be written by CM. Reset type: CM.RESETn

16.8.6.17 CPU1TOCMIPCBOOTMODE Register (Offset = 22h) [Reset = 0h]

CPU1TOCMIPCBOOTMODE is shown in Figure 16-72 and described in Table 16-84.

Return to the Summary Table.

CPU1 to CM BOOT Mode setting

Figure 16-72. CPU1TOCMIPCBOOTMODE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOOTMODE																															
R/W-0h																															

Table 16-84. CPU1TOCMIPCBOOTMODE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BOOTMODE	R/W	0h	This register is used by CPU1 to pass a boot mode information to CM. The data format is software-defined. It can only be written by CPU1. Reset type: CPU1.SYSRSn

16.8.6.18 PUMPREQUEST Register (Offset = 24h) [Reset = 0h]

PUMPREQUEST is shown in Figure 16-73 and described in Table 16-85.

Return to the Summary Table.

Flash programming semaphore PUMP request register

Figure 16-73. PUMPREQUEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
R-0/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														SEM	
R-0h														R/W-0h	

Table 16-85. PUMPREQUEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	In order to write to the semaphore bits, 0x5a5a must be written to these key bits at the same time. Otherwise, writes are ignored. The key is cleared immediately after writing, so it must be written again for every semaphore change. Reset type: CPU1.SYSRSn
15-2	RESERVED	R	0h	Reserved
1-0	SEM	R/W	0h	These bits decide which CPU has control of the flash pump, which allows write access to the flash memory. The possible values are: 00: Read-only state. CPU1 has control of the pump, but CPU2 and CM may seize control at any time. 01: CPU2 has exclusive control of the pump and of these semaphore bits. CPU2 can relinquish control by setting the bits back to 00. 10: CPU1 has exclusive control of the pump and of these semaphore bits. CPU1 can relinquish control by setting the bits back to 00. 11: CM has exclusive control of the pump and of these semaphore bits. CM can relinquish control by setting the bits back to 00. Going from 01->10/11 or 10->01/11 or 11->01/10 is not allowed. The semaphore bits [1:0] must be written along with the correct key in bits [31:16]. Note: This field will be reset by the respective CPU resets depending on who owns the PUMP. For example if CPU2 is owning the pump, then CPU2SYSRSN would reset this field. Reset type: CPU1.SYSRSn, CPU2.SYSRSn, CM.RESETn

16.8.7 CPU2TOCM_IPC_REGS_CPU2VIEW Registers

Table 16-86 lists the memory-mapped registers for the CPU2TOCM_IPC_REGS_CPU2VIEW registers. All register offset addresses not listed in Table 16-86 should be considered as reserved locations and the register contents should not be modified.

Table 16-86. CPU2TOCM_IPC_REGS_CPU2VIEW Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CPU2TOCMIPCACK	CPU2TOCMIPCACK Register		Go
2h	CMTOCPU2IPCSTS	CMTOCPU2IPCSTS Register		Go
4h	CPU2TOCMIPCSET	CPU2TOCMIPCSET Register		Go
6h	CPU2TOCMIPCCLR	CPU2TOCMIPCCLR Register		Go
8h	CPU2TOCMIPCFLG	CPU2TOCMIPCFLG Register		Go
Ch	IPCCOUNTERL	IPCCOUNTERL Register		Go
Eh	IPCCOUNTERH	IPCCOUNTERH Register		Go
10h	CPU2TOCMIPCSENDCOM	CPU2TOCMIPCSENDCOM Register		Go
12h	CPU2TOCMIPCSENDADDR	CPU2TOCMIPCSENDADDR Register		Go
14h	CPU2TOCMIPCSENDDATA	CPU2TOCMIPCSENDDATA Register		Go
16h	CMTOCPU2IPCREPLY	CMTOCPU2IPCREPLY Register		Go
18h	CMTOCPU2IPCRECVCOM	CMTOCPU2IPCRECVCOM Register		Go
1Ah	CMTOCPU2IPCRECVADDR	CMTOCPU2IPCRECVADDR Register		Go
1Ch	CMTOCPU2IPCRECVDATA	CMTOCPU2IPCRECVDATA Register		Go
1Eh	CPU2TOCMIPCREPLY	CPU2TOCMIPCREPLY Register		Go

Complex bit access types are encoded to fit into small table cells. Table 16-87 shows the codes that are used for access types in this section.

Table 16-87. CPU2TOCM_IPC_REGS_CPU2VIEW Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 16-87. CPU2TOCM_IPC_REGS_CPU2VIEW Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

16.8.7.1 CPU2TOCMIPCACK Register (Offset = 0h) [Reset = 0h]

CPU2TOCMIPCACK is shown in Figure 16-74 and described in Table 16-88.

Return to the Summary Table.

CPU2TOCMIPCACK Register

Figure 16-74. CPU2TOCMIPCACK Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-88. CPU2TOCMIPCACK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC31 bit. Reset type: CPU2.SYSRSn
30	IPC30	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC30 bit. Reset type: CPU2.SYSRSn
29	IPC29	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC29 bit. Reset type: CPU2.SYSRSn
28	IPC28	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC28 bit. Reset type: CPU2.SYSRSn
27	IPC27	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC27 bit. Reset type: CPU2.SYSRSn
26	IPC26	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC26 bit. Reset type: CPU2.SYSRSn
25	IPC25	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC25 bit. Reset type: CPU2.SYSRSn
24	IPC24	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC24 bit. Reset type: CPU2.SYSRSn
23	IPC23	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC23 bit. Reset type: CPU2.SYSRSn
22	IPC22	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC22 bit. Reset type: CPU2.SYSRSn
21	IPC21	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC21 bit. Reset type: CPU2.SYSRSn
20	IPC20	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC20 bit. Reset type: CPU2.SYSRSn
19	IPC19	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC19 bit. Reset type: CPU2.SYSRSn
18	IPC18	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC18 bit. Reset type: CPU2.SYSRSn
17	IPC17	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC17 bit. Reset type: CPU2.SYSRSn

Table 16-88. CPU2TOCMIPCACK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	IPC16	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC16 bit. Reset type: CPU2.SYSRSn
15	IPC15	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC15 bit. Reset type: CPU2.SYSRSn
14	IPC14	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC14 bit. Reset type: CPU2.SYSRSn
13	IPC13	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC13 bit. Reset type: CPU2.SYSRSn
12	IPC12	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC12 bit. Reset type: CPU2.SYSRSn
11	IPC11	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC11 bit. Reset type: CPU2.SYSRSn
10	IPC10	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC10 bit. Reset type: CPU2.SYSRSn
9	IPC9	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC9 bit. Reset type: CPU2.SYSRSn
8	IPC8	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC8 bit. Reset type: CPU2.SYSRSn
7	IPC7	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC7 bit. Reset type: CPU2.SYSRSn
6	IPC6	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC6 bit. Reset type: CPU2.SYSRSn
5	IPC5	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC5 bit. Reset type: CPU2.SYSRSn
4	IPC4	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC4 bit. Reset type: CPU2.SYSRSn
3	IPC3	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC3 bit. Reset type: CPU2.SYSRSn
2	IPC2	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC2 bit. Reset type: CPU2.SYSRSn
1	IPC1	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC1 bit. Reset type: CPU2.SYSRSn
0	IPC0	R-0/W1S	0h	Writing 1 to this bit, will clear CMTOCPU2IPCFLG.IPC0 bit. Reset type: CPU2.SYSRSn

16.8.7.2 CMTOCPU2IPCSTS Register (Offset = 2h) [Reset = 0h]

CMTOCPU2IPCSTS is shown in Figure 16-75 and described in Table 16-89.

Return to the Summary Table.

Status of CPU2TOCMIPCFLG register

Figure 16-75. CMTOCPU2IPCSTS Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 16-89. CMTOCPU2IPCSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R	0h	Indicates to CPU2 if the IPC31 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC31 bit. 0: No IPC31 event was set by CM 1: An IPC31 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
30	IPC30	R	0h	Indicates to CPU2 if the IPC30 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC30 bit. 0: No IPC30 event was set by CM 1: An IPC30 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
29	IPC29	R	0h	Indicates to CPU2 if the IPC29 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC29 bit. 0: No IPC29 event was set by CM 1: An IPC29 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
28	IPC28	R	0h	Indicates to CPU2 if the IPC28 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC28 bit. 0: No IPC28 event was set by CM 1: An IPC28 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
27	IPC27	R	0h	Indicates to CPU2 if the IPC27 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC27 bit. 0: No IPC27 event was set by CM 1: An IPC27 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-89. CMTOCPU2IPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	IPC26	R	0h	Indicates to CPU2 if the IPC26 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC26 bit. 0: No IPC26 event was set by CM 1: An IPC26 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
25	IPC25	R	0h	Indicates to CPU2 if the IPC25 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC25 bit. 0: No IPC25 event was set by CM 1: An IPC25 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
24	IPC24	R	0h	Indicates to CPU2 if the IPC24 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC24 bit. 0: No IPC24 event was set by CM 1: An IPC24 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
23	IPC23	R	0h	Indicates to CPU2 if the IPC23 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC23 bit. 0: No IPC23 event was set by CM 1: An IPC23 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
22	IPC22	R	0h	Indicates to CPU2 if the IPC22 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC22 bit. 0: No IPC22 event was set by CM 1: An IPC22 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
21	IPC21	R	0h	Indicates to CPU2 if the IPC21 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC21 bit. 0: No IPC21 event was set by CM 1: An IPC21 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
20	IPC20	R	0h	Indicates to CPU2 if the IPC20 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC20 bit. 0: No IPC20 event was set by CM 1: An IPC20 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
19	IPC19	R	0h	Indicates to CPU2 if the IPC19 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC19 bit. 0: No IPC19 event was set by CM 1: An IPC19 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-89. CMTOCPU2IPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	IPC18	R	0h	Indicates to CPU2 if the IPC18 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC18 bit. 0: No IPC18 event was set by CM 1: An IPC18 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
17	IPC17	R	0h	Indicates to CPU2 if the IPC17 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC17 bit. 0: No IPC17 event was set by CM 1: An IPC17 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
16	IPC16	R	0h	Indicates to CPU2 if the IPC16 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC16 bit. 0: No IPC16 event was set by CM 1: An IPC16 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
15	IPC15	R	0h	Indicates to CPU2 if the IPC15 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC15 bit. 0: No IPC15 event was set by CM 1: An IPC15 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
14	IPC14	R	0h	Indicates to CPU2 if the IPC14 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC14 bit. 0: No IPC14 event was set by CM 1: An IPC14 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
13	IPC13	R	0h	Indicates to CPU2 if the IPC13 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC13 bit. 0: No IPC13 event was set by CM 1: An IPC13 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
12	IPC12	R	0h	Indicates to CPU2 if the IPC12 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC12 bit. 0: No IPC12 event was set by CM 1: An IPC12 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
11	IPC11	R	0h	Indicates to CPU2 if the IPC11 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC11 bit. 0: No IPC11 event was set by CM 1: An IPC11 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-89. CMTOCPU2IPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	IPC10	R	0h	Indicates to CPU2 if the IPC10 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC10 bit. 0: No IPC10 event was set by CM 1: An IPC10 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
9	IPC9	R	0h	Indicates to CPU2 if the IPC9 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC9 bit. 0: No IPC9 event was set by CM 1: An IPC9 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
8	IPC8	R	0h	Indicates to CPU2 if the IPC8 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC8 bit. 0: No IPC8 event was set by CM 1: An IPC8 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
7	IPC7	R	0h	Indicates to CPU2 if the IPC7 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC7 bit. 0: No IPC7 event was set by CM 1: An IPC7 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
6	IPC6	R	0h	Indicates to CPU2 if the IPC6 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC6 bit. 0: No IPC6 event was set by CM 1: An IPC6 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
5	IPC5	R	0h	Indicates to CPU2 if the IPC5 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC5 bit. 0: No IPC5 event was set by CM 1: An IPC5 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
4	IPC4	R	0h	Indicates to CPU2 if the IPC4 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC4 bit. 0: No IPC4 event was set by CM 1: An IPC4 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
3	IPC3	R	0h	Indicates to CPU2 if the IPC3 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC3 bit. 0: No IPC3 event was set by CM 1: An IPC3 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-89. CMTOCPU2IPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	IPC2	R	0h	Indicates to CPU2 if the IPC2 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC2 bit. 0: No IPC2 event was set by CM 1: An IPC2 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
1	IPC1	R	0h	Indicates to CPU2 if the IPC1 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC1 bit. 0: No IPC1 event was set by CM 1: An IPC1 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
0	IPC0	R	0h	Indicates to CPU2 if the IPC0 event flag was set by CM. Reflects the state of CMTOCPU2IPCFLG.IPC0 bit. 0: No IPC0 event was set by CM 1: An IPC0 event was set by CM Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

16.8.7.3 CPU2TOCMIPCSET Register (Offset = 4h) [Reset = 0h]

CPU2TOCMIPCSET is shown in Figure 16-76 and described in Table 16-90.

Return to the Summary Table.

Set CPU2TOCMIPCFLG register

Figure 16-76. CPU2TOCMIPCSET Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-90. CPU2TOCMIPCSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC31 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
30	IPC30	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC30 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
29	IPC29	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC29 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
28	IPC28	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC28 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
27	IPC27	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC27 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-90. CPU2TOCMIPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	IPC26	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC26 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
25	IPC25	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC25 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
24	IPC24	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC24 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
23	IPC23	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC23 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
22	IPC22	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC22 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
21	IPC21	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC21 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
20	IPC20	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC20 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
19	IPC19	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC19 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
18	IPC18	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC18 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
17	IPC17	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC17 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-90. CPU2TOCMIPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	IPC16	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC16 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
15	IPC15	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC15 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
14	IPC14	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC14 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
13	IPC13	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC13 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
12	IPC12	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC12 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
11	IPC11	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC11 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
10	IPC10	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC10 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
9	IPC9	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC9 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
8	IPC8	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC8 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
7	IPC7	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC7 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-90. CPU2TOCMIPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	IPC6	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC6 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
5	IPC5	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC5 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
4	IPC4	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC4 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
3	IPC3	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC3 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
2	IPC2	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC2 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
1	IPC1	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC1 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
0	IPC0	R-0/W1S	0h	Writing 1 to this bit sets the CPU2TOCMIPCFLG.IPC0 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

16.8.7.4 CPU2TOCMIPCCLR Register (Offset = 6h) [Reset = 0h]

CPU2TOCMIPCCLR is shown in Figure 16-77 and described in Table 16-91.

Return to the Summary Table.

Clear CPU2TOCMIPCFLG register

Figure 16-77. CPU2TOCMIPCCLR Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-91. CPU2TOCMIPCCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC31 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
30	IPC30	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC30 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
29	IPC29	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC29 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
28	IPC28	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC28 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
27	IPC27	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC27 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-91. CPU2TOCMIPCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	IPC26	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC26 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
25	IPC25	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC25 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
24	IPC24	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC24 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
23	IPC23	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC23 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
22	IPC22	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC22 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
21	IPC21	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC21 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
20	IPC20	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC20 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
19	IPC19	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC19 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
18	IPC18	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC18 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
17	IPC17	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC17 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-91. CPU2TOCMIPCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	IPC16	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC16 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
15	IPC15	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC15 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
14	IPC14	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC14 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
13	IPC13	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC13 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
12	IPC12	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC12 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
11	IPC11	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC11 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
10	IPC10	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC10 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
9	IPC9	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC9 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
8	IPC8	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC8 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
7	IPC7	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC7 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-91. CPU2TOCMIPCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	IPC6	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC6 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
5	IPC5	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC5 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
4	IPC4	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC4 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
3	IPC3	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC3 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
2	IPC2	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC2 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
1	IPC1	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC1 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
0	IPC0	R-0/W1S	0h	Writing 1 to this bit cleare the CPU2TOCMIPCFLG.IPC0 event flag for CM. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

16.8.7.5 CPU2TOCMIPCFLG Register (Offset = 8h) [Reset = 0h]

CPU2TOCMIPCFLG is shown in Figure 16-78 and described in Table 16-92.

Return to the Summary Table.

CPU2TOCMIPCFLG Register

Figure 16-78. CPU2TOCMIPCFLG Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 16-92. CPU2TOCMIPCFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R	0h	0: No IPC31 event request to CM 1: IPC31 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
30	IPC30	R	0h	0: No IPC30 event request to CM 1: IPC30 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
29	IPC29	R	0h	0: No IPC29 event request to CM 1: IPC29 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
28	IPC28	R	0h	0: No IPC28 event request to CM 1: IPC28 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
27	IPC27	R	0h	0: No IPC27 event request to CM 1: IPC27 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
26	IPC26	R	0h	0: No IPC26 event request to CM 1: IPC26 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-92. CPU2TOCMIPCFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	IPC25	R	0h	0: No IPC25 event request to CM 1: IPC25 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
24	IPC24	R	0h	0: No IPC24 event request to CM 1: IPC24 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
23	IPC23	R	0h	0: No IPC23 event request to CM 1: IPC23 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
22	IPC22	R	0h	0: No IPC22 event request to CM 1: IPC22 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
21	IPC21	R	0h	0: No IPC21 event request to CM 1: IPC21 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
20	IPC20	R	0h	0: No IPC20 event request to CM 1: IPC20 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
19	IPC19	R	0h	0: No IPC19 event request to CM 1: IPC19 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
18	IPC18	R	0h	0: No IPC18 event request to CM 1: IPC18 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
17	IPC17	R	0h	0: No IPC17 event request to CM 1: IPC17 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
16	IPC16	R	0h	0: No IPC16 event request to CM 1: IPC16 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
15	IPC15	R	0h	0: No IPC15 event request to CM 1: IPC15 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
14	IPC14	R	0h	0: No IPC14 event request to CM 1: IPC14 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-92. CPU2TOCMIPCFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	IPC13	R	0h	0: No IPC13 event request to CM 1: IPC13 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
12	IPC12	R	0h	0: No IPC12 event request to CM 1: IPC12 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
11	IPC11	R	0h	0: No IPC11 event request to CM 1: IPC11 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
10	IPC10	R	0h	0: No IPC10 event request to CM 1: IPC10 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
9	IPC9	R	0h	0: No IPC9 event request to CM 1: IPC9 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
8	IPC8	R	0h	0: No IPC8 event request to CM 1: IPC8 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
7	IPC7	R	0h	0: No IPC7 event request to CM 1: IPC7 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
6	IPC6	R	0h	0: No IPC6 event request to CM 1: IPC6 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
5	IPC5	R	0h	0: No IPC5 event request to CM 1: IPC5 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
4	IPC4	R	0h	0: No IPC4 event request to CM 1: IPC4 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
3	IPC3	R	0h	0: No IPC3 event request to CM 1: IPC3 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
2	IPC2	R	0h	0: No IPC2 event request to CM 1: IPC2 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-92. CPU2TOCMIPCFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	IPC1	R	0h	0: No IPC1 event request to CM 1: IPC1 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
0	IPC0	R	0h	0: No IPC0 event request to CM 1: IPC0 event request to CM Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

16.8.7.6 IPCCOUNTERL Register (Offset = Ch) [Reset = 0h]

IPCCOUNTERL is shown in Figure 16-79 and described in Table 16-93.

Return to the Summary Table.

IPC Counter Low Register

Figure 16-79. IPCCOUNTERL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT																															
R-0h																															

Table 16-93. IPCCOUNTERL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNT	R	0h	This is the lower 32-bits of free running 64 bit timestamp counter clocked by the PLLSYSCLK. Reset type: CPU1.SYSRSn

16.8.7.7 IPCCOUNTERH Register (Offset = Eh) [Reset = 0h]

IPCCOUNTERH is shown in Figure 16-80 and described in Table 16-94.

Return to the Summary Table.

IPC Counter High Register

Figure 16-80. IPCCOUNTERH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT																															
R-0h																															

Table 16-94. IPCCOUNTERH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNT	R	0h	This is the upper 32-bits of free running 64 bit timestamp counter clocked by the PLLSYSCLK. Reset type: CPU1.SYSRSn

16.8.7.8 CPU2TOCMIPCSENDCOM Register (Offset = 10h) [Reset = 0h]

CPU2TOCMIPCSENDCOM is shown in Figure 16-81 and described in Table 16-95.

Return to the Summary Table.

CPU2 to CM IPC Command

Figure 16-81. CPU2TOCMIPCSENDCOM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMMAND																															
R/W-0h																															

Table 16-95. CPU2TOCMIPCSENDCOM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COMMAND	R/W	0h	This is a general purpose register used to send software-defined commands to CM from CPU2. Reset type: CPU2.SYSRSn

16.8.7.9 CPU2TOCMIPCSENDADDR Register (Offset = 12h) [Reset = 0h]

CPU2TOCMIPCSENDADDR is shown in Figure 16-82 and described in Table 16-96.

Return to the Summary Table.

CPU2 to CM IPC Address

Figure 16-82. CPU2TOCMIPCSENDADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS																															
R/W-0h																															

Table 16-96. CPU2TOCMIPCSENDADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRESS	R/W	0h	This is a general purpose register used to send software-defined address to CM from CPU2. Reset type: CPU2.SYSRSn

16.8.7.10 CPU2TOCMIPCSENDDATA Register (Offset = 14h) [Reset = 0h]

CPU2TOCMIPCSENDDATA is shown in Figure 16-83 and described in Table 16-97.

Return to the Summary Table.

CPU2 to CM IPC Data

Figure 16-83. CPU2TOCMIPCSENDDATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDATA																															
R/W-0h																															

Table 16-97. CPU2TOCMIPCSENDDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WDATA	R/W	0h	This is a general purpose register used to send software-defined data to CM from CPU2. Reset type: CPU2.SYSRSn

16.8.7.11 CMTOCPU2IPCREPLY Register (Offset = 16h) [Reset = 0h]

CMTOCPU2IPCREPLY is shown in Figure 16-84 and described in Table 16-98.

Return to the Summary Table.

Reply from CM to CPU2TOCMIPCSENDCOM command request

Figure 16-84. CMTOCPU2IPCREPLY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDATA																															
R/W-0h																															

Table 16-98. CMTOCPU2IPCREPLY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RDATA	R/W	0h	This is a general purpose register used to send a reply to CPU2 to CM command from CM. Note: This register is not writable from CPU2. Reset type: CM.RESETn

16.8.7.12 CMTOCPU2IPCRECVCOM Register (Offset = 18h) [Reset = 0h]

CMTOCPU2IPCRECVCOM is shown in Figure 16-85 and described in Table 16-99.

Return to the Summary Table.

Refelects the value in CMTOCPU2IPCSENDCOM Register

Figure 16-85. CMTOCPU2IPCRECVCOM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMMAND																															
R-0h																															

Table 16-99. CMTOCPU2IPCRECVCOM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COMMAND	R	0h	Refelects the state of CMTOCPU2IPCSENDCOM register Reset type: CM.RESETn

16.8.7.13 CMTOCPU2IPCRECVADDR Register (Offset = 1Ah) [Reset = 0h]

CMTOCPU2IPCRECVADDR is shown in Figure 16-86 and described in Table 16-100.

Return to the Summary Table.

Refelects the value in CMTOCPU2IPCSENDADDR Register

Figure 16-86. CMTOCPU2IPCRECVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS																															
R-0h																															

Table 16-100. CMTOCPU2IPCRECVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRESS	R	0h	Refelects the state of CMTOCPU2IPCSENDADDR register Reset type: CM.RESETn

16.8.7.14 CMTOCPU2IPCRECVDATA Register (Offset = 1Ch) [Reset = 0h]

CMTOCPU2IPCRECVDATA is shown in Figure 16-87 and described in Table 16-101.

Return to the Summary Table.

Refelects the value in CMTOCPU2IPCSENDDATA Register

Figure 16-87. CMTOCPU2IPCRECVDATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDATA																															
R-0h																															

Table 16-101. CMTOCPU2IPCRECVDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WDATA	R	0h	Refelects the state of CMTOCPU2IPCSENDDATA register Reset type: CM.RESETn

16.8.7.15 CPU2TOCMIPCREPLY Register (Offset = 1Eh) [Reset = 0h]

CPU2TOCMIPCREPLY is shown in Figure 16-88 and described in Table 16-102.

Return to the Summary Table.

Reply from CPU2 to CMTOCPU2IPCSENDCOM command

Figure 16-88. CPU2TOCMIPCREPLY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDATA																															
R/W-0h																															

Table 16-102. CPU2TOCMIPCREPLY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RDATA	R/W	0h	This is a general purpose register used to send a reply to CM to CPU2 command from CPU2. Note: This register is not writable from CM. Reset type: CPU2.SYSRSn

16.8.8 CPU2TOCM_IPC_REGS_CMVIEW Registers

Table 16-103 lists the memory-mapped registers for the CPU2TOCM_IPC_REGS_CMVIEW registers. All register offset addresses not listed in Table 16-103 should be considered as reserved locations and the register contents should not be modified.

Table 16-103. CPU2TOCM_IPC_REGS_CMVIEW Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CMTOCPU2IPCACK	CMTOCPU2IPCACK Register		Go
2h	CPU2TOCMIPCSTS	CPU2TOCMIPCSTS Register		Go
4h	CMTOCPU2IPCSET	CMTOCPU2IPCSET Register		Go
6h	CMTOCPU2IPCCLR	CMTOCPU2IPCCLR Register		Go
8h	CMTOCPU2IPCFLG	CMTOCPU2IPCFLG Register		Go
Ch	IPCCOUNTERL	IPCCOUNTERL Register		Go
Eh	IPCCOUNTERH	IPCCOUNTERH Register		Go
10h	CPU2TOCMIPCRECVCOM	CPU2TOCMIPCRECVCOM Register		Go
12h	CPU2TOCMIPCRECVADDR	CPU2TOCMIPCRECVADDR Register		Go
14h	CPU2TOCMIPCRECVDATA	CPU2TOCMIPCRECVDATA Register		Go
16h	CMTOCPU2IPCREPLY	CMTOCPU2IPCREPLY Register		Go
18h	CMTOCPU2IPCSENDCOM	CMTOCPU2IPCSENDCOM Register		Go
1Ah	CMTOCPU2IPCSENDADDR	CMTOCPU2IPCSENDADDR Register		Go
1Ch	CMTOCPU2IPCSENDDATA	CMTOCPU2IPCSENDDATA Register		Go
1Eh	CPU2TOCMIPCREPLY	CPU2TOCMIPCREPLY Register		Go

Complex bit access types are encoded to fit into small table cells. Table 16-104 shows the codes that are used for access types in this section.

Table 16-104. CPU2TOCM_IPC_REGS_CMVIEW Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 16-104. CPU2TOCM_IPC_REGS_CMVIEW Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

16.8.8.1 CMTOCPU2IPCACK Register (Offset = 0h) [Reset = 0h]

CMTOCPU2IPCACK is shown in Figure 16-89 and described in Table 16-105.

Return to the Summary Table.

CMTOCPU2IPCACK Register

Figure 16-89. CMTOCPU2IPCACK Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-105. CMTOCPU2IPCACK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC31 bit. Reset type: CM.RESETn
30	IPC30	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC30 bit. Reset type: CM.RESETn
29	IPC29	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC29 bit. Reset type: CM.RESETn
28	IPC28	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC28 bit. Reset type: CM.RESETn
27	IPC27	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC27 bit. Reset type: CM.RESETn
26	IPC26	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC26 bit. Reset type: CM.RESETn
25	IPC25	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC25 bit. Reset type: CM.RESETn
24	IPC24	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC24 bit. Reset type: CM.RESETn
23	IPC23	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC23 bit. Reset type: CM.RESETn
22	IPC22	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC22 bit. Reset type: CM.RESETn
21	IPC21	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC21 bit. Reset type: CM.RESETn
20	IPC20	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC20 bit. Reset type: CM.RESETn
19	IPC19	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC19 bit. Reset type: CM.RESETn
18	IPC18	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC18 bit. Reset type: CM.RESETn
17	IPC17	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC17 bit. Reset type: CM.RESETn

Table 16-105. CMTOCPU2IPCACK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	IPC16	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC16 bit. Reset type: CM.RESETn
15	IPC15	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC15 bit. Reset type: CM.RESETn
14	IPC14	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC14 bit. Reset type: CM.RESETn
13	IPC13	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC13 bit. Reset type: CM.RESETn
12	IPC12	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC12 bit. Reset type: CM.RESETn
11	IPC11	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC11 bit. Reset type: CM.RESETn
10	IPC10	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC10 bit. Reset type: CM.RESETn
9	IPC9	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC9 bit. Reset type: CM.RESETn
8	IPC8	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC8 bit. Reset type: CM.RESETn
7	IPC7	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC7 bit. Reset type: CM.RESETn
6	IPC6	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC6 bit. Reset type: CM.RESETn
5	IPC5	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC5 bit. Reset type: CM.RESETn
4	IPC4	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC4 bit. Reset type: CM.RESETn
3	IPC3	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC3 bit. Reset type: CM.RESETn
2	IPC2	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC2 bit. Reset type: CM.RESETn
1	IPC1	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC1 bit. Reset type: CM.RESETn
0	IPC0	R-0/W1S	0h	Writing 1 to this bit, will clear CPU2TOCMIPCFLG.IPC0 bit. Reset type: CM.RESETn

16.8.8.2 CPU2TOCMIPCSTS Register (Offset = 2h) [Reset = 0h]

CPU2TOCMIPCSTS is shown in Figure 16-90 and described in Table 16-106.

Return to the Summary Table.

Status of CMTOCPU2IPCFLG register

Figure 16-90. CPU2TOCMIPCSTS Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 16-106. CPU2TOCMIPCSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R	0h	Indicates to CM if the IPC31 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC31 bit. 0: No IPC31 event was set by CPU2 1: An IPC31 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
30	IPC30	R	0h	Indicates to CM if the IPC30 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC30 bit. 0: No IPC30 event was set by CPU2 1: An IPC30 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
29	IPC29	R	0h	Indicates to CM if the IPC29 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC29 bit. 0: No IPC29 event was set by CPU2 1: An IPC29 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
28	IPC28	R	0h	Indicates to CM if the IPC28 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC28 bit. 0: No IPC28 event was set by CPU2 1: An IPC28 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
27	IPC27	R	0h	Indicates to CM if the IPC27 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC27 bit. 0: No IPC27 event was set by CPU2 1: An IPC27 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-106. CPU2TOCMIPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	IPC26	R	0h	Indicates to CM if the IPC26 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC26 bit. 0: No IPC26 event was set by CPU2 1: An IPC26 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
25	IPC25	R	0h	Indicates to CM if the IPC25 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC25 bit. 0: No IPC25 event was set by CPU2 1: An IPC25 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
24	IPC24	R	0h	Indicates to CM if the IPC24 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC24 bit. 0: No IPC24 event was set by CPU2 1: An IPC24 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
23	IPC23	R	0h	Indicates to CM if the IPC23 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC23 bit. 0: No IPC23 event was set by CPU2 1: An IPC23 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
22	IPC22	R	0h	Indicates to CM if the IPC22 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC22 bit. 0: No IPC22 event was set by CPU2 1: An IPC22 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
21	IPC21	R	0h	Indicates to CM if the IPC21 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC21 bit. 0: No IPC21 event was set by CPU2 1: An IPC21 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
20	IPC20	R	0h	Indicates to CM if the IPC20 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC20 bit. 0: No IPC20 event was set by CPU2 1: An IPC20 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
19	IPC19	R	0h	Indicates to CM if the IPC19 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC19 bit. 0: No IPC19 event was set by CPU2 1: An IPC19 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-106. CPU2TOCMIPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	IPC18	R	0h	Indicates to CM if the IPC18 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC18 bit. 0: No IPC18 event was set by CPU2 1: An IPC18 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
17	IPC17	R	0h	Indicates to CM if the IPC17 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC17 bit. 0: No IPC17 event was set by CPU2 1: An IPC17 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
16	IPC16	R	0h	Indicates to CM if the IPC16 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC16 bit. 0: No IPC16 event was set by CPU2 1: An IPC16 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
15	IPC15	R	0h	Indicates to CM if the IPC15 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC15 bit. 0: No IPC15 event was set by CPU2 1: An IPC15 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
14	IPC14	R	0h	Indicates to CM if the IPC14 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC14 bit. 0: No IPC14 event was set by CPU2 1: An IPC14 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
13	IPC13	R	0h	Indicates to CM if the IPC13 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC13 bit. 0: No IPC13 event was set by CPU2 1: An IPC13 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
12	IPC12	R	0h	Indicates to CM if the IPC12 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC12 bit. 0: No IPC12 event was set by CPU2 1: An IPC12 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
11	IPC11	R	0h	Indicates to CM if the IPC11 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC11 bit. 0: No IPC11 event was set by CPU2 1: An IPC11 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-106. CPU2TOCMIPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	IPC10	R	0h	Indicates to CM if the IPC10 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC10 bit. 0: No IPC10 event was set by CPU2 1: An IPC10 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
9	IPC9	R	0h	Indicates to CM if the IPC9 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC9 bit. 0: No IPC9 event was set by CPU2 1: An IPC9 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
8	IPC8	R	0h	Indicates to CM if the IPC8 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC8 bit. 0: No IPC8 event was set by CPU2 1: An IPC8 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
7	IPC7	R	0h	Indicates to CM if the IPC7 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC7 bit. 0: No IPC7 event was set by CPU2 1: An IPC7 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
6	IPC6	R	0h	Indicates to CM if the IPC6 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC6 bit. 0: No IPC6 event was set by CPU2 1: An IPC6 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
5	IPC5	R	0h	Indicates to CM if the IPC5 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC5 bit. 0: No IPC5 event was set by CPU2 1: An IPC5 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
4	IPC4	R	0h	Indicates to CM if the IPC4 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC4 bit. 0: No IPC4 event was set by CPU2 1: An IPC4 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
3	IPC3	R	0h	Indicates to CM if the IPC3 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC3 bit. 0: No IPC3 event was set by CPU2 1: An IPC3 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

Table 16-106. CPU2TOCMIPCSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	IPC2	R	0h	Indicates to CM if the IPC2 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC2 bit. 0: No IPC2 event was set by CPU2 1: An IPC2 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
1	IPC1	R	0h	Indicates to CM if the IPC1 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC1 bit. 0: No IPC1 event was set by CPU2 1: An IPC1 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn
0	IPC0	R	0h	Indicates to CM if the IPC0 event flag was set by CPU2. Reflects the state of CPU2TOCMIPCFLG.IPC0 bit. 0: No IPC0 event was set by CPU2 1: An IPC0 event was set by CPU2 Notes [1] IPC event flags 0-7 will trigger interrupts. Reset type: CPU2.SYSRSn

16.8.8.3 CMTOCPU2IPCSET Register (Offset = 4h) [Reset = 0h]

CMTOCPU2IPCSET is shown in Figure 16-91 and described in Table 16-107.

Return to the Summary Table.

Set CMTOCPU2IPCFLG register

Figure 16-91. CMTOCPU2IPCSET Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-107. CMTOCPU2IPCSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC31 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
30	IPC30	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC30 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
29	IPC29	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC29 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
28	IPC28	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC28 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
27	IPC27	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC27 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-107. CMTOCPU2IPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	IPC26	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC26 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
25	IPC25	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC25 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
24	IPC24	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC24 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
23	IPC23	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC23 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
22	IPC22	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC22 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
21	IPC21	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC21 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
20	IPC20	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC20 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
19	IPC19	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC19 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
18	IPC18	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC18 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
17	IPC17	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC17 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-107. CMTOCPU2IPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	IPC16	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC16 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
15	IPC15	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC15 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
14	IPC14	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC14 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
13	IPC13	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC13 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
12	IPC12	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC12 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
11	IPC11	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC11 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
10	IPC10	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC10 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
9	IPC9	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC9 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
8	IPC8	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC8 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
7	IPC7	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC7 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-107. CMTOCPU2IPCSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	IPC6	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC6 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
5	IPC5	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC5 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
4	IPC4	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC4 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
3	IPC3	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC3 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
2	IPC2	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC2 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
1	IPC1	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC1 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
0	IPC0	R-0/W1S	0h	Writing 1 to this bit sets the CMTOCPU2IPCFLG.IPC0 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

16.8.8.4 CMTOCPU2IPCCLR Register (Offset = 6h) [Reset = 0h]

CMTOCPU2IPCCLR is shown in Figure 16-92 and described in Table 16-108.

Return to the Summary Table.

Clear CMTOCPU2IPCFLG register

Figure 16-92. CMTOCPU2IPCCLR Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-108. CMTOCPU2IPCCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC31 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
30	IPC30	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC30 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
29	IPC29	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC29 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
28	IPC28	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC28 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
27	IPC27	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC27 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-108. CMTOCPU2IPCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	IPC26	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC26 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
25	IPC25	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC25 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
24	IPC24	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC24 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
23	IPC23	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC23 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
22	IPC22	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC22 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
21	IPC21	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC21 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
20	IPC20	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC20 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
19	IPC19	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC19 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
18	IPC18	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC18 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
17	IPC17	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC17 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-108. CMTOCPU2IPCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	IPC16	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC16 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
15	IPC15	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC15 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
14	IPC14	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC14 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
13	IPC13	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC13 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
12	IPC12	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC12 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
11	IPC11	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC11 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
10	IPC10	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC10 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
9	IPC9	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC9 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
8	IPC8	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC8 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
7	IPC7	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC7 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-108. CMTOCPU2IPCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	IPC6	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC6 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
5	IPC5	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC5 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
4	IPC4	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC4 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
3	IPC3	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC3 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
2	IPC2	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC2 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
1	IPC1	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC1 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
0	IPC0	R-0/W1S	0h	Writing 1 to this bit cleare the CMTOCPU2IPCFLG.IPC0 event flag for CPU2. Writing 0 has no effect. Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

16.8.8.5 CMTOCPU2IPCFLG Register (Offset = 8h) [Reset = 0h]

CMTOCPU2IPCFLG is shown in Figure 16-93 and described in Table 16-109.

Return to the Summary Table.

CMTOCPU2IPCFLG Register

Figure 16-93. CMTOCPU2IPCFLG Register

31	30	29	28	27	26	25	24
IPC31	IPC30	IPC29	IPC28	IPC27	IPC26	IPC25	IPC24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
IPC23	IPC22	IPC21	IPC20	IPC19	IPC18	IPC17	IPC16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
IPC15	IPC14	IPC13	IPC12	IPC11	IPC10	IPC9	IPC8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
IPC7	IPC6	IPC5	IPC4	IPC3	IPC2	IPC1	IPC0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 16-109. CMTOCPU2IPCFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31	IPC31	R	0h	0: No IPC31 event request to CPU2 1: IPC31 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
30	IPC30	R	0h	0: No IPC30 event request to CPU2 1: IPC30 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
29	IPC29	R	0h	0: No IPC29 event request to CPU2 1: IPC29 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
28	IPC28	R	0h	0: No IPC28 event request to CPU2 1: IPC28 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
27	IPC27	R	0h	0: No IPC27 event request to CPU2 1: IPC27 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
26	IPC26	R	0h	0: No IPC26 event request to CPU2 1: IPC26 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-109. CMTOCPU2IPCFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	IPC25	R	0h	0: No IPC25 event request to CPU2 1: IPC25 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
24	IPC24	R	0h	0: No IPC24 event request to CPU2 1: IPC24 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
23	IPC23	R	0h	0: No IPC23 event request to CPU2 1: IPC23 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
22	IPC22	R	0h	0: No IPC22 event request to CPU2 1: IPC22 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
21	IPC21	R	0h	0: No IPC21 event request to CPU2 1: IPC21 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
20	IPC20	R	0h	0: No IPC20 event request to CPU2 1: IPC20 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
19	IPC19	R	0h	0: No IPC19 event request to CPU2 1: IPC19 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
18	IPC18	R	0h	0: No IPC18 event request to CPU2 1: IPC18 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
17	IPC17	R	0h	0: No IPC17 event request to CPU2 1: IPC17 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
16	IPC16	R	0h	0: No IPC16 event request to CPU2 1: IPC16 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
15	IPC15	R	0h	0: No IPC15 event request to CPU2 1: IPC15 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
14	IPC14	R	0h	0: No IPC14 event request to CPU2 1: IPC14 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-109. CMTOCPU2IPCFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	IPC13	R	0h	0: No IPC13 event request to CPU2 1: IPC13 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
12	IPC12	R	0h	0: No IPC12 event request to CPU2 1: IPC12 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
11	IPC11	R	0h	0: No IPC11 event request to CPU2 1: IPC11 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
10	IPC10	R	0h	0: No IPC10 event request to CPU2 1: IPC10 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
9	IPC9	R	0h	0: No IPC9 event request to CPU2 1: IPC9 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
8	IPC8	R	0h	0: No IPC8 event request to CPU2 1: IPC8 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
7	IPC7	R	0h	0: No IPC7 event request to CPU2 1: IPC7 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
6	IPC6	R	0h	0: No IPC6 event request to CPU2 1: IPC6 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
5	IPC5	R	0h	0: No IPC5 event request to CPU2 1: IPC5 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
4	IPC4	R	0h	0: No IPC4 event request to CPU2 1: IPC4 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
3	IPC3	R	0h	0: No IPC3 event request to CPU2 1: IPC3 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
2	IPC2	R	0h	0: No IPC2 event request to CPU2 1: IPC2 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

Table 16-109. CMTOCPU2IPCFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	IPC1	R	0h	0: No IPC1 event request to CPU2 1: IPC1 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn
0	IPC0	R	0h	0: No IPC0 event request to CPU2 1: IPC0 event request to CPU2 Notes: [1] IPC event flags 0-7 will trigger interrupts. Reset type: CM.RESETn

16.8.8.6 IPCCOUNTERL Register (Offset = Ch) [Reset = 0h]

IPCCOUNTERL is shown in Figure 16-94 and described in Table 16-110.

Return to the Summary Table.

IPC Counter Low Register

Figure 16-94. IPCCOUNTERL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT																															
R-0h																															

Table 16-110. IPCCOUNTERL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNT	R	0h	This is the lower 32-bits of free running 64 bit timestamp counter clocked by the PLLSYSCLK. Reset type: CPU1.SYSRSn

16.8.8.7 IPCCOUNTERH Register (Offset = Eh) [Reset = 0h]

IPCCOUNTERH is shown in Figure 16-95 and described in Table 16-111.

Return to the Summary Table.

IPC Counter High Register

Figure 16-95. IPCCOUNTERH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT																															
R-0h																															

Table 16-111. IPCCOUNTERH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNT	R	0h	This is the upper 32-bits of free running 64 bit timestamp counter clocked by the PLLSYSCLK. Reset type: CPU1.SYSRSn

16.8.8.8 CPU2TOCMIPCRECVCOM Register (Offset = 10h) [Reset = 0h]

CPU2TOCMIPCRECVCOM is shown in Figure 16-96 and described in Table 16-112.

Return to the Summary Table.

Refelects the value in CPU2TOCMIPCSENDCOM Register

Figure 16-96. CPU2TOCMIPCRECVCOM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMMAND																															
R-0h																															

Table 16-112. CPU2TOCMIPCRECVCOM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COMMAND	R	0h	Refelects the state of CPU2TOCMIPCSENDCOM register Reset type: CPU2.SYSRSn

16.8.8.9 CPU2TOCMIPCRECVADDR Register (Offset = 12h) [Reset = 0h]

CPU2TOCMIPCRECVADDR is shown in Figure 16-97 and described in Table 16-113.

Return to the Summary Table.

Refelects the value in CPU2TOCMIPCSENADDR Register

Figure 16-97. CPU2TOCMIPCRECVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS																															
R-0h																															

Table 16-113. CPU2TOCMIPCRECVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRESS	R	0h	Refelects the state of CPU2TOCMIPCSENDADDR register Reset type: CPU2.SYSRSn

16.8.8.10 CPU2TOCMIPCRECVDATA Register (Offset = 14h) [Reset = 0h]

CPU2TOCMIPCRECVDATA is shown in Figure 16-98 and described in Table 16-114.

Return to the Summary Table.

Refelects the value in CPU2TOCMIPCSENDDATA Register

Figure 16-98. CPU2TOCMIPCRECVDATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDATA																															
R-0h																															

Table 16-114. CPU2TOCMIPCRECVDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WDATA	R	0h	Refelects the state of CPU2TOCMIPCSENDDATA register Reset type: CPU2.SYSRSn

16.8.8.11 CMTOCPU2IPCREPLY Register (Offset = 16h) [Reset = 0h]

CMTOCPU2IPCREPLY is shown in Figure 16-99 and described in Table 16-115.

Return to the Summary Table.

Reply from CM to CPU2TOCMIPCSENDCOM command

Figure 16-99. CMTOCPU2IPCREPLY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDATA																															
R/W-0h																															

Table 16-115. CMTOCPU2IPCREPLY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RDATA	R/W	0h	This is a general purpose register used to send a reply to CPU2 to CM command from CM. Note: This register is not writable from CPU2. Reset type: CM.RESETn

16.8.8.12 CMTOCPU2IPCSENDCOM Register (Offset = 18h) [Reset = 0h]

CMTOCPU2IPCSENDCOM is shown in Figure 16-100 and described in Table 16-116.

Return to the Summary Table.

CM to CPU2 IPC Command

Figure 16-100. CMTOCPU2IPCSENDCOM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMMAND																															
R/W-0h																															

Table 16-116. CMTOCPU2IPCSENDCOM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COMMAND	R/W	0h	This is a general purpose register used to send software-defined commands to CPU2 from CM. Reset type: CM.RESETn

16.8.8.13 CMTOCPU2IPCSENDADDR Register (Offset = 1Ah) [Reset = 0h]

CMTOCPU2IPCSENDADDR is shown in Figure 16-101 and described in Table 16-117.

Return to the Summary Table.

CM to CPU2 IPC Address

Figure 16-101. CMTOCPU2IPCSENDADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS																															
R/W-0h																															

Table 16-117. CMTOCPU2IPCSENDADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRESS	R/W	0h	This is a general purpose register used to send software-defined address to CPU2 from CM. Reset type: CM.RESETn

16.8.8.14 CMTOCPU2IPCSENDDATA Register (Offset = 1Ch) [Reset = 0h]

CMTOCPU2IPCSENDDATA is shown in Figure 16-102 and described in Table 16-118.

Return to the Summary Table.

CM to CPU2 IPC Data

Figure 16-102. CMTOCPU2IPCSENDDATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDATA																															
R/W-0h																															

Table 16-118. CMTOCPU2IPCSENDDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WDATA	R/W	0h	This is a general purpose register used to send software-defined data to CPU2 from CM. Reset type: CM.RESETn

16.8.8.15 CPU2TOCMIPCREPLY Register (Offset = 1Eh) [Reset = 0h]

CPU2TOCMIPCREPLY is shown in Figure 16-103 and described in Table 16-119.

Return to the Summary Table.

Reply from CPU2 to CMTOCPU2IPCSENDCOM command request

Figure 16-103. CPU2TOCMIPCREPLY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDATA																															
R/W-0h																															

Table 16-119. CPU2TOCMIPCREPLY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RDATA	R/W	0h	This is a general purpose register used to send a reply to CM to CPU2 command from CPU2. Note: This register is not writable from CM. Reset type: CPU2.SYSRSn

16.8.9 IPC Registers to Driverlib Functions

Table 16-120. IPC Registers to Driverlib Functions

File	Driverlib Function
CPU1TOCMIPCACK	
-	
CMTOCPU1IPCSTS	
-	
CPU1TOCMIPCSET	
-	
CPU1TOCMIPCCLR	
-	
CPU1TOCMIPCFLG	
-	
COUNTERL	
-	
COUNTERH	
-	
CPU1TOCMIPCSENDCOM	
-	
CPU1TOCMIPCSENDADDR	
-	
CPU1TOCMIPCSENDDATA	
-	
CMTOCPU1IPCREPLY	
-	
CMTOCPU1IPCRECVCOM	
-	
CMTOCPU1IPCRECVADDR	
-	
CMTOCPU1IPCRECVDATA	
-	

Table 16-120. IPC Registers to Driverlib Functions (continued)

File	Driverlib Function
CPU1TOCMIPCREPLY	
-	
CMTOCPU1IPCBOOTSTS	
-	
CPU1TOCMIPCBOOTMODE	
-	
CMTOCPU1IPCACK	
-	
CPU1TOCMIPCSTS	
-	
CMTOCPU1IPCSET	
-	
CMTOCPU1IPCCLR	
-	
CMTOCPU1IPCFLG	
-	
COUNTERL	
-	
COUNTERH	
-	
CPU1TOCMIPCRECVCOM	
-	
CPU1TOCMIPCRECVADDR	
-	
CPU1TOCMIPCRECVDATA	
-	
CMTOCPU1IPCREPLY	
-	
CMTOCPU1IPCSENDCOM	
-	
CMTOCPU1IPCSENDADDR	
-	
CMTOCPU1IPCSENDDATA	
-	
CPU1TOCMIPCREPLY	
-	
CMTOCPU1IPCBOOTSTS	
-	
CPU1TOCMIPCBOOTMODE	
-	
PUMPREQUEST	
-	
CPU2TOCMIPCACK	
-	
CMTOCPU2IPCSTS	

Table 16-120. IPC Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
CPU2TOCMIPCSET	
-	
CPU2TOCMIPCCLR	
-	
CPU2TOCMIPCFLG	
-	
COUNTERL	
-	
COUNTERH	
-	
CPU2TOCMIPCSENDCOM	
-	
CPU2TOCMIPCSENDADDR	
-	
CPU2TOCMIPCSENDDATA	
-	
CMTOCPU2IPCREPLY	
-	
CMTOCPU2IPCRECVCOM	
-	
CMTOCPU2IPCRECVADDR	
-	
CMTOCPU2IPCRECVDATA	
-	
CPU2TOCMIPCREPLY	
-	
CMTOCPU2IPCACK	
-	
CPU2TOCMIPCSTS	
-	
CMTOCPU2IPCSET	
-	
CMTOCPU2IPCCLR	
-	
CMTOCPU2IPCFLG	
-	
COUNTERL	
-	
COUNTERH	
-	
CPU2TOCMIPCRECVCOM	
-	
CPU2TOCMIPCRECVADDR	
-	

Table 16-120. IPC Registers to Driverlib Functions (continued)

File	Driverlib Function
CPU2TOCMIPCRECVDATA	
-	
CMTOCPU2IPCREPLY	
-	
CMTOCPU2IPCSENDCOM	
-	
CMTOCPU2IPCSENDADDR	
-	
CMTOCPU2IPCSENDDATA	
-	
CPU2TOCMIPCREPLY	
-	
CPU1TOCPU2IPCACK	
-	
CPU2TOCPU1IPCSTS	
-	
CPU1TOCPU2IPCSET	
-	
CPU1TOCPU2IPCCLR	
-	
CPU1TOCPU2IPCFLG	
-	
COUNTERL	
-	
COUNTERH	
-	
CPU1TOCPU2IPCSENDCOM	
-	
CPU1TOCPU2IPCSENDADDR	
-	
CPU1TOCPU2IPCSENDDATA	
-	
CPU2TOCPU1IPCREPLY	
-	
CPU2TOCPU1IPCRECVCOM	
-	
CPU2TOCPU1IPCRECVADDR	
-	
CPU2TOCPU1IPCRECVDATA	
-	
CPU1TOCPU2IPCREPLY	
-	
CPU2TOCPU1IPCBOOTSTS	
-	
CPU1TOCPU2IPCBOOTMODE	

Table 16-120. IPC Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
PUMPREQUEST	
-	
CPU2TOCPU1IPCACK	
-	
CPU1TOCPU2IPCSTS	
-	
CPU2TOCPU1IPCSET	
-	
CPU2TOCPU1IPCCLR	
-	
CPU2TOCPU1IPCFLG	
-	
COUNTERL	
-	
COUNTERH	
-	
CPU1TOCPU2IPCRECVCOM	
-	
CPU1TOCPU2IPCRECVADDR	
-	
CPU1TOCPU2IPCRECVDATA	
-	
CPU2TOCPU1IPCREPLY	
-	
CPU2TOCPU1IPCSENDCOM	
-	
CPU2TOCPU1IPCSENDADDR	
-	
CPU2TOCPU1IPCSENDDATA	
-	
CPU1TOCPU2IPCREPLY	
-	
CPU2TOCPU1IPCBOOTSTS	
-	
CPU1TOCPU2IPCBOOTMODE	
-	
PUMPREQUEST	
-	

This page intentionally left blank.

Chapter 17

Crossbar (X-BAR)

The crossbars (referred to as X-BAR throughout this chapter) provide flexibility to connect device inputs, outputs, and internal resources in a variety of configurations.

The device contains a total of six X-BARs:

- Input X-BAR
- CLB Input X-BAR
- Output X-BAR
- CLB Output X-BAR
- CLB X-BAR
- ePWM X-BAR

Each of the X-BARs is named according to where the X-BAR takes signals. For example, the Input X-BAR and CLB Input X-BAR bring external signals “in” to the device. The Output X-BAR and CLB Output X-BAR take internal signals “out” of the device to a GPIO. The CLB X-BAR and ePWM X-BAR take signals to the CLB and ePWM modules, respectively.

17.1 Input X-BAR and CLB Input X-BAR

On this device, the Input X-BAR is used to route signals from a GPIO to many different IP blocks such as the ADC, eCAP, ePWM, and external interrupts. The input of each Input X-BAR instance (INPUTx) can be any GPIO, while the output of each instance connects to various IP blocks in the device. The digital input of AIOs are also available as inputs to the Input X-BAR. This flexibility relieves some of the constraints on peripheral muxing by allowing the user to connect any GPIO to the specified outputs of each Input X-BAR instance. Note that the GPIO selected by the Input X-BAR can be configured as either an input or an output. The Input X-BAR simply connects the signal on the input buffer to the output of the selected Input X-BAR instance. Therefore, you can do things such as route the output of an ePWM to the eCAP module for a frequency test).

The Input X-BAR is configured by way of the INPUTxSELECT registers. The destinations for each INPUTx are shown in Figure 17-1 and Table 17-1. For additional details on how each Input X-BAR connects to other IP blocks throughout the device, look for references to Input X-BAR in the chapter associated with that IP. Note that the destinations of each INPUTx are fixed and are not user-configurable. For more information on configuring the Input X-BAR, see the INPUT_XBAR_REGS register definitions in the *XBAR Registers* section.

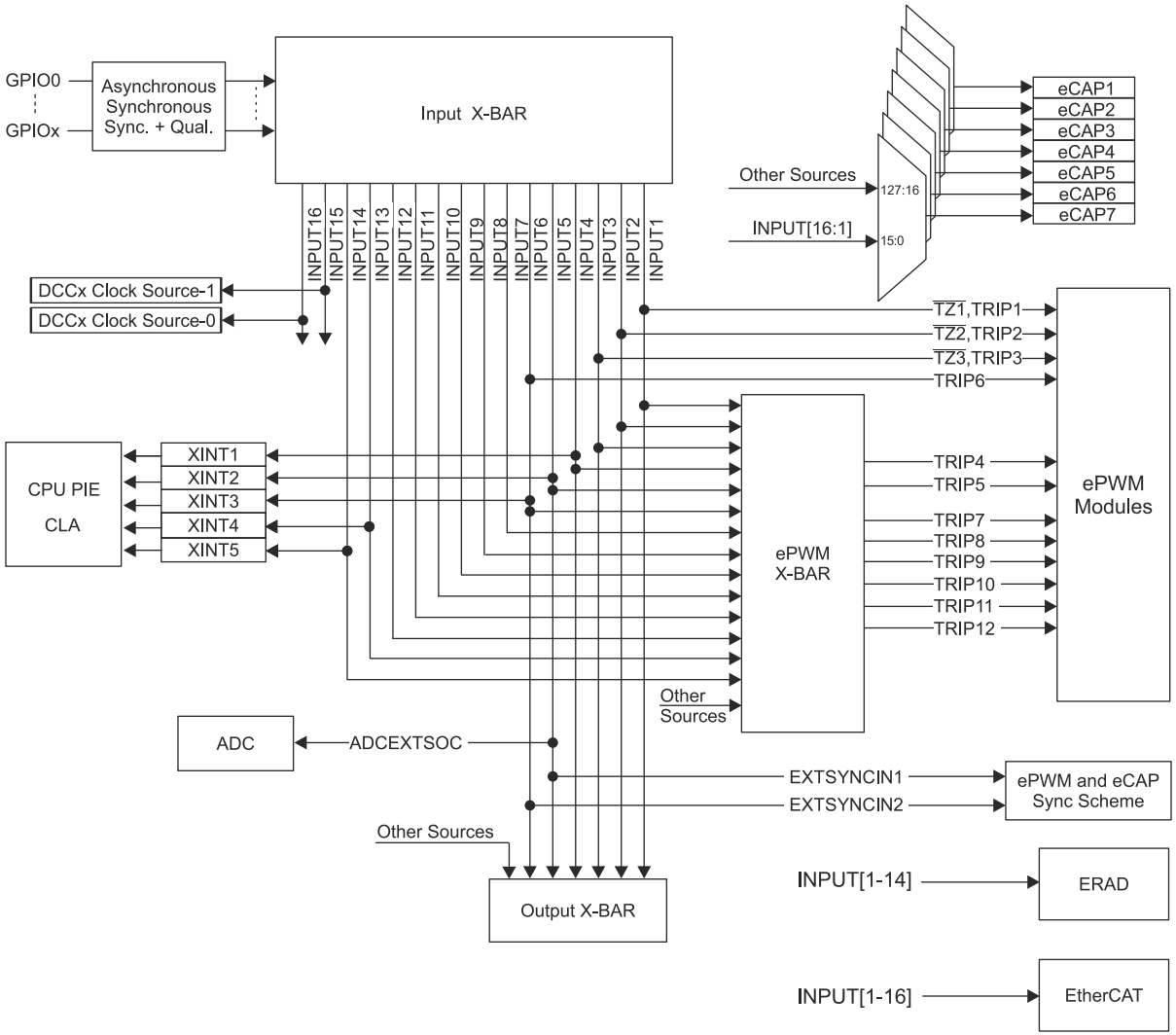


Figure 17-1. Input X-BAR

Note

INPUTXBARx, INPUTXBAR_INPUTx, and INPUTx (when referenced in the context of Input X-BAR) are equivalent in all C2000 software and documentation.

Table 17-1. Input X-BAR Destinations

INPUT	ECAP / HRCAP	EPWM X-BAR	CLB X-BAR	OUTPUT X-BAR	CPU XINT	EPWM TRIP	ADC START OF CONVERSION	EPWM / ECAP SYNC	DCC	Ether CAT	ERAD
1	Yes	Yes	Yes	Yes	-	TZ1,TRIP1	-	-	-	Yes	Yes
2	Yes	Yes	Yes	Yes	-	TZ2,TRIP2	-	-	-	Yes	Yes
3	Yes	Yes	Yes	Yes	-	TZ3,TRIP3	-	-	-	Yes	Yes
4	Yes	Yes	Yes	Yes	XINT1	-	-	-	-	Yes	Yes
5	Yes	Yes	Yes	Yes	XINT2	-	ADCEXTSOC	EXTSYNCIN 1	-	Yes	Yes
6	Yes	Yes	Yes	Yes	XINT3	TRIP6	-	EXTSYNCIN 2	-	Yes	Yes
7	Yes	Yes	Yes	-	-	-	-	-	-	Yes	Yes
8	Yes	Yes	Yes	-	-	-	-	-	-	Yes	Yes
9	Yes	Yes	Yes	-	-	-	-	-	-	Yes	Yes
10	Yes	Yes	Yes	-	-	-	-	-	-	Yes	Yes
11	Yes	Yes	Yes	-	-	-	-	-	-	Yes	Yes
12	Yes	Yes	Yes	-	-	-	-	-	-	Yes	Yes
13	Yes	Yes	Yes	-	XINT4	-	-	-	-	Yes	Yes
14	Yes	Yes	Yes	-	XINT5	-	-	-	-	Yes	Yes
15	Yes	-	-	-	-	-	-	-	CLK1	Yes	-
16	Yes	-	-	-	-	-	-	-	CLK0	Yes	-

17.1.1 CLB Input X-BAR

The CLB Input X-BAR is architecturally identical to the Input X-BAR. The only difference is the destination for each INPUTx. The destination for each INPUTx is only the CLB Tiles as shown in Table 17-2. This allows for GPIOs to be accessed by the CLB tiles without using the combination of Input X-BAR and CLB X-BAR.

Note

Signals routed into the CLB using the XBAR must be synchronized within the CLB.

Table 17-2. CLB Input X-BAR Destinations

INPUT	CLB
1	Yes
2	Yes
3	Yes
4	Yes
5	Yes
6	Yes
7	Yes
8	Yes
9	Yes
10	Yes
11	Yes
12	Yes
13	Yes
14	Yes
15	Yes
16	Yes

17.2 ePWM, CLB, and GPIO Output X-BAR

This section describes the ePWM, CLB, and GPIO Output X-BAR.

17.2.1 ePWM X-BAR

The ePWM X-BAR brings signals to the ePWM modules. Specifically, the ePWM X-BAR is connected to the Digital Compare (DC) submodule of each ePWM module for actions such as tripzones and syncing. Refer to the *Enhanced Pulse Width Modulator (ePWM)* chapter for more information on additional ways the DC submodule can be used. Figure 17-2 shows the architecture of the ePWM X-BAR. Note that the architecture of the ePWM X-BAR is identical to the architecture of the GPIO Output X-BAR (with the exception of the output latch).

17.2.1.1 ePWM X-BAR Architecture

The ePWM X-BAR has eight outputs that are routed to each ePWM module. Figure 17-2 represents the architecture of a single output, but this output is identical to the architecture of all of the other outputs.

First, determine the signals that can be passed to the ePWM by referencing Table 17-3. Select up to one signal per mux for each TRIPx output. Select the inputs to ePWM X-BAR using the TRIPxMUX0TO15CFG and TRIPxMUX16TO31CFG registers. To pass any signal through to the ePWM, enable it via the TRIPxMUXENABLE register. All signals that are enabled are logically ORed before being passed on to the respective TRIPx signal on the ePWM. To optionally invert the signal, use the TRIPOUTINV register.

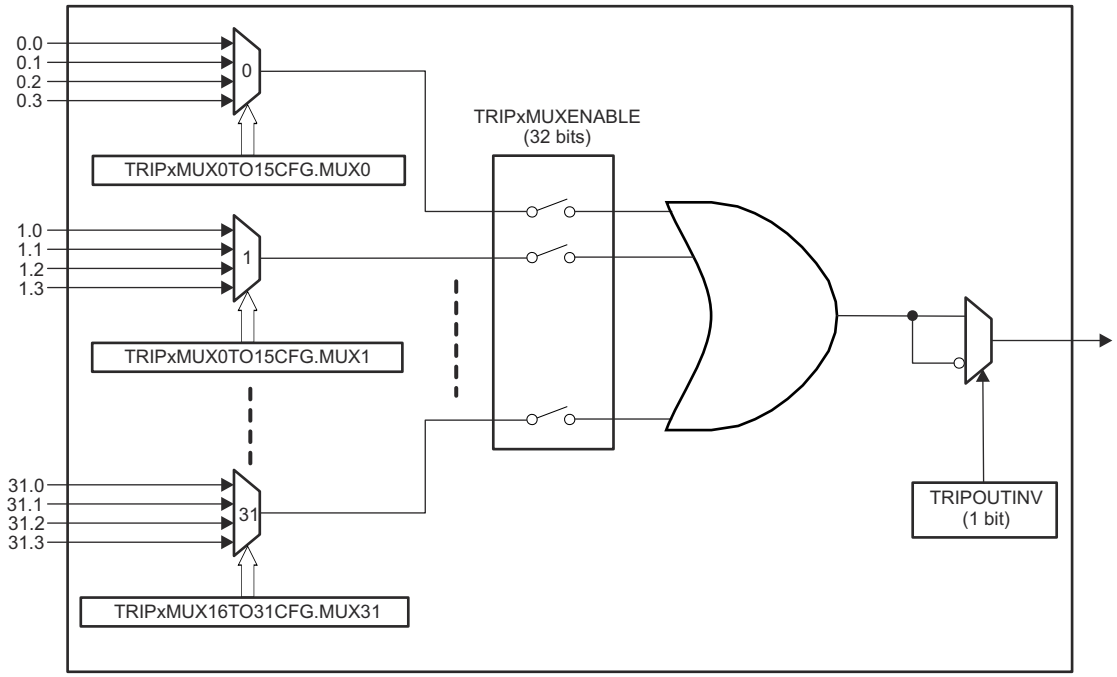


Figure 17-2. ePWM X-BAR Architecture - Single Output

Note

Do not use "Reserved" signals in your application.

Table 17-3. EPWM X-BAR Mux Configuration Table

Mux	0	1	2	3
G0	CMPSS1_CTRIPH	CMPSS1_CTRIPH_OR_CTRIPL	ADCAEVT1	ECAP1_OUT
G1	CMPSS1_CTRIPL	INPUTXBAR1	CLB1_OUT12	ADCCEVT1
G2	CMPSS2_CTRIPH	CMPSS2_CTRIPH_OR_CTRIPL	ADCAEVT2	ECAP2_OUT
G3	CMPSS2_CTRIPL	INPUTXBAR2	CLB1_OUT13	ADCCEVT2
G4	CMPSS3_CTRIPH	CMPSS3_CTRIPH_OR_CTRIPL	ADCAEVT3	ECAP3_OUT
G5	CMPSS3_CTRIPL	INPUTXBAR3	CLB2_OUT12	ADCCEVT3
G6	CMPSS4_CTRIPH	CMPSS4_CTRIPH_OR_CTRIPL	ADCAEVT4	ECAP4_OUT
G7	CMPSS4_CTRIPL	INPUTXBAR4	CLB2_OUT13	ADCCEVT4
G8	CMPSS5_CTRIPH	CMPSS5_CTRIPH_OR_CTRIPL	ADCBEVT1	ECAP5_OUT
G9	CMPSS5_CTRIPL	INPUTXBAR5	CLB3_OUT12	ADCDEVT1
G10	CMPSS6_CTRIPH	CMPSS6_CTRIPH_OR_CTRIPL	ADCBEVT2	ECAP6_OUT
G11	CMPSS6_CTRIPL	INPUTXBAR6	CLB3_OUT13	ADCDEVT2
G12	CMPSS7_CTRIPH	CMPSS7_CTRIPH_OR_CTRIPL	ADCBEVT3	ECAP7_OUT
G13	CMPSS7_CTRIPL	ADCSOCAO	CLB4_OUT12	ADCDEVT3
G14	CMPSS8_CTRIPH	CMPSS8_CTRIPH_OR_CTRIPL	ADCBEVT4	EXTSYNCOUT
G15	CMPSS8_CTRIPL	ADCSOCBO	CLB4_OUT13	ADCDEVT4
G16	SD1FLT1_CEVT1	SD1FLT1_CEVT1_OR_CEVT2	Reserved	ERRORSTS
G17	SD1FLT1_CEVT2	INPUTXBAR7	CLB5_OUT12	CLAHALT
G18	SD1FLT2_CEVT1	SD1FLT2_CEVT1_OR_CEVT2	Reserved	ECAT_SYNC0
G19	SD1FLT2_CEVT2	INPUTXBAR8	CLB5_OUT13	ECAT_SYNC1
G20	SD1FLT3_CEVT1	SD1FLT3_CEVT1_OR_CEVT2	Reserved	Reserved
G21	SD1FLT3_CEVT2	INPUTXBAR9	CLB6_OUT12	Reserved
G22	SD1FLT4_CEVT1	SD1FLT4_CEVT1_OR_CEVT2	Reserved	Reserved
G23	SD1FLT4_CEVT2	INPUTXBAR10	CLB6_OUT13	Reserved
G24	SD2FLT1_CEVT1	SD2FLT1_CEVT1_OR_CEVT2	Reserved	Reserved
G25	SD2FLT1_CEVT2	INPUTXBAR11	MCANA_FEVT0	CLB7_OUT12
G26	SD2FLT2_CEVT1	SD2FLT2_CEVT1_OR_CEVT2	Reserved	Reserved
G27	SD2FLT2_CEVT2	INPUTXBAR12	MCANA_FEVT1	CLB7_OUT13
G28	SD2FLT3_CEVT1	SD2FLT3_CEVT1_OR_CEVT2	Reserved	Reserved
G29	SD2FLT3_CEVT2	INPUTXBAR13	MCANA_FEVT2	CLB8_OUT12
G30	SD2FLT4_CEVT1	SD2FLT4_CEVT1_OR_CEVT2	Reserved	Reserved
G31	SD2FLT4_CEVT2	INPUTXBAR14	Reserved	CLB8_OUT13

17.2.2 CLB X-BAR

The CLB X-BAR brings signals to the CLB modules. Figure 17-3 shows the architecture of the CLB X-BAR. Note that the architecture of the CLB X-BAR is identical to the architecture of the GPIO Output X-BAR (with the exception of the output latch).

17.2.2.1 CLB X-BAR Architecture

The CLB X-BAR has eight outputs that are routed to each CLB module. Figure 17-3 represents the architecture of a single output, but the output is identical to the architecture of all of the other outputs.

First, determine the signals that can be passed to the CLB by referencing Table 17-4 . Select up to one signal per mux (31 total muxes) for each AUXSIGx output. Select the inputs to each mux using the AUXSIGxMUX0TO15CFG and AUXSIGxMUX16TO31CFG registers. To pass any signal through to the CLB, enable the mux in the AUXSIGxMUXENABLE register. All muxes that are enabled are logically ORed before being passed on to the respective AUXSIGx signal on the CLB. To optionally invert the signal, use the AUXSIGOUTINV register.

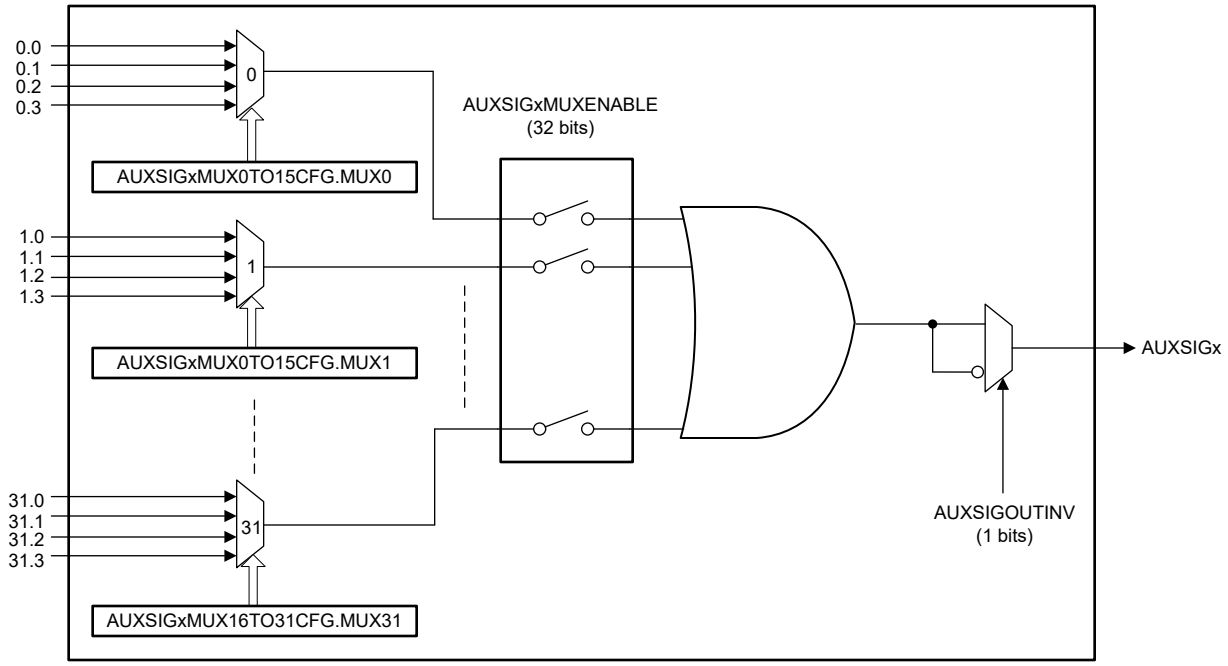


Figure 17-3. CLB X-BAR Architecture - Single Output

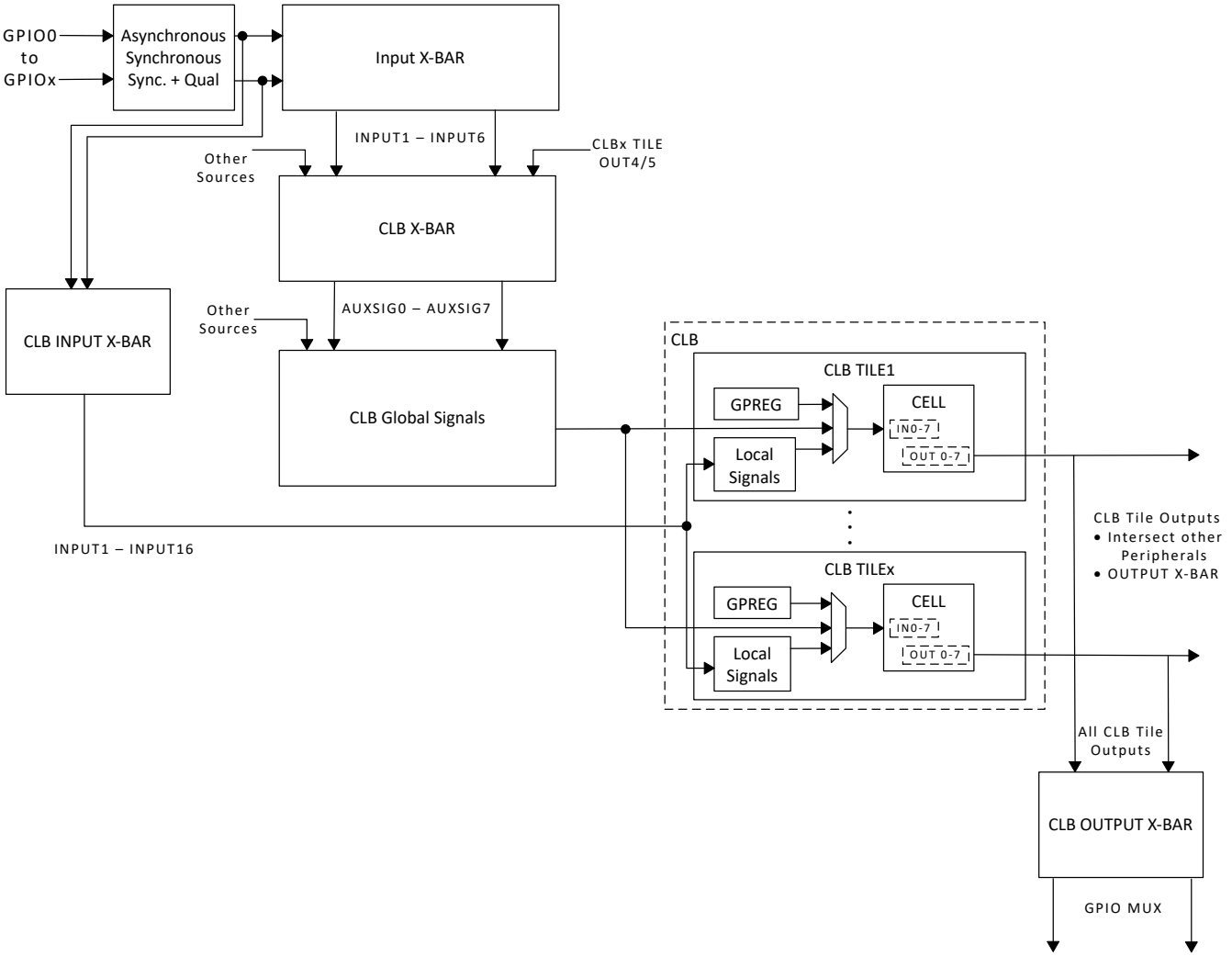


Figure 17-4. GPIO to CLB Tile Connections

Table 17-4. CLB X-BAR Mux Configuration Table

Mux	0	1	2	3
G0	CMPSS1_CTRIPH	CMPSS1_CTRIPH_OR_CTRIPL	ADCAEVT1	ECAP1_OUT
G1	CMPSS1_CTRIPL	INPUTXBAR1	CLB1_OUT12	ADCCEVT1
G2	CMPSS2_CTRIPH	CMPSS2_CTRIPH_OR_CTRIPL	ADCAEVT2	ECAP2_OUT
G3	CMPSS2_CTRIPL	INPUTXBAR2	CLB1_OUT13	ADCCEVT2
G4	CMPSS3_CTRIPH	CMPSS3_CTRIPH_OR_CTRIPL	ADCAEVT3	ECAP3_OUT
G5	CMPSS3_CTRIPL	INPUTXBAR3	CLB2_OUT12	ADCCEVT3
G6	CMPSS4_CTRIPH	CMPSS4_CTRIPH_OR_CTRIPL	ADCAEVT4	ECAP4_OUT
G7	CMPSS4_CTRIPL	INPUTXBAR4	CLB2_OUT13	ADCCEVT4
G8	CMPSS5_CTRIPH	CMPSS5_CTRIPH_OR_CTRIPL	ADCBEVT1	ECAP5_OUT
G9	CMPSS5_CTRIPL	INPUTXBAR5	CLB3_OUT12	ADCDEVT1
G10	CMPSS6_CTRIPH	CMPSS6_CTRIPH_OR_CTRIPL	ADCBEVT2	ECAP6_OUT
G11	CMPSS6_CTRIPL	INPUTXBAR6	CLB3_OUT13	ADCDEVT2
G12	CMPSS7_CTRIPH	CMPSS7_CTRIPH_OR_CTRIPL	ADCBEVT3	ECAP7_OUT
G13	CMPSS7_CTRIPL	ADCSOCAO	CLB4_OUT12	ADCDEVT3
G14	CMPSS8_CTRIPH	CMPSS8_CTRIPH_OR_CTRIPL	ADCBEVT4	EXTSYNCOUT
G15	CMPSS8_CTRIPL	ADCSOCBO	CLB4_OUT13	ADCDEVT4
G16	SD1FLT1_COMPH	SD1FLT1_COMPH_OR_COMPL	SD1FLT1_COMPZ	SD1FLT1_DRINT
G17	SD1FLT1_COMPL	INPUTXBAR7	CLB5_OUT12	CLAHALT
G18	SD1FLT2_COMPH	SD1FLT2_COMPH_OR_COMPL	SD1FLT2_COMPZ	SD1FLT2_DRINT
G19	SD1FLT2_COMPL	INPUTXBAR8	CLB5_OUT13	Reserved
G20	SD1FLT3_COMPH	SD1FLT3_COMPH_OR_COMPL	SD1FLT3_COMPZ	SD1FLT3_DRINT
G21	SD1FLT3_COMPL	INPUTXBAR9	CLB6_OUT12	Reserved
G22	SD1FLT4_COMPH	SD1FLT4_COMPH_OR_COMPL	SD1FLT4_COMPZ	SD1FLT4_DRINT
G23	SD1FLT4_COMPL	INPUTXBAR10	CLB6_OUT13	EMAC_PPS1
G24	SD2FLT1_COMPH	SD2FLT1_COMPH_OR_COMPL	SD2FLT1_COMPZ	SD2FLT1_DRINT
G25	SD2FLT1_COMPL	INPUTXBAR11	MCANA_FEVT0	CLB7_OUT12
G26	SD2FLT2_COMPH	SD2FLT2_COMPH_OR_COMPL	SD2FLT2_COMPZ	SD2FLT2_DRINT
G27	SD2FLT2_COMPL	INPUTXBAR12	MCANA_FEVT1	CLB7_OUT13
G28	SD2FLT3_COMPH	SD2FLT3_COMPH_OR_COMPL	SD2FLT3_COMPZ	SD2FLT3_DRINT
G29	SD2FLT3_COMPL	INPUTXBAR13	MCANA_FEVT2	CLB8_OUT12
G30	SD2FLT4_COMPH	SD2FLT4_COMPH_OR_COMPL	SD2FLT4_COMPZ	SD2FLT4_DRINT
G31	SD2FLT4_COMPL	INPUTXBAR14	EMAC_PPS0	CLB8_OUT13

17.2.3 GPIO Output X-BAR

The GPIO Output X-BAR takes signals from inside the device and brings them out to a GPIO. Figure 17-5 shows the architecture of the GPIO Output X-BAR. The X-BAR contains eight outputs and each contains at least one position on the GPIO mux, denoted as OUTPUTXBARx. The X-BAR allows the selection of a single signal or a logical-OR of up to 32 signals.

17.2.3.1 GPIO Output X-BAR Architecture

The GPIO Output X-BAR has eight outputs that are routed to the GPIO module. Figure 17-5 represents the architecture of a single output, but this output is identical to the architecture of all of the other outputs. Note that the architecture of the Output X-BAR (with the exception of the output latch) is similar to the architecture of the ePWM X-BAR.

First, determine the signals that can be passed to the GPIO by referencing Table 17-5 . Select up to one signal per mux (32 total muxes) for each OUTPUTXBARx output. Select the inputs to each mux using the OUTPUTxMUX0TO15CFG and OUTPUTxMUX16TO31CFG registers. To pass any signal through to the GPIO, enable the mux in the OUTPUTxMUXENABLE register. All muxes that are enabled are logically ORed before being passed on to the respective OUTPUTx signal on the GPIO module. To optionally invert the signal, use the OUTPUTINV register. The signal is only recognized on the GPIO, if the proper OUTPUTx muxing options are selected using the GpioCtrlRegs.GPxMUX and GpioCtrlRegs.GPxGMUX registers.

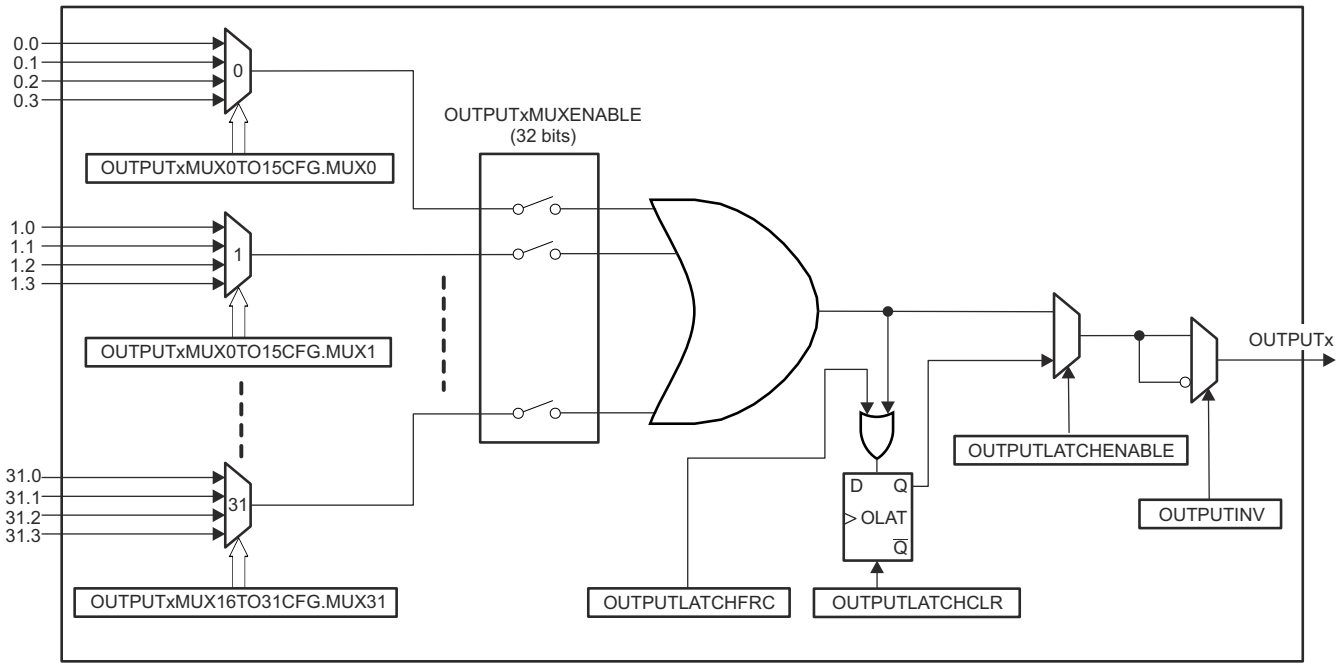


Figure 17-5. GPIO Output X-BAR Architecture

Note

Do not use "Reserved" signals in your application.

Table 17-5. Output X-BAR Mux Configuration Table

Mux	0	1	2	3
G0	CMPSS1_CTRIPOUTH	CMPSS1_CTRIPOUTH_OR_CTRIPOUTL	ADCAEVT1	ECAP1_OUT
G1	CMPSS1_CTRIPOUTL	INPUTXBAR1	CLB1_OUT12	ADCCEVT1
G2	CMPSS2_CTRIPOUTH	CMPSS2_CTRIPOUTH_OR_CTRIPOUTL	ADCAEVT2	ECAP2_OUT
G3	CMPSS2_CTRIPOUTL	INPUTXBAR2	CLB1_OUT13	ADCCEVT2
G4	CMPSS3_CTRIPOUTH	CMPSS3_CTRIPOUTH_OR_CTRIPOUTL	ADCAEVT3	ECAP3_OUT
G5	CMPSS3_CTRIPOUTL	INPUTXBAR3	CLB2_OUT12	ADCCEVT3
G6	CMPSS4_CTRIPOUTH	CMPSS4_CTRIPOUTH_OR_CTRIPOUTL	ADCAEVT4	ECAP4_OUT
G7	CMPSS4_CTRIPOUTL	INPUTXBAR4	CLB2_OUT13	ADCCEVT4
G8	CMPSS5_CTRIPOUTH	CMPSS5_CTRIPOUTH_OR_CTRIPOUTL	ADCBEVT1	ECAP5_OUT
G9	CMPSS5_CTRIPOUTL	INPUTXBAR5	CLB3_OUT12	ADCDEVT1
G10	CMPSS6_CTRIPOUTH	CMPSS6_CTRIPOUTH_OR_CTRIPOUTL	ADCBEVT2	ECAP6_OUT
G11	CMPSS6_CTRIPOUTL	INPUTXBAR6	CLB3_OUT13	ADCDEVT2
G12	CMPSS7_CTRIPOUTH	CMPSS7_CTRIPOUTH_OR_CTRIPOUTL	ADCBEVT3	ECAP7_OUT
G13	CMPSS7_CTRIPOUTL	ADCSOCAO	CLB4_OUT12	ADCDEVT3
G14	CMPSS8_CTRIPOUTH	CMPSS8_CTRIPOUTH_OR_CTRIPOUTL	ADCBEVT4	EXTSYNCOUT
G15	CMPSS8_CTRIPOUTL	ADCSOCBO	CLB4_OUT13	ADCDEVT4
G16	SD1FLT1_CEVT1	SD1FLT1_CEVT1_OR_CEVT2	Reserved	Reserved
G17	SD1FLT1_CEVT2	Reserved	CLB5_OUT12	CLAHALT
G18	SD1FLT2_CEVT1	SD1FLT2_CEVT1_OR_CEVT2	Reserved	Reserved
G19	SD1FLT2_CEVT2	Reserved	CLB5_OUT13	Reserved
G20	SD1FLT3_CEVT1	SD1FLT3_CEVT1_OR_CEVT2	Reserved	Reserved
G21	SD1FLT3_CEVT2	Reserved	CLB6_OUT12	FSIRXA_TRIG2
G22	SD1FLT4_CEVT1	SD1FLT4_CEVT1_OR_CEVT2	Reserved	Reserved
G23	SD1FLT4_CEVT2	Reserved	CLB6_OUT13	Reserved
G24	SD2FLT1_CEVT1	SD2FLT1_CEVT1_OR_CEVT2	Reserved	Reserved
G25	SD2FLT1_CEVT2	Reserved	Reserved	CLB7_OUT12
G26	SD2FLT2_CEVT1	SD2FLT2_CEVT1_OR_CEVT2	Reserved	Reserved
G27	SD2FLT2_CEVT2	Reserved	ERRORSTS	CLB7_OUT13
G28	SD2FLT3_CEVT1	SD2FLT3_CEVT1_OR_CEVT2	XCLKOUT	Reserved
G29	SD2FLT3_CEVT2	Reserved	Reserved	CLB8_OUT12
G30	SD2FLT4_CEVT1	SD2FLT4_CEVT1_OR_CEVT2	Reserved	Reserved
G31	SD2FLT4_CEVT2	Reserved	Reserved	CLB8_OUT13

17.2.4 CLB Output X-BAR

The CLB Output X-BAR takes signals from inside the CLB Tiles and brings them out to a GPIO.

17.2.4.1 CLB Output X-BAR Architecture

The CLB Output X-BAR has eight outputs that are routed to the GPIO module. CLB Output X-BAR architecture is identical to the architecture of the GPIO Output X-BAR. First, determine the signals that can be passed to the GPIO by referencing Table 17-6.

Note

Do not use "Reserved" signals in your application.

Table 17-6. CLB Output X-BAR Mux Configuration Table

Mux	0	1	2	3
G0	CLB1_OUT0	CLB5_OUT0	Reserved	Reserved
G1	CLB1_OUT1	CLB5_OUT1	Reserved	Reserved
G2	CLB1_OUT2	CLB5_OUT2	Reserved	Reserved
G3	CLB1_OUT3	CLB5_OUT3	Reserved	Reserved
G4	CLB1_OUT4	CLB5_OUT4	Reserved	Reserved
G5	CLB1_OUT5	CLB5_OUT5	Reserved	Reserved
G6	CLB1_OUT6	CLB5_OUT6	Reserved	Reserved
G7	CLB1_OUT7	CLB5_OUT7	Reserved	Reserved
G8	CLB2_OUT0	CLB6_OUT0	Reserved	Reserved
G9	CLB2_OUT1	CLB6_OUT1	Reserved	Reserved
G10	CLB2_OUT2	CLB6_OUT2	Reserved	Reserved
G11	CLB2_OUT3	CLB6_OUT3	Reserved	Reserved
G12	CLB2_OUT4	CLB6_OUT4	Reserved	Reserved
G13	CLB2_OUT5	CLB6_OUT5	Reserved	Reserved
G14	CLB2_OUT6	CLB6_OUT6	Reserved	Reserved
G15	CLB2_OUT7	CLB6_OUT7	Reserved	Reserved
G16	CLB3_OUT0	CLB7_OUT0	Reserved	Reserved
G17	CLB3_OUT1	CLB7_OUT1	Reserved	Reserved
G18	CLB3_OUT2	CLB7_OUT2	Reserved	Reserved
G19	CLB3_OUT3	CLB7_OUT3	Reserved	Reserved
G20	CLB3_OUT4	CLB7_OUT4	Reserved	Reserved
G21	CLB3_OUT5	CLB7_OUT5	Reserved	Reserved
G22	CLB3_OUT6	CLB7_OUT6	Reserved	Reserved
G23	CLB3_OUT7	CLB7_OUT7	Reserved	Reserved
G24	CLB4_OUT0	CLB8_OUT0	Reserved	Reserved
G25	CLB4_OUT1	CLB8_OUT1	Reserved	Reserved
G26	CLB4_OUT2	CLB8_OUT2	Reserved	Reserved
G27	CLB4_OUT3	CLB8_OUT3	Reserved	Reserved
G28	CLB4_OUT4	CLB8_OUT4	Reserved	Reserved
G29	CLB4_OUT5	CLB8_OUT5	Reserved	Reserved

Table 17-6. CLB Output X-BAR Mux Configuration Table (continued)

Mux	0	1	2	3
G30	CLB4_OUT6	CLB8_OUT6	Reserved	Reserved
G31	CLB4_OUT7	CLB8_OUT7	Reserved	Reserved

17.2.5 X-BAR Flags

With the exception of the CMPSS signals, the ePWM X-BAR and the Output X-BAR have all of the same input signals. Due to the inputs being similar between the ePWM X-BAR, CLB X-BAR, and Output X-BAR, all X-BAR modules leverage a single set of input flags to indicate which input signals have been triggered. This allows software to check the input flags when an event occurs. See Figure 17-6 for more information. There is a bit allocated for each input signal in one of the XBARFLGx registers. The flag remains set until cleared through the appropriate XBARCLRx register.

Note

Not all input sources are routed to all X-BAR modules. Refer to the X-BAR specific configuration tables for exact connections.

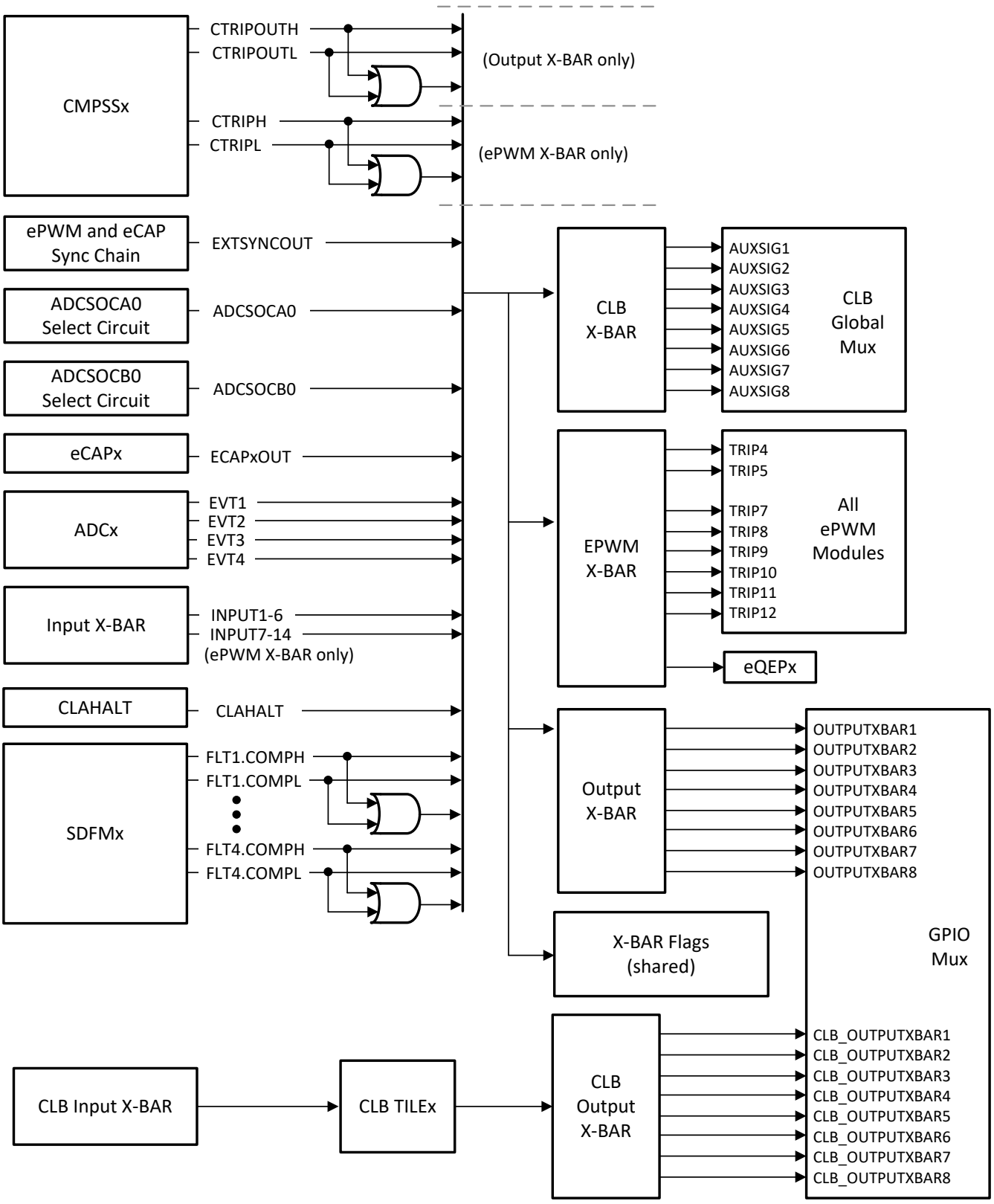


Figure 17-6. X-BAR Input Sources

17.3 XBAR Registers

This section describes the Crossbar registers.

17.3.1 XBAR Base Address Table (C28)

Table 17-7. XBAR Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
EPwmXbarRegs	EPWM_XBAR_REGS	EPWMXBAR_BASE	0x0000_7A00	YES	-	-	-	YES
CLBXbarRegs	CLB_XBAR_REGS	CLBXBAR_BASE	0x0000_7A40	YES	-	-	-	YES
InputXbarRegs	INPUT_XBAR_REGS	INPUTXBAR_BASE	0x0000_7900	YES	-	-	-	YES
XbarRegs	XBAR_REGS	XBAR_BASE	0x0000_7920	YES	-	-	-	YES
ClbInputXbarRegs	INPUT_XBAR_REGS	CLBINPUTXBAR_BASE	0x0000_7960	YES	-	-	-	YES
OutputXbarRegs	OUTPUT_XBAR_REGS	OUTPUTXBAR_BASE	0x0000_7A80	YES	-	-	-	YES
ClbOutputXbarRegs	OUTPUT_XBAR_REGS	CLBOUTPUTXBAR_BASE	0x0000_7BC0	YES	-	-	-	YES

17.3.2 INPUT_XBAR_REGS Registers

Table 17-8 lists the memory-mapped registers for the INPUT_XBAR_REGS registers. All register offset addresses not listed in Table 17-8 should be considered as reserved locations and the register contents should not be modified.

Table 17-8. INPUT_XBAR_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	INPUT1SELECT	INPUT1 Input Select Register (GPIO0 to x)	EALLOW	Go
1h	INPUT2SELECT	INPUT2 Input Select Register (GPIO0 to x)	EALLOW	Go
2h	INPUT3SELECT	INPUT3 Input Select Register (GPIO0 to x)	EALLOW	Go
3h	INPUT4SELECT	INPUT4 Input Select Register (GPIO0 to x)	EALLOW	Go
4h	INPUT5SELECT	INPUT5 Input Select Register (GPIO0 to x)	EALLOW	Go
5h	INPUT6SELECT	INPUT6 Input Select Register (GPIO0 to x)	EALLOW	Go
6h	INPUT7SELECT	INPUT7 Input Select Register (GPIO0 to x)	EALLOW	Go
7h	INPUT8SELECT	INPUT8 Input Select Register (GPIO0 to x)	EALLOW	Go
8h	INPUT9SELECT	INPUT9 Input Select Register (GPIO0 to x)	EALLOW	Go
9h	INPUT10SELECT	INPUT10 Input Select Register (GPIO0 to x)	EALLOW	Go
Ah	INPUT11SELECT	INPUT11 Input Select Register (GPIO0 to x)	EALLOW	Go
Bh	INPUT12SELECT	INPUT12 Input Select Register (GPIO0 to x)	EALLOW	Go
Ch	INPUT13SELECT	INPUT13 Input Select Register (GPIO0 to x)	EALLOW	Go
Dh	INPUT14SELECT	INPUT14 Input Select Register (GPIO0 to x)	EALLOW	Go
Eh	INPUT15SELECT	INPUT15 Input Select Register (GPIO0 to x)	EALLOW	Go
Fh	INPUT16SELECT	INPUT16 Input Select Register (GPIO0 to x)	EALLOW	Go
1Eh	INPUTSELECTLOCK	Input Select Lock Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 17-9 shows the codes that are used for access types in this section.

Table 17-9. INPUT_XBAR_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 17-9. INPUT_XBAR_REGS Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

17.3.2.1 INPUT1SELECT Register (Offset = 0h) [Reset = FFFEh]

INPUT1SELECT is shown in Figure 17-7 and described in Table 17-10.

Return to the Summary Table.

INPUT1 Input Select Register (GPIO0 to x)

Figure 17-7. INPUT1SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 17-10. INPUT1SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT1 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

17.3.2.2 INPUT2SELECT Register (Offset = 1h) [Reset = FFFEh]

INPUT2SELECT is shown in Figure 17-8 and described in Table 17-11.

Return to the Summary Table.

INPUT2 Input Select Register (GPIO0 to x)

Figure 17-8. INPUT2SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 17-11. INPUT2SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT2 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

17.3.2.3 INPUT3SELECT Register (Offset = 2h) [Reset = FFFEh]

INPUT3SELECT is shown in Figure 17-9 and described in Table 17-12.

Return to the Summary Table.

INPUT3 Input Select Register (GPIO0 to x)

Figure 17-9. INPUT3SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 17-12. INPUT3SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT3 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

17.3.2.4 INPUT4SELECT Register (Offset = 3h) [Reset = FFFEh]

INPUT4SELECT is shown in Figure 17-10 and described in Table 17-13.

Return to the Summary Table.

INPUT4 Input Select Register (GPIO0 to x)

Figure 17-10. INPUT4SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 17-13. INPUT4SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT4 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

17.3.2.5 INPUT5SELECT Register (Offset = 4h) [Reset = FFFEh]

INPUT5SELECT is shown in Figure 17-11 and described in Table 17-14.

Return to the Summary Table.

INPUT5 Input Select Register (GPIO0 to x)

Figure 17-11. INPUT5SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 17-14. INPUT5SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT5 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

17.3.2.6 INPUT6SELECT Register (Offset = 5h) [Reset = FFFEh]

INPUT6SELECT is shown in Figure 17-12 and described in Table 17-15.

Return to the Summary Table.

INPUT6 Input Select Register (GPIO0 to x)

Figure 17-12. INPUT6SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 17-15. INPUT6SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT6 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

17.3.2.7 INPUT7SELECT Register (Offset = 6h) [Reset = FFFEh]

INPUT7SELECT is shown in Figure 17-13 and described in Table 17-16.

Return to the Summary Table.

INPUT7 Input Select Register (GPIO0 to x)

Figure 17-13. INPUT7SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 17-16. INPUT7SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT7 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

17.3.2.8 INPUT8SELECT Register (Offset = 7h) [Reset = FFFEh]

INPUT8SELECT is shown in Figure 17-14 and described in Table 17-17.

Return to the Summary Table.

INPUT8 Input Select Register (GPIO0 to x)

Figure 17-14. INPUT8SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 17-17. INPUT8SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT8 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

17.3.2.9 INPUT9SELECT Register (Offset = 8h) [Reset = FFFEh]

INPUT9SELECT is shown in Figure 17-15 and described in Table 17-18.

Return to the Summary Table.

INPUT9 Input Select Register (GPIO0 to x)

Figure 17-15. INPUT9SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 17-18. INPUT9SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT9 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

17.3.2.10 INPUT10SELECT Register (Offset = 9h) [Reset = FFFEh]

INPUT10SELECT is shown in Figure 17-16 and described in Table 17-19.

Return to the Summary Table.

INPUT10 Input Select Register (GPIO0 to x)

Figure 17-16. INPUT10SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 17-19. INPUT10SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT10 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

17.3.2.11 INPUT11SELECT Register (Offset = Ah) [Reset = FFFEh]

INPUT11SELECT is shown in Figure 17-17 and described in Table 17-20.

Return to the Summary Table.

INPUT11 Input Select Register (GPIO0 to x)

Figure 17-17. INPUT11SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 17-20. INPUT11SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT11 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

17.3.2.12 INPUT12SELECT Register (Offset = Bh) [Reset = FFFEh]

INPUT12SELECT is shown in Figure 17-18 and described in Table 17-21.

Return to the Summary Table.

INPUT12 Input Select Register (GPIO0 to x)

Figure 17-18. INPUT12SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 17-21. INPUT12SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT12 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

17.3.2.13 INPUT13SELECT Register (Offset = Ch) [Reset = FFFEh]

INPUT13SELECT is shown in Figure 17-19 and described in Table 17-22.

Return to the Summary Table.

INPUT13 Input Select Register (GPIO0 to x)

Figure 17-19. INPUT13SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 17-22. INPUT13SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT13 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

17.3.2.14 INPUT14SELECT Register (Offset = Dh) [Reset = FFFEh]

INPUT14SELECT is shown in Figure 17-20 and described in Table 17-23.

Return to the Summary Table.

INPUT14 Input Select Register (GPIO0 to x)

Figure 17-20. INPUT14SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 17-23. INPUT14SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT14 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

17.3.2.15 INPUT15SELECT Register (Offset = Eh) [Reset = FFFEh]

INPUT15SELECT is shown in Figure 17-21 and described in Table 17-24.

Return to the Summary Table.

INPUT15 Input Select Register (GPIO0 to x)

Figure 17-21. INPUT15SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 17-24. INPUT15SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT15 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

17.3.2.16 INPUT16SELECT Register (Offset = Fh) [Reset = FFFEh]

INPUT16SELECT is shown in Figure 17-22 and described in Table 17-25.

Return to the Summary Table.

INPUT16 Input Select Register (GPIO0 to x)

Figure 17-22. INPUT16SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 17-25. INPUT16SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT16 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

17.3.2.17 INPUTSELECTLOCK Register (Offset = 1Eh) [Reset = 0h]

INPUTSELECTLOCK is shown in Figure 17-23 and described in Table 17-26.

Return to the Summary Table.

Input Select Lock Register.
Any bit in this register, once set can only be cleared through SYSRSn. Write of 0 to any bit of this register has no effect. Reads to the registers which have LOCK protection are always allowed.

Figure 17-23. INPUTSELECTLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
INPUT16SELECT	INPUT15SELECT	INPUT14SELECT	INPUT13SELECT	INPUT12SELECT	INPUT11SELECT	INPUT10SELECT	INPUT9SELECT
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
7	6	5	4	3	2	1	0
INPUT8SELECT	INPUT7SELECT	INPUT6SELECT	INPUT5SELECT	INPUT4SELECT	INPUT3SELECT	INPUT2SELECT	INPUT1SELECT
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 17-26. INPUTSELECTLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15	INPUT16SELECT	R/WSonce	0h	Lock bit for INPUT16SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
14	INPUT15SELECT	R/WSonce	0h	Lock bit for INPUT15SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
13	INPUT14SELECT	R/WSonce	0h	Lock bit for INPUT14SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
12	INPUT13SELECT	R/WSonce	0h	Lock bit for INPUT13SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
11	INPUT12SELECT	R/WSonce	0h	Lock bit for INPUT12SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
10	INPUT11SELECT	R/WSonce	0h	Lock bit for INPUT11SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
9	INPUT10SELECT	R/WSonce	0h	Lock bit for INPUT10SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn

Table 17-26. INPUTSELECTLOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	INPUT9SELECT	R/WSonce	0h	Lock bit for INPUT9SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
7	INPUT8SELECT	R/WSonce	0h	Lock bit for INPUT8SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
6	INPUT7SELECT	R/WSonce	0h	Lock bit for INPUT7SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
5	INPUT6SELECT	R/WSonce	0h	Lock bit for INPUT6SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
4	INPUT5SELECT	R/WSonce	0h	Lock bit for INPUT5SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
3	INPUT4SELECT	R/WSonce	0h	Lock bit for INPUT4SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
2	INPUT3SELECT	R/WSonce	0h	Lock bit for INPUT3SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
1	INPUT2SELECT	R/WSonce	0h	Lock bit for INPUT2SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
0	INPUT1SELECT	R/WSonce	0h	Lock bit for INPUT1SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn

17.3.3 XBAR_REGS Registers

Table 17-27 lists the memory-mapped registers for the XBAR_REGS registers. All register offset addresses not listed in Table 17-27 should be considered as reserved locations and the register contents should not be modified.

Table 17-27. XBAR_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	XBARFLG1	X-Bar Input Flag Register 1		Go
2h	XBARFLG2	X-Bar Input Flag Register 2		Go
4h	XBARFLG3	X-Bar Input Flag Register 3		Go
6h	XBARFLG4	X-Bar Input Flag Register 4		Go
8h	XBARCLR1	X-Bar Input Flag Clear Register 1		Go
Ah	XBARCLR2	X-Bar Input Flag Clear Register 2		Go
Ch	XBARCLR3	X-Bar Input Flag Clear Register 3		Go
Eh	XBARCLR4	X-Bar Input Flag Clear Register 4		Go

Complex bit access types are encoded to fit into small table cells. Table 17-28 shows the codes that are used for access types in this section.

Table 17-28. XBAR_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

17.3.3.1 XBARFLG1 Register (Offset = 0h) [Reset = 0h]

XBARFLG1 is shown in Figure 17-24 and described in Table 17-29.

Return to the Summary Table.

This register is used to flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered.
1: Corresponding Input was triggered
0: Corresponding Input was not triggered

Figure 17-24. XBARFLG1 Register

31	30	29	28	27	26	25	24
CMPSS8_CTRIPOUTH	CMPSS8_CTRIPOUTL	CMPSS7_CTRIPOUTH	CMPSS7_CTRIPOUTL	CMPSS6_CTRIPOUTH	CMPSS6_CTRIPOUTL	CMPSS5_CTRIPOUTH	CMPSS5_CTRIPOUTL
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
CMPSS4_CTRIPOUTH	CMPSS4_CTRIPOUTL	CMPSS3_CTRIPOUTH	CMPSS3_CTRIPOUTL	CMPSS2_CTRIPOUTH	CMPSS2_CTRIPOUTL	CMPSS1_CTRIPOUTH	CMPSS1_CTRIPOUTL
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
CMPSS8_CTRIPH	CMPSS8_CTRIPL	CMPSS7_CTRIPH	CMPSS7_CTRIPL	CMPSS6_CTRIPH	CMPSS6_CTRIPL	CMPSS5_CTRIPH	CMPSS5_CTRIPL
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
CMPSS4_CTRIPH	CMPSS4_CTRIPL	CMPSS3_CTRIPH	CMPSS3_CTRIPL	CMPSS2_CTRIPH	CMPSS2_CTRIPL	CMPSS1_CTRIPH	CMPSS1_CTRIPL
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 17-29. XBARFLG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CMPSS8_CTRIPOUTH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS8_CTRIPOUTH input was triggered 0: CMPSS8_CTRIPOUTH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
30	CMPSS8_CTRIPOUTL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS8_CTRIPOUTL input was triggered 0: CMPSS8_CTRIPOUTL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
29	CMPSS7_CTRIPOUTH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS7_CTRIPOUTH input was triggered 0: CMPSS7_CTRIPOUTH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
28	CMPSS7_CTRIPOUTL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS7_CTRIPOUTL input was triggered 0: CMPSS7_CTRIPOUTL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 17-29. XBARFLG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	CMPSS6_CTRIPOUTH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS6_CTRIPOUTH input was triggered 0: CMPSS6_CTRIPOUTH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
26	CMPSS6_CTRIPOUTL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS6_CTRIPOUTL input was triggered 0: CMPSS6_CTRIPOUTL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
25	CMPSS5_CTRIPOUTH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS5_CTRIPOUTH input was triggered 0: CMPSS5_CTRIPOUTH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
24	CMPSS5_CTRIPOUTL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS5_CTRIPOUTL input was triggered 0: CMPSS5_CTRIPOUTL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
23	CMPSS4_CTRIPOUTH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS4_CTRIPOUTH input was triggered 0: CMPSS4_CTRIPOUTH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
22	CMPSS4_CTRIPOUTL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS4_CTRIPOUTL input was triggered 0: CMPSS4_CTRIPOUTL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
21	CMPSS3_CTRIPOUTH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS3_CTRIPOUTH input was triggered 0: CMPSS3_CTRIPOUTH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
20	CMPSS3_CTRIPOUTL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS3_CTRIPOUTL input was triggered 0: CMPSS3_CTRIPOUTL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 17-29. XBARFLG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	CMPSS2_CTRIPOUTH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS2_CTRIPOUTH input was triggered 0: CMPSS2_CTRIPOUTH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
18	CMPSS2_CTRIPOUTL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS2_CTRIPOUTL input was triggered 0: CMPSS2_CTRIPOUTL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
17	CMPSS1_CTRIPOUTH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS1_CTRIPOUTH input was triggered 0: CMPSS1_CTRIPOUTH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
16	CMPSS1_CTRIPOUTL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS1_CTRIPOUTL input was triggered 0: CMPSS1_CTRIPOUTL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
15	CMPSS8_CTRIPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS8_CTRIPH input was triggered 0: CMPSS8_CTRIPH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
14	CMPSS8_CTRIPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS8_CTRIPL input was triggered 0: CMPSS8_CTRIPL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
13	CMPSS7_CTRIPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS7_CTRIPH input was triggered 0: CMPSS7_CTRIPH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
12	CMPSS7_CTRIPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS7_CTRIPL input was triggered 0: CMPSS7_CTRIPL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 17-29. XBARFLG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	CMPSS6_CTRIPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS6_CTRIPH input was triggered 0: CMPSS6_CTRIPH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
10	CMPSS6_CTRIPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS6_CTRIPL input was triggered 0: CMPSS6_CTRIPL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
9	CMPSS5_CTRIPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS5_CTRIPH input was triggered 0: CMPSS5_CTRIPH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
8	CMPSS5_CTRIPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS5_CTRIPL input was triggered 0: CMPSS5_CTRIPL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
7	CMPSS4_CTRIPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS4_CTRIPH input was triggered 0: CMPSS4_CTRIPH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
6	CMPSS4_CTRIPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS4_CTRIPL input was triggered 0: CMPSS4_CTRIPL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
5	CMPSS3_CTRIPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS3_CTRIPH input was triggered 0: CMPSS3_CTRIPH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
4	CMPSS3_CTRIPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS3_CTRIPL input was triggered 0: CMPSS3_CTRIPL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 17-29. XBARFLG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	CMPSS2_CTRIPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS2_CTRIPH input was triggered 0: CMPSS2_CTRIPH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
2	CMPSS2_CTRIPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS2_CTRIPL input was triggered 0: CMPSS2_CTRIPL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
1	CMPSS1_CTRIPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS1_CTRIPH input was triggered 0: CMPSS1_CTRIPH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
0	CMPSS1_CTRIPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CMPSS1_CTRIPL input was triggered 0: CMPSS1_CTRIPL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

17.3.3.2 XBARFLG2 Register (Offset = 2h) [Reset = 0h]

XBARFLG2 is shown in Figure 17-25 and described in Table 17-30.

Return to the Summary Table.

This register is used to flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered.
1: Corresponding Input was triggered
0: Corresponding Input was not triggered

Figure 17-25. XBARFLG2 Register

31	30	29	28	27	26	25	24
ADCCEVT1	ADCBEVT4	ADCBEVT3	ADCBEVT2	ADCBEVT1	ADCAEVT4	ADCAEVT3	ADCAEVT2
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
ADCAEVT1	EXTSYNCOUT	ECAP6_OUT	ECAP5_OUT	ECAP4_OUT	ECAP3_OUT	ECAP2_OUT	ECAP1_OUT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
INPUT14	INPUT13	INPUT12	INPUT11	INPUT10	INPUT9	INPUT8	INPUT7
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
ADCSOCB	ADCSOCA	INPUT6	INPUT5	INPUT4	INPUT3	INPUT2	INPUT1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 17-30. XBARFLG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ADCCEVT1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ADCCEVT1 input was triggered 0: ADCCEVT1 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
30	ADCBEVT4	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ADCBEVT4 input was triggered 0: ADCBEVT4 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
29	ADCBEVT3	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ADCBEVT3 input was triggered 0: ADCBEVT3 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
28	ADCBEVT2	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ADCBEVT2 input was triggered 0: ADCBEVT2 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 17-30. XBARFLG2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	ADCBEVT1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ADCBEVT1 input was triggered 0: ADCBEVT1 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
26	ADCAEVT4	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ADCAEVT4 input was triggered 0: ADCAEVT4 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
25	ADCAEVT3	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ADCAEVT3 input was triggered 0: ADCAEVT3 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
24	ADCAEVT2	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ADCAEVT2 input was triggered 0: ADCAEVT2 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
23	ADCAEVT1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ADCAEVT1 input was triggered 0: ADCAEVT1 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
22	EXTSYNCOUT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: EXTSYNCOUT input was triggered 0: EXTSYNCOUT Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
21	ECAP6_OUT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ECAP6_OUT input was triggered 0: ECAP6_OUT Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
20	ECAP5_OUT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ECAP5_OUT input was triggered 0: ECAP5_OUT Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 17-30. XBARFLG2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	ECAP4_OUT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ECAP4_OUT input was triggered 0: ECAP4_OUT Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
18	ECAP3_OUT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ECAP3_OUT input was triggered 0: ECAP3_OUT Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
17	ECAP2_OUT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ECAP2_OUT input was triggered 0: ECAP2_OUT Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
16	ECAP1_OUT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ECAP1_OUT input was triggered 0: ECAP1_OUT Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
15	INPUT14	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: INPUT14 input was triggered 0: INPUT14 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
14	INPUT13	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: INPUT13 input was triggered 0: INPUT13 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
13	INPUT12	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: INPUT12 input was triggered 0: INPUT12 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
12	INPUT11	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: INPUT11 input was triggered 0: INPUT11 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 17-30. XBARFLG2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	INPUT10	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: INPUT10 input was triggered 0: INPUT10 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
10	INPUT9	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: INPUT9 input was triggered 0: INPUT9 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
9	INPUT8	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: INPUT8 input was triggered 0: INPUT8 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
8	INPUT7	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: INPUT7 input was triggered 0: INPUT7 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
7	ADCSOCB	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ADCSOCB input was triggered 0: ADCSOCB Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
6	ADCSOCA	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ADCSOCA input was triggered 0: ADCSOCA Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
5	INPUT6	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: INPUT6 input was triggered 0: INPUT6 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
4	INPUT5	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: INPUT5 input was triggered 0: INPUT5 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 17-30. XBARFLG2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	INPUT4	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: INPUT4 input was triggered 0: INPUT4 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
2	INPUT3	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: INPUT3 input was triggered 0: INPUT3 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
1	INPUT2	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: INPUT2 input was triggered 0: INPUT2 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
0	INPUT1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: INPUT1 input was triggered 0: INPUT1 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

17.3.3.3 XBARFLG3 Register (Offset = 4h) [Reset = 0h]

XBARFLG3 is shown in Figure 17-26 and described in Table 17-31.

Return to the Summary Table.

This register is used to flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered.
1: Corresponding Input was triggered
0: Corresponding Input was not triggered

Figure 17-26. XBARFLG3 Register

31	30	29	28	27	26	25	24
SD1FLT4_DRINT	SD1FLT4_COMPZ	SD1FLT3_DRINT	SD1FLT3_COMPZ	SD1FLT2_DRINT	SD1FLT2_COMPZ	SD1FLT1_DRINT	SD1FLT1_COMPZ
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
ECAP7_OUT	SD2FLT4_COMPH	SD2FLT4_COMPL	SD2FLT3_COMPH	SD2FLT3_COMPL	SD2FLT2_COMPH	SD2FLT2_COMPL	SD2FLT1_COMPH
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
SD2FLT1_COMPL	SD1FLT4_COMPH	SD1FLT4_COMPL	SD1FLT3_COMPH	SD1FLT3_COMPL	SD1FLT2_COMPH	SD1FLT2_COMPL	SD1FLT1_COMPH
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
SD1FLT1_COMPL	ADCDEVT4	ADCDEVT3	ADCDEVT2	ADCDEVT1	ADCCEVT4	ADCCEVT3	ADCCEVT2
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 17-31. XBARFLG3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SD1FLT4_DRINT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD1FLT4_DRINT input was triggered 0: SD1FLT4_DRINT Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
30	SD1FLT4_COMPZ	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD1FLT4_COMPZ input was triggered 0: SD1FLT4_COMPZ Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
29	SD1FLT3_DRINT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD1FLT3_DRINT input was triggered 0: SD1FLT3_DRINT Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
28	SD1FLT3_COMPZ	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD1FLT3_COMPZ input was triggered 0: SD1FLT3_COMPZ Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 17-31. XBARFLG3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	SD1FLT2_DRINT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD1FLT2_DRINT input was triggered 0: SD1FLT2_DRINT Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
26	SD1FLT2_COMPZ	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD1FLT2_COMPZ input was triggered 0: SD1FLT2_COMPZ Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
25	SD1FLT1_DRINT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD1FLT1_DRINT input was triggered 0: SD1FLT1_DRINT Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
24	SD1FLT1_COMPZ	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD1FLT1_COMPZ input was triggered 0: SD1FLT1_COMPZ Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
23	ECAP7_OUT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ECAP7_OUT input was triggered 0: ECAP7_OUT Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
22	SD2FLT4_COMPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD2FLT4_COMPH input was triggered 0: SD2FLT4_COMPH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
21	SD2FLT4_COMPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD2FLT4_COMPL input was triggered 0: SD2FLT4_COMPL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
20	SD2FLT3_COMPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD2FLT3_COMPH input was triggered 0: SD2FLT3_COMPH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 17-31. XBARFLG3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	SD2FLT3_COMPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD2FLT3_COMPL input was triggered 0: SD2FLT3_COMPL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
18	SD2FLT2_COMPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD2FLT2_COMPH input was triggered 0: SD2FLT2_COMPH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
17	SD2FLT2_COMPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD2FLT2_COMPL input was triggered 0: SD2FLT2_COMPL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
16	SD2FLT1_COMPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD2FLT1_COMPH input was triggered 0: SD2FLT1_COMPH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
15	SD2FLT1_COMPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD2FLT1_COMPL input was triggered 0: SD2FLT1_COMPL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
14	SD1FLT4_COMPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD1FLT4_COMPH input was triggered 0: SD1FLT4_COMPH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
13	SD1FLT4_COMPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD1FLT4_COMPL input was triggered 0: SD1FLT4_COMPL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
12	SD1FLT3_COMPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD1FLT3_COMPH input was triggered 0: SD1FLT3_COMPH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 17-31. XBARFLG3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	SD1FLT3_COMPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD1FLT3_COMPL input was triggered 0: SD1FLT3_COMPL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
10	SD1FLT2_COMPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD1FLT2_COMPH input was triggered 0: SD1FLT2_COMPH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
9	SD1FLT2_COMPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD1FLT2_COMPL input was triggered 0: SD1FLT2_COMPL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
8	SD1FLT1_COMPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD1FLT1_COMPH input was triggered 0: SD1FLT1_COMPH Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
7	SD1FLT1_COMPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD1FLT1_COMPL input was triggered 0: SD1FLT1_COMPL Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
6	ADCDEVT4	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ADCDEVT4 input was triggered 0: ADCDEVT4 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
5	ADCDEVT3	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ADCDEVT3 input was triggered 0: ADCDEVT3 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
4	ADCDEVT2	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ADCDEVT2 input was triggered 0: ADCDEVT2 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 17-31. XBARFLG3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	ADCDEVT1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ADCDEVT1 input was triggered 0: ADCDEVT1 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
2	ADCCEVT4	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ADCCEVT4 input was triggered 0: ADCCEVT4 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
1	ADCCEVT3	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ADCCEVT3 input was triggered 0: ADCCEVT3 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
0	ADCCEVT2	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ADCCEVT2 input was triggered 0: ADCCEVT2 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

17.3.3.4 XBARFLG4 Register (Offset = 6h) [Reset = 0h]

XBARFLG4 is shown in Figure 17-27 and described in Table 17-32.

Return to the Summary Table.

This register is used to flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered.
1: Corresponding Input was triggered
0: Corresponding Input was not triggered

Figure 17-27. XBARFLG4 Register

31	30	29	28	27	26	25	24
CLAHALT	ECATSYNC1	ECATSYNC0	ERRORSTS_ERROR	CLB6_OUT5	CLB6_OUT4	CLB5_OUT5	CLB5_OUT4
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
CLB4_OUT5	CLB4_OUT4	CLB3_OUT5	CLB3_OUT4	CLB2_OUT5	CLB2_OUT4	CLB1_OUT5	CLB1_OUT4
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
CLB8_OUT5	CLB8_OUT4	CLB7_OUT5	CLB7_OUT4	MCANA_FEVT2	MCANA_FEVT1	MCANA_FEVT0	EMAC_PPS0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
SD2FLT4_DRINT	SD2FLT4_COMPZ	SD2FLT3_DRINT	SD2FLT3_COMPZ	SD2FLT2_DRINT	SD2FLT2_COMPZ	SD2FLT1_DRINT	SD2FLT1_COMPZ
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 17-32. XBARFLG4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CLAHALT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CLAHALT input was triggered 0: CLAHALT Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
30	ECATSYNC1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ECATSYNC1 input was triggered 0: ECATSYNC1 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
29	ECATSYNC0	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ECATSYNC0 input was triggered 0: ECATSYNC0 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
28	ERRORSTS_ERROR	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ERRORSTS_ERROR input was triggered 0: ERRORSTS_ERROR Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 17-32. XBARFLG4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	CLB6_OUT5	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CLB6_OUT5 input was triggered 0: CLB6_OUT5 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
26	CLB6_OUT4	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CLB6_OUT4 input was triggered 0: CLB6_OUT4 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
25	CLB5_OUT5	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CLB5_OUT5 input was triggered 0: CLB5_OUT5 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
24	CLB5_OUT4	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CLB5_OUT4 input was triggered 0: CLB5_OUT4 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
23	CLB4_OUT5	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CLB4_OUT5 input was triggered 0: CLB4_OUT5 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
22	CLB4_OUT4	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CLB4_OUT4 input was triggered 0: CLB4_OUT4 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
21	CLB3_OUT5	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CLB3_OUT5 input was triggered 0: CLB3_OUT5 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
20	CLB3_OUT4	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CLB3_OUT4 input was triggered 0: CLB3_OUT4 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 17-32. XBARFLG4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	CLB2_OUT5	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CLB2_OUT5 input was triggered 0: CLB2_OUT5 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
18	CLB2_OUT4	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CLB2_OUT4 input was triggered 0: CLB2_OUT4 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
17	CLB1_OUT5	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CLB1_OUT5 input was triggered 0: CLB1_OUT5 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
16	CLB1_OUT4	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CLB1_OUT4 input was triggered 0: CLB1_OUT4 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
15	CLB8_OUT5	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CLB8_OUT5 input was triggered 0: CLB8_OUT5 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
14	CLB8_OUT4	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CLB8_OUT4 input was triggered 0: CLB8_OUT4 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
13	CLB7_OUT5	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CLB7_OUT5 input was triggered 0: CLB7_OUT5 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
12	CLB7_OUT4	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: CLB7_OUT4 input was triggered 0: CLB7_OUT4 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 17-32. XBARFLG4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	MCANA_FEVT2	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: MCANA_FEVT2 input was triggered 0: MCANA_FEVT2 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
10	MCANA_FEVT1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: MCANA_FEVT1 input was triggered 0: MCANA_FEVT1 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
9	MCANA_FEVT0	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: MCANA_FEVT0 input was triggered 0: MCANA_FEVT0 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
8	EMAC_PPS0	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: EMAC_PPS0 input was triggered 0: EMAC_PPS0 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
7	SD2FLT4_DRINT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD2FLT4_DRINT input was triggered 0: SD2FLT4_DRINT Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
6	SD2FLT4_COMPZ	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD2FLT4_COMPZ input was triggered 0: SD2FLT4_COMPZ Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
5	SD2FLT3_DRINT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD2FLT3_DRINT input was triggered 0: SD2FLT3_DRINT Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
4	SD2FLT3_COMPZ	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD2FLT3_COMPZ input was triggered 0: SD2FLT3_COMPZ Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 17-32. XBARFLG4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	SD2FLT2_DRINT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD2FLT2_DRINT input was triggered 0: SD2FLT2_DRINT Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
2	SD2FLT2_COMPZ	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD2FLT2_COMPZ input was triggered 0: SD2FLT2_COMPZ Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
1	SD2FLT1_DRINT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD2FLT1_DRINT input was triggered 0: SD2FLT1_DRINT Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
0	SD2FLT1_COMPZ	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: SD2FLT1_COMPZ input was triggered 0: SD2FLT1_COMPZ Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

17.3.3.5 XBARCLR1 Register (Offset = 8h) [Reset = 0h]

XBARCLR1 is shown in Figure 17-28 and described in Table 17-33.

Return to the Summary Table.

This register is used to clear the flag(s) in the XBARFLG1 register.
1: Clears the corresponding bit in the XBARFLG1 register.
0: Writing 0 has no effect

Figure 17-28. XBARCLR1 Register

31	30	29	28	27	26	25	24
CMPSS8_CTRIPOUTH	CMPSS8_CTRIPOUTL	CMPSS7_CTRIPOUTH	CMPSS7_CTRIPOUTL	CMPSS6_CTRIPOUTH	CMPSS6_CTRIPOUTL	CMPSS5_CTRIPOUTH	CMPSS5_CTRIPOUTL
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
CMPSS4_CTRIPOUTH	CMPSS4_CTRIPOUTL	CMPSS3_CTRIPOUTH	CMPSS3_CTRIPOUTL	CMPSS2_CTRIPOUTH	CMPSS2_CTRIPOUTL	CMPSS1_CTRIPOUTH	CMPSS1_CTRIPOUTL
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
CMPSS8_CTRIPH	CMPSS8_CTRIPL	CMPSS7_CTRIPH	CMPSS7_CTRIPL	CMPSS6_CTRIPH	CMPSS6_CTRIPL	CMPSS5_CTRIPH	CMPSS5_CTRIPL
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
CMPSS4_CTRIPH	CMPSS4_CTRIPL	CMPSS3_CTRIPH	CMPSS3_CTRIPL	CMPSS2_CTRIPH	CMPSS2_CTRIPL	CMPSS1_CTRIPH	CMPSS1_CTRIPL
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 17-33. XBARCLR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CMPSS8_CTRIPOUTH	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS8_CTRIPOUTH bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
30	CMPSS8_CTRIPOUTL	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS8_CTRIPOUTL bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
29	CMPSS7_CTRIPOUTH	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS7_CTRIPOUTH bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
28	CMPSS7_CTRIPOUTL	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS7_CTRIPOUTL bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
27	CMPSS6_CTRIPOUTH	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS6_CTRIPOUTH bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
26	CMPSS6_CTRIPOUTL	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS6_CTRIPOUTL bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

Table 17-33. XBARCLR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	CMPSS5_CTRIPOUTH	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS5_CTRIPOUTH bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
24	CMPSS5_CTRIPOUTL	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS5_CTRIPOUTL bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
23	CMPSS4_CTRIPOUTH	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS4_CTRIPOUTH bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
22	CMPSS4_CTRIPOUTL	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS4_CTRIPOUTL bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
21	CMPSS3_CTRIPOUTH	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS3_CTRIPOUTH bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
20	CMPSS3_CTRIPOUTL	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS3_CTRIPOUTL bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
19	CMPSS2_CTRIPOUTH	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS2_CTRIPOUTH bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
18	CMPSS2_CTRIPOUTL	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS2_CTRIPOUTL bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
17	CMPSS1_CTRIPOUTH	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS1_CTRIPOUTH bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
16	CMPSS1_CTRIPOUTL	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS1_CTRIPOUTL bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
15	CMPSS8_CTRIPH	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS8_CTRIPH bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
14	CMPSS8_CTRIPL	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS8_CTRIPL bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
13	CMPSS7_CTRIPH	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS7_CTRIPH bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
12	CMPSS7_CTRIPL	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS7_CTRIPL bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

Table 17-33. XBARCLR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	CMPSS6_CTRIPH	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS6_CTRIPH bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
10	CMPSS6_CTRIPL	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS6_CTRIPL bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
9	CMPSS5_CTRIPH	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS5_CTRIPH bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
8	CMPSS5_CTRIPL	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS5_CTRIPL bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
7	CMPSS4_CTRIPH	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS4_CTRIPH bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
6	CMPSS4_CTRIPL	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS4_CTRIPL bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
5	CMPSS3_CTRIPH	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS3_CTRIPH bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
4	CMPSS3_CTRIPL	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS3_CTRIPL bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
3	CMPSS2_CTRIPH	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS2_CTRIPH bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
2	CMPSS2_CTRIPL	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS2_CTRIPL bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
1	CMPSS1_CTRIPH	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS1_CTRIPH bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
0	CMPSS1_CTRIPL	R-0/W1S	0h	Writing 1 to this bit clears the CMPSS1_CTRIPL bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

17.3.3.6 XBARCLR2 Register (Offset = Ah) [Reset = 0h]

XBARCLR2 is shown in Figure 17-29 and described in Table 17-34.

Return to the Summary Table.

This register is used to clear the flag(s) in the XBARFLG2 register.
1: Clears the corresponding bit in the XBARFLG2 register.
0: Writing 0 has no effect

Figure 17-29. XBARCLR2 Register

31	30	29	28	27	26	25	24
ADCCEVT1	ADCBEVT4	ADCBEVT3	ADCBEVT2	ADCBEVT1	ADCAEVT4	ADCAEVT3	ADCAEVT2
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
ADCAEVT1	EXTSYNCOUT	ECAP6_OUT	ECAP5_OUT	ECAP4_OUT	ECAP3_OUT	ECAP2_OUT	ECAP1_OUT
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
INPUT14	INPUT13	INPUT12	INPUT11	INPUT10	INPUT9	INPUT8	INPUT7
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
ADCSOCB	ADCSOCA	INPUT6	INPUT5	INPUT4	INPUT3	INPUT2	INPUT1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 17-34. XBARCLR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ADCCEVT1	R-0/W1S	0h	Writing 1 to this bit clears the ADCCEVT1 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
30	ADCBEVT4	R-0/W1S	0h	Writing 1 to this bit clears the ADCBEVT4 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
29	ADCBEVT3	R-0/W1S	0h	Writing 1 to this bit clears the ADCBEVT3 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
28	ADCBEVT2	R-0/W1S	0h	Writing 1 to this bit clears the ADCBEVT2 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
27	ADCBEVT1	R-0/W1S	0h	Writing 1 to this bit clears the ADCBEVT1 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
26	ADCAEVT4	R-0/W1S	0h	Writing 1 to this bit clears the ADCAEVT4 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
25	ADCAEVT3	R-0/W1S	0h	Writing 1 to this bit clears the ADCAEVT3 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

Table 17-34. XBARCLR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24	ADCAEVT2	R-0/W1S	0h	Writing 1 to this bit clears the ADCAEVT2 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
23	ADCAEVT1	R-0/W1S	0h	Writing 1 to this bit clears the ADCAEVT1 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
22	EXTSYNCOUT	R-0/W1S	0h	Writing 1 to this bit clears the EXTSYNCOUT bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
21	ECAP6_OUT	R-0/W1S	0h	Writing 1 to this bit clears the ECAP6_OUT bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
20	ECAP5_OUT	R-0/W1S	0h	Writing 1 to this bit clears the ECAP5_OUT bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
19	ECAP4_OUT	R-0/W1S	0h	Writing 1 to this bit clears the ECAP4_OUT bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
18	ECAP3_OUT	R-0/W1S	0h	Writing 1 to this bit clears the ECAP3_OUT bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
17	ECAP2_OUT	R-0/W1S	0h	Writing 1 to this bit clears the ECAP2_OUT bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
16	ECAP1_OUT	R-0/W1S	0h	Writing 1 to this bit clears the ECAP1_OUT bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
15	INPUT14	R-0/W1S	0h	Writing 1 to this bit clears the INPUT14 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
14	INPUT13	R-0/W1S	0h	Writing 1 to this bit clears the INPUT13 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
13	INPUT12	R-0/W1S	0h	Writing 1 to this bit clears the INPUT12 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
12	INPUT11	R-0/W1S	0h	Writing 1 to this bit clears the INPUT11 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
11	INPUT10	R-0/W1S	0h	Writing 1 to this bit clears the INPUT10 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
10	INPUT9	R-0/W1S	0h	Writing 1 to this bit clears the INPUT9 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
9	INPUT8	R-0/W1S	0h	Writing 1 to this bit clears the INPUT8 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

Table 17-34. XBARCLR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	INPUT7	R-0/W1S	0h	Writing 1 to this bit clears the INPUT7 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
7	ADCSOCB	R-0/W1S	0h	Writing 1 to this bit clears the ADCSOCB bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
6	ADCSOCA	R-0/W1S	0h	Writing 1 to this bit clears the ADCSOCA bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
5	INPUT6	R-0/W1S	0h	Writing 1 to this bit clears the INPUT6 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
4	INPUT5	R-0/W1S	0h	Writing 1 to this bit clears the INPUT5 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
3	INPUT4	R-0/W1S	0h	Writing 1 to this bit clears the INPUT4 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
2	INPUT3	R-0/W1S	0h	Writing 1 to this bit clears the INPUT3 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
1	INPUT2	R-0/W1S	0h	Writing 1 to this bit clears the INPUT2 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
0	INPUT1	R-0/W1S	0h	Writing 1 to this bit clears the INPUT1 bit in the XBARFLG2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

17.3.3.7 XBARCLR3 Register (Offset = Ch) [Reset = 0h]

XBARCLR3 is shown in Figure 17-30 and described in Table 17-35.

Return to the Summary Table.

This register is used to clear the flag(s) in the XBARFLG3 register.
1: Clears the corresponding bit in the XBARFLG3 register.
0: Writing 0 has no effect

Figure 17-30. XBARCLR3 Register

31	30	29	28	27	26	25	24
SD1FLT4_DRINT	SD1FLT4_COMPZ	SD1FLT3_DRINT	SD1FLT3_COMPZ	SD1FLT2_DRINT	SD1FLT2_COMPZ	SD1FLT1_DRINT	SD1FLT1_COMPZ
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
ECAP7_OUT	SD2FLT4_COMPH	SD2FLT4_COMPL	SD2FLT3_COMPH	SD2FLT3_COMPL	SD2FLT2_COMPH	SD2FLT2_COMPL	SD2FLT1_COMPH
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
SD2FLT1_COMPL	SD1FLT4_COMPH	SD1FLT4_COMPL	SD1FLT3_COMPH	SD1FLT3_COMPL	SD1FLT2_COMPH	SD1FLT2_COMPL	SD1FLT1_COMPH
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
SD1FLT1_COMPL	ADCDEVT4	ADCDEVT3	ADCDEVT2	ADCDEVT1	ADCCEVT4	ADCCEVT3	ADCCEVT2
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 17-35. XBARCLR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SD1FLT4_DRINT	R-0/W1S	0h	Writing 1 to this bit clears the SD1FLT4_DRINT bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
30	SD1FLT4_COMPZ	R-0/W1S	0h	Writing 1 to this bit clears the SD1FLT4_COMPZ bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
29	SD1FLT3_DRINT	R-0/W1S	0h	Writing 1 to this bit clears the SD1FLT3_DRINT bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
28	SD1FLT3_COMPZ	R-0/W1S	0h	Writing 1 to this bit clears the SD1FLT3_COMPZ bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
27	SD1FLT2_DRINT	R-0/W1S	0h	Writing 1 to this bit clears the SD1FLT2_DRINT bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
26	SD1FLT2_COMPZ	R-0/W1S	0h	Writing 1 to this bit clears the SD1FLT2_COMPZ bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

Table 17-35. XBARCLR3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	SD1FLT1_DRINT	R-0/W1S	0h	Writing 1 to this bit clears the SD1FLT1_DRINT bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
24	SD1FLT1_COMPZ	R-0/W1S	0h	Writing 1 to this bit clears the SD1FLT1_COMPZ bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
23	ECAP7_OUT	R-0/W1S	0h	Writing 1 to this bit clears the ECAP7_OUT bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
22	SD2FLT4_COMPH	R-0/W1S	0h	Writing 1 to this bit clears the SD2FLT4_COMPH bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
21	SD2FLT4_COMPL	R-0/W1S	0h	Writing 1 to this bit clears the SD2FLT4_COMPL bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
20	SD2FLT3_COMPH	R-0/W1S	0h	Writing 1 to this bit clears the SD2FLT3_COMPH bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
19	SD2FLT3_COMPL	R-0/W1S	0h	Writing 1 to this bit clears the SD2FLT3_COMPL bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
18	SD2FLT2_COMPH	R-0/W1S	0h	Writing 1 to this bit clears the SD2FLT2_COMPH bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
17	SD2FLT2_COMPL	R-0/W1S	0h	Writing 1 to this bit clears the SD2FLT2_COMPL bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
16	SD2FLT1_COMPH	R-0/W1S	0h	Writing 1 to this bit clears the SD2FLT1_COMPH bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
15	SD2FLT1_COMPL	R-0/W1S	0h	Writing 1 to this bit clears the SD2FLT1_COMPL bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
14	SD1FLT4_COMPH	R-0/W1S	0h	Writing 1 to this bit clears the SD1FLT4_COMPH bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
13	SD1FLT4_COMPL	R-0/W1S	0h	Writing 1 to this bit clears the SD1FLT4_COMPL bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
12	SD1FLT3_COMPH	R-0/W1S	0h	Writing 1 to this bit clears the SD1FLT3_COMPH bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

Table 17-35. XBARCLR3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	SD1FLT3_COMPL	R-0/W1S	0h	Writing 1 to this bit clears the SD1FLT3_COMPL bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
10	SD1FLT2_COMPH	R-0/W1S	0h	Writing 1 to this bit clears the SD1FLT2_COMPH bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
9	SD1FLT2_COMPL	R-0/W1S	0h	Writing 1 to this bit clears the SD1FLT2_COMPL bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
8	SD1FLT1_COMPH	R-0/W1S	0h	Writing 1 to this bit clears the SD1FLT1_COMPH bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
7	SD1FLT1_COMPL	R-0/W1S	0h	Writing 1 to this bit clears the SD1FLT1_COMPL bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
6	ADCDEVT4	R-0/W1S	0h	Writing 1 to this bit clears the ADCDEVT4 bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
5	ADCDEVT3	R-0/W1S	0h	Writing 1 to this bit clears the ADCDEVT3 bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
4	ADCDEVT2	R-0/W1S	0h	Writing 1 to this bit clears the ADCDEVT2 bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
3	ADCDEVT1	R-0/W1S	0h	Writing 1 to this bit clears the ADCDEVT1 bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
2	ADCCEVT4	R-0/W1S	0h	Writing 1 to this bit clears the ADCCEVT4 bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
1	ADCCEVT3	R-0/W1S	0h	Writing 1 to this bit clears the ADCCEVT3 bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
0	ADCCEVT2	R-0/W1S	0h	Writing 1 to this bit clears the ADCCEVT2 bit in the XBARFLG3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

17.3.3.8 XBARCLR4 Register (Offset = Eh) [Reset = 0h]

XBARCLR4 is shown in Figure 17-31 and described in Table 17-36.

Return to the Summary Table.

This register is used to clear the flag(s) in the XBARFLG4 register.
1: Clears the corresponding bit in the XBARFLG4 register.
0: Writing 0 has no effect

Figure 17-31. XBARCLR4 Register

31	30	29	28	27	26	25	24
CLAHALT	ECATSYNC1	ECATSYNC0	ERRORSTS_ERROR	CLB6_OUT5	CLB6_OUT4	CLB5_OUT5	CLB5_OUT4
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
CLB4_OUT5	CLB4_OUT4	CLB3_OUT5	CLB3_OUT4	CLB2_OUT5	CLB2_OUT4	CLB1_OUT5	CLB1_OUT4
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
CLB8_OUT5	CLB8_OUT4	CLB7_OUT5	CLB7_OUT4	MCANA_FEVT2	MCANA_FEVT1	MCANA_FEVT0	EMAC_PPS0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
SD2FLT4_DRINT	SD2FLT4_COMPZ	SD2FLT3_DRINT	SD2FLT3_COMPZ	SD2FLT2_DRINT	SD2FLT2_COMPZ	SD2FLT1_DRINT	SD2FLT1_COMPZ
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 17-36. XBARCLR4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CLAHALT	R	0h	Writing 1 to this bit clears the CLAHALT bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
30	ECATSYNC1	R-0/W1S	0h	Writing 1 to this bit clears the ECATSYNC1 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
29	ECATSYNC0	R-0/W1S	0h	Writing 1 to this bit clears the ECATSYNC0 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
28	ERRORSTS_ERROR	R-0/W1S	0h	Writing 1 to this bit clears the ERRORSTS_ERROR bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
27	CLB6_OUT5	R-0/W1S	0h	Writing 1 to this bit clears the CLB6_OUT5 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
26	CLB6_OUT4	R-0/W1S	0h	Writing 1 to this bit clears the CLB6_OUT4 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
25	CLB5_OUT5	R-0/W1S	0h	Writing 1 to this bit clears the CLB5_OUT5 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

Table 17-36. XBARCLR4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24	CLB5_OUT4	R-0/W1S	0h	Writing 1 to this bit clears the CLB5_OUT4 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
23	CLB4_OUT5	R-0/W1S	0h	Writing 1 to this bit clears the CLB4_OUT5 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
22	CLB4_OUT4	R-0/W1S	0h	Writing 1 to this bit clears the CLB4_OUT4 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
21	CLB3_OUT5	R-0/W1S	0h	Writing 1 to this bit clears the CLB3_OUT5 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
20	CLB3_OUT4	R-0/W1S	0h	Writing 1 to this bit clears the CLB3_OUT4 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
19	CLB2_OUT5	R-0/W1S	0h	Writing 1 to this bit clears the CLB2_OUT5 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
18	CLB2_OUT4	R-0/W1S	0h	Writing 1 to this bit clears the CLB2_OUT4 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
17	CLB1_OUT5	R-0/W1S	0h	Writing 1 to this bit clears the CLB1_OUT5 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
16	CLB1_OUT4	R-0/W1S	0h	Writing 1 to this bit clears the CLB1_OUT4 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
15	CLB8_OUT5	R-0/W1S	0h	Writing 1 to this bit clears the CLB8_OUT5 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
14	CLB8_OUT4	R-0/W1S	0h	Writing 1 to this bit clears the CLB8_OUT4 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
13	CLB7_OUT5	R-0/W1S	0h	Writing 1 to this bit clears the CLB7_OUT5 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
12	CLB7_OUT4	R-0/W1S	0h	Writing 1 to this bit clears the CLB7_OUT4 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
11	MCANA_FEVT2	R-0/W1S	0h	Writing 1 to this bit clears the MCANA_FEVT2 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

Table 17-36. XBARCLR4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MCANA_FEVT1	R-0/W1S	0h	Writing 1 to this bit clears the MCANA_FEVT1 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
9	MCANA_FEVT0	R-0/W1S	0h	Writing 1 to this bit clears the MCANA_FEVT0 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
8	EMAC_PPS0	R-0/W1S	0h	Writing 1 to this bit clears the EMAC_PPS0 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
7	SD2FLT4_DRINT	R-0/W1S	0h	Writing 1 to this bit clears the SD2FLT4_DRINT bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
6	SD2FLT4_COMPZ	R-0/W1S	0h	Writing 1 to this bit clears the SD2FLT4_COMPZ bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
5	SD2FLT3_DRINT	R-0/W1S	0h	Writing 1 to this bit clears the SD2FLT3_DRINT bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
4	SD2FLT3_COMPZ	R-0/W1S	0h	Writing 1 to this bit clears the SD2FLT3_COMPZ bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
3	SD2FLT2_DRINT	R-0/W1S	0h	Writing 1 to this bit clears the SD2FLT2_DRINT bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
2	SD2FLT2_COMPZ	R-0/W1S	0h	Writing 1 to this bit clears the SD2FLT2_COMPZ bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
1	SD2FLT1_DRINT	R-0/W1S	0h	Writing 1 to this bit clears the SD2FLT1_DRINT bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
0	SD2FLT1_COMPZ	R-0/W1S	0h	Writing 1 to this bit clears the SD2FLT1_COMPZ bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

17.3.4 EPWM_XBAR_REGS Registers

Table 17-37 lists the memory-mapped registers for the EPWM_XBAR_REGS registers. All register offset addresses not listed in Table 17-37 should be considered as reserved locations and the register contents should not be modified.

Table 17-37. EPWM_XBAR_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	TRIP4MUX0TO15CFG	ePWM XBAR Mux Configuration for TRIP4	EALLOW	Go
2h	TRIP4MUX16TO31CFG	ePWM XBAR Mux Configuration for TRIP4	EALLOW	Go
4h	TRIP5MUX0TO15CFG	ePWM XBAR Mux Configuration for TRIP5	EALLOW	Go
6h	TRIP5MUX16TO31CFG	ePWM XBAR Mux Configuration for TRIP5	EALLOW	Go
8h	TRIP7MUX0TO15CFG	ePWM XBAR Mux Configuration for TRIP7	EALLOW	Go
Ah	TRIP7MUX16TO31CFG	ePWM XBAR Mux Configuration for TRIP7	EALLOW	Go
Ch	TRIP8MUX0TO15CFG	ePWM XBAR Mux Configuration for TRIP8	EALLOW	Go
Eh	TRIP8MUX16TO31CFG	ePWM XBAR Mux Configuration for TRIP8	EALLOW	Go
10h	TRIP9MUX0TO15CFG	ePWM XBAR Mux Configuration for TRIP9	EALLOW	Go
12h	TRIP9MUX16TO31CFG	ePWM XBAR Mux Configuration for TRIP9	EALLOW	Go
14h	TRIP10MUX0TO15CFG	ePWM XBAR Mux Configuration for TRIP10	EALLOW	Go
16h	TRIP10MUX16TO31CFG	ePWM XBAR Mux Configuration for TRIP10	EALLOW	Go
18h	TRIP11MUX0TO15CFG	ePWM XBAR Mux Configuration for TRIP11	EALLOW	Go
1Ah	TRIP11MUX16TO31CFG	ePWM XBAR Mux Configuration for TRIP11	EALLOW	Go
1Ch	TRIP12MUX0TO15CFG	ePWM XBAR Mux Configuration for TRIP12	EALLOW	Go
1Eh	TRIP12MUX16TO31CFG	ePWM XBAR Mux Configuration for TRIP12	EALLOW	Go
20h	TRIP4MUXENABLE	ePWM XBAR Mux Enable for TRIP4	EALLOW	Go
22h	TRIP5MUXENABLE	ePWM XBAR Mux Enable for TRIP5	EALLOW	Go
24h	TRIP7MUXENABLE	ePWM XBAR Mux Enable for TRIP7	EALLOW	Go
26h	TRIP8MUXENABLE	ePWM XBAR Mux Enable for TRIP8	EALLOW	Go
28h	TRIP9MUXENABLE	ePWM XBAR Mux Enable for TRIP9	EALLOW	Go
2Ah	TRIP10MUXENABLE	ePWM XBAR Mux Enable for TRIP10	EALLOW	Go
2Ch	TRIP11MUXENABLE	ePWM XBAR Mux Enable for TRIP11	EALLOW	Go
2Eh	TRIP12MUXENABLE	ePWM XBAR Mux Enable for TRIP12	EALLOW	Go
38h	TRIPOUTINV	ePWM XBAR Output Inversion Register	EALLOW	Go
3Eh	TRIPLOCK	ePWM XBAR Configuration Lock register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 17-38 shows the codes that are used for access types in this section.

Table 17-38. EPWM_XBAR_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		

Table 17-38. EPWM_XBAR_REGS Access Type Codes (continued)

Access Type	Code	Description
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

17.3.4.1 TRIP4MUX0TO15CFG Register (Offset = 0h) [Reset = 0h]

TRIP4MUX0TO15CFG is shown in Figure 17-32 and described in Table 17-39.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP4

Figure 17-32. TRIP4MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-39. TRIP4MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-39. TRIP4MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-39. TRIP4MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.2 TRIP4MUX16TO31CFG Register (Offset = 2h) [Reset = 0h]

TRIP4MUX16TO31CFG is shown in Figure 17-33 and described in Table 17-40.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP4

Figure 17-33. TRIP4MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-40. TRIP4MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-40. TRIP4MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-40. TRIP4MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.3 TRIP5MUX0TO15CFG Register (Offset = 4h) [Reset = 0h]

TRIP5MUX0TO15CFG is shown in Figure 17-34 and described in Table 17-41.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP5

Figure 17-34. TRIP5MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-41. TRIP5MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-41. TRIP5MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-41. TRIP5MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.4 TRIP5MUX16TO31CFG Register (Offset = 6h) [Reset = 0h]

TRIP5MUX16TO31CFG is shown in Figure 17-35 and described in Table 17-42.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP5

Figure 17-35. TRIP5MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-42. TRIP5MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-42. TRIP5MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-42. TRIP5MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.5 TRIP7MUX0TO15CFG Register (Offset = 8h) [Reset = 0h]

TRIP7MUX0TO15CFG is shown in Figure 17-36 and described in Table 17-43.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP7

Figure 17-36. TRIP7MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-43. TRIP7MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-43. TRIP7MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-43. TRIP7MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.6 TRIP7MUX16TO31CFG Register (Offset = Ah) [Reset = 0h]

TRIP7MUX16TO31CFG is shown in Figure 17-37 and described in Table 17-44.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP7

Figure 17-37. TRIP7MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-44. TRIP7MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-44. TRIP7MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-44. TRIP7MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.7 TRIP8MUX0TO15CFG Register (Offset = Ch) [Reset = 0h]

TRIP8MUX0TO15CFG is shown in Figure 17-38 and described in Table 17-45.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP8

Figure 17-38. TRIP8MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-45. TRIP8MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-45. TRIP8MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-45. TRIP8MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.8 TRIP8MUX16TO31CFG Register (Offset = Eh) [Reset = 0h]

TRIP8MUX16TO31CFG is shown in Figure 17-39 and described in Table 17-46.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP8

Figure 17-39. TRIP8MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-46. TRIP8MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-46. TRIP8MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-46. TRIP8MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.9 TRIP9MUX0TO15CFG Register (Offset = 10h) [Reset = 0h]

TRIP9MUX0TO15CFG is shown in Figure 17-40 and described in Table 17-47.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP9

Figure 17-40. TRIP9MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-47. TRIP9MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-47. TRIP9MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-47. TRIP9MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.10 TRIP9MUX16TO31CFG Register (Offset = 12h) [Reset = 0h]

TRIP9MUX16TO31CFG is shown in Figure 17-41 and described in Table 17-48.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP9

Figure 17-41. TRIP9MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-48. TRIP9MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-48. TRIP9MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-48. TRIP9MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.11 TRIP10MUX0TO15CFG Register (Offset = 14h) [Reset = 0h]

TRIP10MUX0TO15CFG is shown in Figure 17-42 and described in Table 17-49.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP10

Figure 17-42. TRIP10MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-49. TRIP10MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-49. TRIP10MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-49. TRIP10MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.12 TRIP10MUX16TO31CFG Register (Offset = 16h) [Reset = 0h]

TRIP10MUX16TO31CFG is shown in Figure 17-43 and described in Table 17-50.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP10

Figure 17-43. TRIP10MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-50. TRIP10MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-50. TRIP10MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-50. TRIP10MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.13 TRIP11MUX0TO15CFG Register (Offset = 18h) [Reset = 0h]

TRIP11MUX0TO15CFG is shown in Figure 17-44 and described in Table 17-51.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP11

Figure 17-44. TRIP11MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-51. TRIP11MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-51. TRIP11MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-51. TRIP11MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.14 TRIP11MUX16TO31CFG Register (Offset = 1Ah) [Reset = 0h]

TRIP11MUX16TO31CFG is shown in Figure 17-45 and described in Table 17-52.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP11

Figure 17-45. TRIP11MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-52. TRIP11MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-52. TRIP11MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-52. TRIP11MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.15 TRIP12MUX0TO15CFG Register (Offset = 1Ch) [Reset = 0h]

TRIP12MUX0TO15CFG is shown in Figure 17-46 and described in Table 17-53.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP12

Figure 17-46. TRIP12MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-53. TRIP12MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-53. TRIP12MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-53. TRIP12MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.16 TRIP12MUX16TO31CFG Register (Offset = 1Eh) [Reset = 0h]

TRIP12MUX16TO31CFG is shown in Figure 17-47 and described in Table 17-54.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP12

Figure 17-47. TRIP12MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-54. TRIP12MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-54. TRIP12MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-54. TRIP12MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.17 TRIP4MUXENABLE Register (Offset = 20h) [Reset = 0h]

TRIP4MUXENABLE is shown in Figure 17-48 and described in Table 17-55.

Return to the Summary Table.

ePWM XBAR Mux Enable for TRIP4

Figure 17-48. TRIP4MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-55. TRIP4MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux31 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux31 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux30 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux30 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux29 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux29 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux28 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux28 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux27 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux27 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-55. TRIP4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux26 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux26 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux25 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux25 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux24 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux24 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux23 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux23 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux22 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux22 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux21 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux21 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux20 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux20 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux19 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux19 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-55. TRIP4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux18 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux18 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux17 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux17 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux16 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux16 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux15 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux15 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux14 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux14 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux13 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux13 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux12 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux12 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux11 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux11 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-55. TRIP4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux10 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux10 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux9 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux9 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux8 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux8 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux7 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux7 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux6 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux6 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux5 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux5 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux4 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux4 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux3 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux3 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-55. TRIP4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux2 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux2 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux1 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux1 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of Mux0 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux0 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux0 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.18 TRIP5MUXENABLE Register (Offset = 22h) [Reset = 0h]

TRIP5MUXENABLE is shown in Figure 17-49 and described in Table 17-56.

Return to the Summary Table.

ePWM XBAR Mux Enable for TRIP5

Figure 17-49. TRIP5MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-56. TRIP5MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux31 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux31 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux30 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux30 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux29 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux29 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux28 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux28 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux27 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux27 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-56. TRIP5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux26 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux26 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux25 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux25 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux24 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux24 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux23 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux23 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux22 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux22 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux21 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux21 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux20 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux20 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux19 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux19 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-56. TRIP5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux18 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux18 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux17 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux17 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux16 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux16 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux15 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux15 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux14 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux14 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux13 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux13 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux12 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux12 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux11 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux11 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-56. TRIP5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux10 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux10 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux9 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux9 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux8 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux8 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux7 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux7 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux6 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux6 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux5 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux5 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux4 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux4 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux3 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux3 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-56. TRIP5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux2 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux2 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux1 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux1 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux0 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux0 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.19 TRIP7MUXENABLE Register (Offset = 24h) [Reset = 0h]

TRIP7MUXENABLE is shown in Figure 17-50 and described in Table 17-57.

Return to the Summary Table.

ePWM XBAR Mux Enable for TRIP7

Figure 17-50. TRIP7MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-57. TRIP7MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux31 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux31 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux30 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux30 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux29 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux29 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux28 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux28 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux27 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux27 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-57. TRIP7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux26 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux26 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux25 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux25 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux24 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux24 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux23 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux23 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux22 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux22 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux21 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux21 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux20 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux20 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux19 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux19 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-57. TRIP7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux18 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux18 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux17 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux17 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux16 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux16 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux15 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux15 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux14 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux14 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux13 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux13 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux12 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux12 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux11 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux11 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-57. TRIP7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux10 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux10 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux9 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux9 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux8 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux8 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux7 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux7 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux6 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux6 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux5 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux5 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux4 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux4 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux3 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux3 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-57. TRIP7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux2 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux2 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux1 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux1 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux0 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux0 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.20 TRIP8MUXENABLE Register (Offset = 26h) [Reset = 0h]

TRIP8MUXENABLE is shown in Figure 17-51 and described in Table 17-58.

Return to the Summary Table.

ePWM XBAR Mux Enable for TRIP8

Figure 17-51. TRIP8MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-58. TRIP8MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux31 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux31 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux30 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux30 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux29 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux29 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux28 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux28 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux27 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux27 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-58. TRIP8MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux26 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux26 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux25 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux25 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux24 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux24 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux23 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux23 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux22 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux22 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux21 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux21 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux20 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux20 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux19 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux19 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-58. TRIP8MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux18 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux18 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux17 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux17 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux16 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux16 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux15 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux15 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux14 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux14 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux13 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux13 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux12 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux12 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux11 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux11 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-58. TRIP8MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux10 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux10 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux9 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux9 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux8 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux8 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux7 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux7 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux6 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux6 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux5 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux5 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux4 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux4 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux3 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux3 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-58. TRIP8MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux2 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux2 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux1 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux1 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux0 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux0 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.21 TRIP9MUXENABLE Register (Offset = 28h) [Reset = 0h]

TRIP9MUXENABLE is shown in Figure 17-52 and described in Table 17-59.

Return to the Summary Table.

ePWM XBAR Mux Enable for TRIP9

Figure 17-52. TRIP9MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-59. TRIP9MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux31 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux31 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux30 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux30 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux29 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux29 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux28 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux28 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux27 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux27 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-59. TRIP9MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux26 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux26 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux25 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux25 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux24 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux24 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux23 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux23 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux22 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux22 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux21 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux21 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux20 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux20 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux19 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux19 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-59. TRIP9MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux18 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux18 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux17 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux17 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux16 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux16 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux15 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux15 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux14 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux14 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux13 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux13 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux12 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux12 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux11 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux11 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-59. TRIP9MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux10 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux10 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux9 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux9 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux8 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux8 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux7 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux7 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux6 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux6 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux5 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux5 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux4 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux4 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux3 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux3 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-59. TRIP9MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux2 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux2 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux1 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux1 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux0 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux0 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.22 TRIP10MUXENABLE Register (Offset = 2Ah) [Reset = 0h]

TRIP10MUXENABLE is shown in Figure 17-53 and described in Table 17-60.

Return to the Summary Table.

ePWM XBAR Mux Enable for TRIP10

Figure 17-53. TRIP10MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-60. TRIP10MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux31 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux31 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux30 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux30 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux29 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux29 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux28 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux28 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux27 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux27 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-60. TRIP10MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux26 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux26 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux25 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux25 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux24 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux24 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux23 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux23 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux22 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux22 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux21 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux21 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux20 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux20 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux19 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux19 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-60. TRIP10MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux18 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux18 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux17 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux17 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux16 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux16 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux15 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux15 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux14 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux14 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux13 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux13 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux12 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux12 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux11 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux11 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-60. TRIP10MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux10 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux10 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux9 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux9 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux8 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux8 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux7 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux7 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux6 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux6 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux5 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux5 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux4 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux4 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux3 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux3 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-60. TRIP10MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux2 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux2 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux1 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux1 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux0 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux0 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.23 TRIP11MUXENABLE Register (Offset = 2Ch) [Reset = 0h]

TRIP11MUXENABLE is shown in Figure 17-54 and described in Table 17-61.

Return to the Summary Table.

ePWM XBAR Mux Enable for TRIP11

Figure 17-54. TRIP11MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-61. TRIP11MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux31 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux31 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux30 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux30 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux29 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux29 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux28 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux28 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux27 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux27 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-61. TRIP11MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux26 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux26 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux25 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux25 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux24 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux24 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux23 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux23 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux22 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux22 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux21 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux21 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux20 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux20 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux19 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux19 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-61. TRIP11MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux18 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux18 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux17 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux17 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux16 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux16 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux15 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux15 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux14 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux14 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux13 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux13 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux12 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux12 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux11 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux11 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-61. TRIP11MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux10 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux10 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux9 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux9 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux8 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux8 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux7 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux7 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux6 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux6 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux5 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux5 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux4 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux4 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux3 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux3 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-61. TRIP11MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux2 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux2 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux1 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux1 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux0 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux0 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.24 TRIP12MUXENABLE Register (Offset = 2Eh) [Reset = 0h]

TRIP12MUXENABLE is shown in Figure 17-55 and described in Table 17-62.

Return to the Summary Table.

ePWM XBAR Mux Enable for TRIP12

Figure 17-55. TRIP12MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-62. TRIP12MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux31 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux31 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux30 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux30 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux29 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux29 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux28 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux28 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux27 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux27 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-62. TRIP12MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux26 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux26 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux25 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux25 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux24 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux24 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux23 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux23 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux22 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux22 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux21 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux21 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux20 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux20 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux19 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux19 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-62. TRIP12MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux18 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux18 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux17 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux17 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux16 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux16 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux15 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux15 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux14 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux14 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux13 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux13 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux12 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux12 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux11 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux11 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-62. TRIP12MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux10 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux10 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux9 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux9 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux8 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux8 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux7 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux7 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux6 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux6 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux5 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux5 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux4 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux4 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux3 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux3 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-62. TRIP12MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux2 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux2 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux1 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux1 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux0 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux0 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.25 TRIPOUTINV Register (Offset = 38h) [Reset = 0h]

TRIPOUTINV is shown in Figure 17-56 and described in Table 17-63.

Return to the Summary Table.

ePWM XBAR Output Inversion Register

Figure 17-56. TRIPOUTINV Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
TRIP12	TRIP11	TRIP10	TRIP9	TRIP8	TRIP7	TRIP5	TRIP4
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-63. TRIPOUTINV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	RESERVED	R-0	0h	Reserved
7	TRIP12	R/W	0h	Selects polarity for TRIP12 of EPWM-XBAR 0: drives active high output 1: drives active-low output Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	TRIP11	R/W	0h	Selects polarity for TRIP11 of EPWM-XBAR 0: drives active high output 1: drives active-low output Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	TRIP10	R/W	0h	Selects polarity for TRIP10 of EPWM-XBAR 0: drives active high output 1: drives active-low output Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	TRIP9	R/W	0h	Selects polarity for TRIP9 of EPWM-XBAR 0: drives active high output 1: drives active-low output Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	TRIP8	R/W	0h	Selects polarity for TRIP8 of EPWM-XBAR 0: drives active high output 1: drives active-low output Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
2	TRIP7	R/W	0h	Selects polarity for TRIP7 of EPWM-XBAR 0: drives active high output 1: drives active-low output Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-63. TRIPOUTINV Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	TRIP5	R/W	0h	Selects polarity for TRIP5 of EPWM-XBAR 0: drives active high output 1: drives active-low output Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	TRIP4	R/W	0h	Selects polarity for TRIP4 of EPWM-XBAR 0: drives active high output 1: drives active-low output Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.4.26 TRIPLOCK Register (Offset = 3Eh) [Reset = 0h]

TRIPLOCK is shown in Figure 17-57 and described in Table 17-64.

Return to the Summary Table.

ePWM XBAR Configuration Lock register

Figure 17-57. TRIPLOCK Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK
R-0-0h							R/WSonce-0h

Table 17-64. TRIPLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Bit-0 of this register can be set only if KEY= 0x5a5a Reset type: CPU1.SYSRSn
15-1	RESERVED	R-0	0h	Reserved
0	LOCK	R/WSonce	0h	Locks the configuration for EPWM-XBAR. Once the configuration is locked, writes to the below registers for EPWM-XBAR is blocked. Registers Affected by the LOCK mechanism: EPWM-XBAROUTyMUX0TO15CFG EPWM-XBAROUTyMUX16TO31CFG EPWM-XBAROUTyMUXENABLE EPWM-XBAROUTLATEN EPWM-XBAROUTINV 0: Writes to the above registers are allowed 1: Writes to the above registers are blocked Note: [1] LOCK mechanism only apples to writes. Reads are never blocked. Reset type: CPU1.SYSRSn

17.3.5 CLB_XBAR_REGS Registers

Table 17-65 lists the memory-mapped registers for the CLB_XBAR_REGS registers. All register offset addresses not listed in Table 17-65 should be considered as reserved locations and the register contents should not be modified.

Table 17-65. CLB_XBAR_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	AUXSIG0MUX0TO15CFG	CLB XBAR Mux Configuration for Output-0	EALLOW	Go
2h	AUXSIG0MUX16TO31CFG	CLB XBAR Mux Configuration for Output-0	EALLOW	Go
4h	AUXSIG1MUX0TO15CFG	CLB XBAR Mux Configuration for Output-1	EALLOW	Go
6h	AUXSIG1MUX16TO31CFG	CLB XBAR Mux Configuration for Output-1	EALLOW	Go
8h	AUXSIG2MUX0TO15CFG	CLB XBAR Mux Configuration for Output-2	EALLOW	Go
Ah	AUXSIG2MUX16TO31CFG	CLB XBAR Mux Configuration for Output-2	EALLOW	Go
Ch	AUXSIG3MUX0TO15CFG	CLB XBAR Mux Configuration for Output-3	EALLOW	Go
Eh	AUXSIG3MUX16TO31CFG	CLB XBAR Mux Configuration for Output-3	EALLOW	Go
10h	AUXSIG4MUX0TO15CFG	CLB XBAR Mux Configuration for Output-4	EALLOW	Go
12h	AUXSIG4MUX16TO31CFG	CLB XBAR Mux Configuration for Output-4	EALLOW	Go
14h	AUXSIG5MUX0TO15CFG	CLB XBAR Mux Configuration for Output-5	EALLOW	Go
16h	AUXSIG5MUX16TO31CFG	CLB XBAR Mux Configuration for Output-5	EALLOW	Go
18h	AUXSIG6MUX0TO15CFG	CLB XBAR Mux Configuration for Output-6	EALLOW	Go
1Ah	AUXSIG6MUX16TO31CFG	CLB XBAR Mux Configuration for Output-6	EALLOW	Go
1Ch	AUXSIG7MUX0TO15CFG	CLB XBAR Mux Configuration for Output-7	EALLOW	Go
1Eh	AUXSIG7MUX16TO31CFG	CLB XBAR Mux Configuration for Output-7	EALLOW	Go
20h	AUXSIG0MUXENABLE	CLB XBAR Mux Enable Register for Output-0	EALLOW	Go
22h	AUXSIG1MUXENABLE	CLB XBAR Mux Enable Register for Output-1	EALLOW	Go
24h	AUXSIG2MUXENABLE	CLB XBAR Mux Enable Register for Output-2	EALLOW	Go
26h	AUXSIG3MUXENABLE	CLB XBAR Mux Enable Register for Output-3	EALLOW	Go
28h	AUXSIG4MUXENABLE	CLB XBAR Mux Enable Register for Output-4	EALLOW	Go
2Ah	AUXSIG5MUXENABLE	CLB XBAR Mux Enable Register for Output-5	EALLOW	Go
2Ch	AUXSIG6MUXENABLE	CLB XBAR Mux Enable Register for Output-6	EALLOW	Go
2Eh	AUXSIG7MUXENABLE	CLB XBAR Mux Enable Register for Output-7	EALLOW	Go
38h	AUXSIGOUTINV	CLB XBAR Output Inversion Register	EALLOW	Go
3Eh	AUXSIGLOCK	ClbXbar Configuration Lock register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 17-66 shows the codes that are used for access types in this section.

Table 17-66. CLB_XBAR_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		

Table 17-66. CLB_XBAR_REGS Access Type Codes (continued)

Access Type	Code	Description
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

17.3.5.1 AUXSIG0MUX0TO15CFG Register (Offset = 0h) [Reset = 0h]

AUXSIG0MUX0TO15CFG is shown in Figure 17-58 and described in Table 17-67.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-0

Figure 17-58. AUXSIG0MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-67. AUXSIG0MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Group Select Bits for MUX15: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX15 01 : Select .1 input for MUX15 10 : Select .2 input for MUX15 11 : Select .3 input for MUX15 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Group Select Bits for MUX14: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX14 01 : Select .1 input for MUX14 10 : Select .2 input for MUX14 11 : Select .3 input for MUX14 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Group Select Bits for MUX13: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX13 01 : Select .1 input for MUX13 10 : Select .2 input for MUX13 11 : Select .3 input for MUX13 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Group Select Bits for MUX12: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX12 01 : Select .1 input for MUX12 10 : Select .2 input for MUX12 11 : Select .3 input for MUX12 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Group Select Bits for MUX11: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX11 01 : Select .1 input for MUX11 10 : Select .2 input for MUX11 11 : Select .3 input for MUX11 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-67. AUXSIG0MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX10	R/W	0h	Group Select Bits for MUX10: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX10 01 : Select .1 input for MUX10 10 : Select .2 input for MUX10 11 : Select .3 input for MUX10 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX9	R/W	0h	Group Select Bits for MUX9: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX9 01 : Select .1 input for MUX9 10 : Select .2 input for MUX9 11 : Select .3 input for MUX9 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Group Select Bits for MUX8: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX8 01 : Select .1 input for MUX8 10 : Select .2 input for MUX8 11 : Select .3 input for MUX8 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Group Select Bits for MUX7: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX7 01 : Select .1 input for MUX7 10 : Select .2 input for MUX7 11 : Select .3 input for MUX7 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Group Select Bits for MUX6: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX6 01 : Select .1 input for MUX6 10 : Select .2 input for MUX6 11 : Select .3 input for MUX6 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Group Select Bits for MUX5: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX5 01 : Select .1 input for MUX5 10 : Select .2 input for MUX5 11 : Select .3 input for MUX5 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Group Select Bits for MUX4: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX4 01 : Select .1 input for MUX4 10 : Select .2 input for MUX4 11 : Select .3 input for MUX4 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-67. AUXSIG0MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX3	R/W	0h	Group Select Bits for MUX3: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX3 01 : Select .1 input for MUX3 10 : Select .2 input for MUX3 11 : Select .3 input for MUX3 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Group Select Bits for MUX2: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX2 01 : Select .1 input for MUX2 10 : Select .2 input for MUX2 11 : Select .3 input for MUX2 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX1	R/W	0h	Group Select Bits for MUX1: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX1 01 : Select .1 input for MUX1 10 : Select .2 input for MUX1 11 : Select .3 input for MUX1 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Group Select Bits for MUX0: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX0 01 : Select .1 input for MUX0 10 : Select .2 input for MUX0 11 : Select .3 input for MUX0 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.2 AUXSIG0MUX16TO31CFG Register (Offset = 2h) [Reset = 0h]

AUXSIG0MUX16TO31CFG is shown in Figure 17-59 and described in Table 17-68.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-0

Figure 17-59. AUXSIG0MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-68. AUXSIG0MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Group Select Bits for MUX31: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX31 01 : Select .1 input for MUX31 10 : Select .2 input for MUX31 11 : Select .3 input for MUX31 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Group Select Bits for MUX30: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX30 01 : Select .1 input for MUX30 10 : Select .2 input for MUX30 11 : Select .3 input for MUX30 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Group Select Bits for MUX29: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX29 01 : Select .1 input for MUX29 10 : Select .2 input for MUX29 11 : Select .3 input for MUX29 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Group Select Bits for MUX28: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX28 01 : Select .1 input for MUX28 10 : Select .2 input for MUX28 11 : Select .3 input for MUX28 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Group Select Bits for MUX27: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX27 01 : Select .1 input for MUX27 10 : Select .2 input for MUX27 11 : Select .3 input for MUX27 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-68. AUXSIG0MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX26	R/W	0h	Group Select Bits for MUX26: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX26 01 : Select .1 input for MUX26 10 : Select .2 input for MUX26 11 : Select .3 input for MUX26 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX25	R/W	0h	Group Select Bits for MUX25: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX25 01 : Select .1 input for MUX25 10 : Select .2 input for MUX25 11 : Select .3 input for MUX25 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Group Select Bits for MUX24: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX24 01 : Select .1 input for MUX24 10 : Select .2 input for MUX24 11 : Select .3 input for MUX24 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Group Select Bits for MUX23: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX23 01 : Select .1 input for MUX23 10 : Select .2 input for MUX23 11 : Select .3 input for MUX23 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Group Select Bits for MUX22: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX22 01 : Select .1 input for MUX22 10 : Select .2 input for MUX22 11 : Select .3 input for MUX22 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Group Select Bits for MUX21: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX21 01 : Select .1 input for MUX21 10 : Select .2 input for MUX21 11 : Select .3 input for MUX21 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Group Select Bits for MUX20: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX20 01 : Select .1 input for MUX20 10 : Select .2 input for MUX20 11 : Select .3 input for MUX20 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-68. AUXSIG0MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX19	R/W	0h	Group Select Bits for MUX19: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX19 01 : Select .1 input for MUX19 10 : Select .2 input for MUX19 11 : Select .3 input for MUX19 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Group Select Bits for MUX18: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX18 01 : Select .1 input for MUX18 10 : Select .2 input for MUX18 11 : Select .3 input for MUX18 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX17	R/W	0h	Group Select Bits for MUX17: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX17 01 : Select .1 input for MUX17 10 : Select .2 input for MUX17 11 : Select .3 input for MUX17 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Group Select Bits for MUX16: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX16 01 : Select .1 input for MUX16 10 : Select .2 input for MUX16 11 : Select .3 input for MUX16 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.3 AUXSIG1MUX0TO15CFG Register (Offset = 4h) [Reset = 0h]

AUXSIG1MUX0TO15CFG is shown in Figure 17-60 and described in Table 17-69.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-1

Figure 17-60. AUXSIG1MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-69. AUXSIG1MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Group Select Bits for MUX15: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX15 01 : Select .1 input for MUX15 10 : Select .2 input for MUX15 11 : Select .3 input for MUX15 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Group Select Bits for MUX14: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX14 01 : Select .1 input for MUX14 10 : Select .2 input for MUX14 11 : Select .3 input for MUX14 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Group Select Bits for MUX13: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX13 01 : Select .1 input for MUX13 10 : Select .2 input for MUX13 11 : Select .3 input for MUX13 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Group Select Bits for MUX12: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX12 01 : Select .1 input for MUX12 10 : Select .2 input for MUX12 11 : Select .3 input for MUX12 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Group Select Bits for MUX11: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX11 01 : Select .1 input for MUX11 10 : Select .2 input for MUX11 11 : Select .3 input for MUX11 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-69. AUXSIG1MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX10	R/W	0h	Group Select Bits for MUX10: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX10 01 : Select .1 input for MUX10 10 : Select .2 input for MUX10 11 : Select .3 input for MUX10 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX9	R/W	0h	Group Select Bits for MUX9: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX9 01 : Select .1 input for MUX9 10 : Select .2 input for MUX9 11 : Select .3 input for MUX9 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Group Select Bits for MUX8: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX8 01 : Select .1 input for MUX8 10 : Select .2 input for MUX8 11 : Select .3 input for MUX8 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Group Select Bits for MUX7: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX7 01 : Select .1 input for MUX7 10 : Select .2 input for MUX7 11 : Select .3 input for MUX7 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Group Select Bits for MUX6: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX6 01 : Select .1 input for MUX6 10 : Select .2 input for MUX6 11 : Select .3 input for MUX6 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Group Select Bits for MUX5: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX5 01 : Select .1 input for MUX5 10 : Select .2 input for MUX5 11 : Select .3 input for MUX5 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Group Select Bits for MUX4: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX4 01 : Select .1 input for MUX4 10 : Select .2 input for MUX4 11 : Select .3 input for MUX4 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-69. AUXSIG1MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX3	R/W	0h	Group Select Bits for MUX3: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX3 01 : Select .1 input for MUX3 10 : Select .2 input for MUX3 11 : Select .3 input for MUX3 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Group Select Bits for MUX2: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX2 01 : Select .1 input for MUX2 10 : Select .2 input for MUX2 11 : Select .3 input for MUX2 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX1	R/W	0h	Group Select Bits for MUX1: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX1 01 : Select .1 input for MUX1 10 : Select .2 input for MUX1 11 : Select .3 input for MUX1 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Group Select Bits for MUX0: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX0 01 : Select .1 input for MUX0 10 : Select .2 input for MUX0 11 : Select .3 input for MUX0 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.4 AUXSIG1MUX16TO31CFG Register (Offset = 6h) [Reset = 0h]

AUXSIG1MUX16TO31CFG is shown in Figure 17-61 and described in Table 17-70.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-1

Figure 17-61. AUXSIG1MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-70. AUXSIG1MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Group Select Bits for MUX31: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX31 01 : Select .1 input for MUX31 10 : Select .2 input for MUX31 11 : Select .3 input for MUX31 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Group Select Bits for MUX30: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX30 01 : Select .1 input for MUX30 10 : Select .2 input for MUX30 11 : Select .3 input for MUX30 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Group Select Bits for MUX29: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX29 01 : Select .1 input for MUX29 10 : Select .2 input for MUX29 11 : Select .3 input for MUX29 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Group Select Bits for MUX28: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX28 01 : Select .1 input for MUX28 10 : Select .2 input for MUX28 11 : Select .3 input for MUX28 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Group Select Bits for MUX27: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX27 01 : Select .1 input for MUX27 10 : Select .2 input for MUX27 11 : Select .3 input for MUX27 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-70. AUXSIG1MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX26	R/W	0h	Group Select Bits for MUX26: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX26 01 : Select .1 input for MUX26 10 : Select .2 input for MUX26 11 : Select .3 input for MUX26 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX25	R/W	0h	Group Select Bits for MUX25: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX25 01 : Select .1 input for MUX25 10 : Select .2 input for MUX25 11 : Select .3 input for MUX25 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Group Select Bits for MUX24: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX24 01 : Select .1 input for MUX24 10 : Select .2 input for MUX24 11 : Select .3 input for MUX24 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Group Select Bits for MUX23: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX23 01 : Select .1 input for MUX23 10 : Select .2 input for MUX23 11 : Select .3 input for MUX23 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Group Select Bits for MUX22: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX22 01 : Select .1 input for MUX22 10 : Select .2 input for MUX22 11 : Select .3 input for MUX22 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Group Select Bits for MUX21: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX21 01 : Select .1 input for MUX21 10 : Select .2 input for MUX21 11 : Select .3 input for MUX21 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Group Select Bits for MUX20: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX20 01 : Select .1 input for MUX20 10 : Select .2 input for MUX20 11 : Select .3 input for MUX20 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-70. AUXSIG1MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX19	R/W	0h	Group Select Bits for MUX19: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX19 01 : Select .1 input for MUX19 10 : Select .2 input for MUX19 11 : Select .3 input for MUX19 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Group Select Bits for MUX18: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX18 01 : Select .1 input for MUX18 10 : Select .2 input for MUX18 11 : Select .3 input for MUX18 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX17	R/W	0h	Group Select Bits for MUX17: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX17 01 : Select .1 input for MUX17 10 : Select .2 input for MUX17 11 : Select .3 input for MUX17 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Group Select Bits for MUX16: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX16 01 : Select .1 input for MUX16 10 : Select .2 input for MUX16 11 : Select .3 input for MUX16 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.5 AUXSIG2MUX0TO15CFG Register (Offset = 8h) [Reset = 0h]

AUXSIG2MUX0TO15CFG is shown in Figure 17-62 and described in Table 17-71.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-2

Figure 17-62. AUXSIG2MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-71. AUXSIG2MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Group Select Bits for MUX15: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX15 01 : Select .1 input for MUX15 10 : Select .2 input for MUX15 11 : Select .3 input for MUX15 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Group Select Bits for MUX14: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX14 01 : Select .1 input for MUX14 10 : Select .2 input for MUX14 11 : Select .3 input for MUX14 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Group Select Bits for MUX13: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX13 01 : Select .1 input for MUX13 10 : Select .2 input for MUX13 11 : Select .3 input for MUX13 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Group Select Bits for MUX12: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX12 01 : Select .1 input for MUX12 10 : Select .2 input for MUX12 11 : Select .3 input for MUX12 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Group Select Bits for MUX11: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX11 01 : Select .1 input for MUX11 10 : Select .2 input for MUX11 11 : Select .3 input for MUX11 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-71. AUXSIG2MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX10	R/W	0h	Group Select Bits for MUX10: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX10 01 : Select .1 input for MUX10 10 : Select .2 input for MUX10 11 : Select .3 input for MUX10 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX9	R/W	0h	Group Select Bits for MUX9: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX9 01 : Select .1 input for MUX9 10 : Select .2 input for MUX9 11 : Select .3 input for MUX9 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Group Select Bits for MUX8: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX8 01 : Select .1 input for MUX8 10 : Select .2 input for MUX8 11 : Select .3 input for MUX8 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Group Select Bits for MUX7: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX7 01 : Select .1 input for MUX7 10 : Select .2 input for MUX7 11 : Select .3 input for MUX7 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Group Select Bits for MUX6: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX6 01 : Select .1 input for MUX6 10 : Select .2 input for MUX6 11 : Select .3 input for MUX6 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Group Select Bits for MUX5: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX5 01 : Select .1 input for MUX5 10 : Select .2 input for MUX5 11 : Select .3 input for MUX5 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Group Select Bits for MUX4: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX4 01 : Select .1 input for MUX4 10 : Select .2 input for MUX4 11 : Select .3 input for MUX4 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-71. AUXSIG2MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX3	R/W	0h	Group Select Bits for MUX3: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX3 01 : Select .1 input for MUX3 10 : Select .2 input for MUX3 11 : Select .3 input for MUX3 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Group Select Bits for MUX2: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX2 01 : Select .1 input for MUX2 10 : Select .2 input for MUX2 11 : Select .3 input for MUX2 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX1	R/W	0h	Group Select Bits for MUX1: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX1 01 : Select .1 input for MUX1 10 : Select .2 input for MUX1 11 : Select .3 input for MUX1 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Group Select Bits for MUX0: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX0 01 : Select .1 input for MUX0 10 : Select .2 input for MUX0 11 : Select .3 input for MUX0 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.6 AUXSIG2MUX16TO31CFG Register (Offset = Ah) [Reset = 0h]

AUXSIG2MUX16TO31CFG is shown in Figure 17-63 and described in Table 17-72.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-2

Figure 17-63. AUXSIG2MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-72. AUXSIG2MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Group Select Bits for MUX31: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX31 01 : Select .1 input for MUX31 10 : Select .2 input for MUX31 11 : Select .3 input for MUX31 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Group Select Bits for MUX30: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX30 01 : Select .1 input for MUX30 10 : Select .2 input for MUX30 11 : Select .3 input for MUX30 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Group Select Bits for MUX29: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX29 01 : Select .1 input for MUX29 10 : Select .2 input for MUX29 11 : Select .3 input for MUX29 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Group Select Bits for MUX28: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX28 01 : Select .1 input for MUX28 10 : Select .2 input for MUX28 11 : Select .3 input for MUX28 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Group Select Bits for MUX27: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX27 01 : Select .1 input for MUX27 10 : Select .2 input for MUX27 11 : Select .3 input for MUX27 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-72. AUXSIG2MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX26	R/W	0h	Group Select Bits for MUX26: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX26 01 : Select .1 input for MUX26 10 : Select .2 input for MUX26 11 : Select .3 input for MUX26 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX25	R/W	0h	Group Select Bits for MUX25: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX25 01 : Select .1 input for MUX25 10 : Select .2 input for MUX25 11 : Select .3 input for MUX25 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Group Select Bits for MUX24: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX24 01 : Select .1 input for MUX24 10 : Select .2 input for MUX24 11 : Select .3 input for MUX24 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Group Select Bits for MUX23: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX23 01 : Select .1 input for MUX23 10 : Select .2 input for MUX23 11 : Select .3 input for MUX23 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Group Select Bits for MUX22: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX22 01 : Select .1 input for MUX22 10 : Select .2 input for MUX22 11 : Select .3 input for MUX22 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Group Select Bits for MUX21: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX21 01 : Select .1 input for MUX21 10 : Select .2 input for MUX21 11 : Select .3 input for MUX21 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Group Select Bits for MUX20: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX20 01 : Select .1 input for MUX20 10 : Select .2 input for MUX20 11 : Select .3 input for MUX20 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-72. AUXSIG2MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX19	R/W	0h	Group Select Bits for MUX19: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX19 01 : Select .1 input for MUX19 10 : Select .2 input for MUX19 11 : Select .3 input for MUX19 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Group Select Bits for MUX18: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX18 01 : Select .1 input for MUX18 10 : Select .2 input for MUX18 11 : Select .3 input for MUX18 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX17	R/W	0h	Group Select Bits for MUX17: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX17 01 : Select .1 input for MUX17 10 : Select .2 input for MUX17 11 : Select .3 input for MUX17 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Group Select Bits for MUX16: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX16 01 : Select .1 input for MUX16 10 : Select .2 input for MUX16 11 : Select .3 input for MUX16 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.7 AUXSIG3MUX0TO15CFG Register (Offset = Ch) [Reset = 0h]

AUXSIG3MUX0TO15CFG is shown in Figure 17-64 and described in Table 17-73.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-3

Figure 17-64. AUXSIG3MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-73. AUXSIG3MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Group Select Bits for MUX15: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX15 01 : Select .1 input for MUX15 10 : Select .2 input for MUX15 11 : Select .3 input for MUX15 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Group Select Bits for MUX14: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX14 01 : Select .1 input for MUX14 10 : Select .2 input for MUX14 11 : Select .3 input for MUX14 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Group Select Bits for MUX13: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX13 01 : Select .1 input for MUX13 10 : Select .2 input for MUX13 11 : Select .3 input for MUX13 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Group Select Bits for MUX12: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX12 01 : Select .1 input for MUX12 10 : Select .2 input for MUX12 11 : Select .3 input for MUX12 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Group Select Bits for MUX11: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX11 01 : Select .1 input for MUX11 10 : Select .2 input for MUX11 11 : Select .3 input for MUX11 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-73. AUXSIG3MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX10	R/W	0h	Group Select Bits for MUX10: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX10 01 : Select .1 input for MUX10 10 : Select .2 input for MUX10 11 : Select .3 input for MUX10 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX9	R/W	0h	Group Select Bits for MUX9: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX9 01 : Select .1 input for MUX9 10 : Select .2 input for MUX9 11 : Select .3 input for MUX9 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Group Select Bits for MUX8: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX8 01 : Select .1 input for MUX8 10 : Select .2 input for MUX8 11 : Select .3 input for MUX8 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Group Select Bits for MUX7: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX7 01 : Select .1 input for MUX7 10 : Select .2 input for MUX7 11 : Select .3 input for MUX7 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Group Select Bits for MUX6: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX6 01 : Select .1 input for MUX6 10 : Select .2 input for MUX6 11 : Select .3 input for MUX6 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Group Select Bits for MUX5: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX5 01 : Select .1 input for MUX5 10 : Select .2 input for MUX5 11 : Select .3 input for MUX5 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Group Select Bits for MUX4: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX4 01 : Select .1 input for MUX4 10 : Select .2 input for MUX4 11 : Select .3 input for MUX4 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-73. AUXSIG3MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX3	R/W	0h	Group Select Bits for MUX3: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX3 01 : Select .1 input for MUX3 10 : Select .2 input for MUX3 11 : Select .3 input for MUX3 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Group Select Bits for MUX2: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX2 01 : Select .1 input for MUX2 10 : Select .2 input for MUX2 11 : Select .3 input for MUX2 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX1	R/W	0h	Group Select Bits for MUX1: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX1 01 : Select .1 input for MUX1 10 : Select .2 input for MUX1 11 : Select .3 input for MUX1 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Group Select Bits for MUX0: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX0 01 : Select .1 input for MUX0 10 : Select .2 input for MUX0 11 : Select .3 input for MUX0 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.8 AUXSIG3MUX16TO31CFG Register (Offset = Eh) [Reset = 0h]

AUXSIG3MUX16TO31CFG is shown in Figure 17-65 and described in Table 17-74.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-3

Figure 17-65. AUXSIG3MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-74. AUXSIG3MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Group Select Bits for MUX31: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX31 01 : Select .1 input for MUX31 10 : Select .2 input for MUX31 11 : Select .3 input for MUX31 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Group Select Bits for MUX30: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX30 01 : Select .1 input for MUX30 10 : Select .2 input for MUX30 11 : Select .3 input for MUX30 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Group Select Bits for MUX29: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX29 01 : Select .1 input for MUX29 10 : Select .2 input for MUX29 11 : Select .3 input for MUX29 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Group Select Bits for MUX28: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX28 01 : Select .1 input for MUX28 10 : Select .2 input for MUX28 11 : Select .3 input for MUX28 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Group Select Bits for MUX27: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX27 01 : Select .1 input for MUX27 10 : Select .2 input for MUX27 11 : Select .3 input for MUX27 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-74. AUXSIG3MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX26	R/W	0h	Group Select Bits for MUX26: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX26 01 : Select .1 input for MUX26 10 : Select .2 input for MUX26 11 : Select .3 input for MUX26 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX25	R/W	0h	Group Select Bits for MUX25: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX25 01 : Select .1 input for MUX25 10 : Select .2 input for MUX25 11 : Select .3 input for MUX25 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Group Select Bits for MUX24: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX24 01 : Select .1 input for MUX24 10 : Select .2 input for MUX24 11 : Select .3 input for MUX24 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Group Select Bits for MUX23: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX23 01 : Select .1 input for MUX23 10 : Select .2 input for MUX23 11 : Select .3 input for MUX23 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Group Select Bits for MUX22: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX22 01 : Select .1 input for MUX22 10 : Select .2 input for MUX22 11 : Select .3 input for MUX22 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Group Select Bits for MUX21: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX21 01 : Select .1 input for MUX21 10 : Select .2 input for MUX21 11 : Select .3 input for MUX21 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Group Select Bits for MUX20: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX20 01 : Select .1 input for MUX20 10 : Select .2 input for MUX20 11 : Select .3 input for MUX20 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-74. AUXSIG3MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX19	R/W	0h	Group Select Bits for MUX19: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX19 01 : Select .1 input for MUX19 10 : Select .2 input for MUX19 11 : Select .3 input for MUX19 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Group Select Bits for MUX18: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX18 01 : Select .1 input for MUX18 10 : Select .2 input for MUX18 11 : Select .3 input for MUX18 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX17	R/W	0h	Group Select Bits for MUX17: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX17 01 : Select .1 input for MUX17 10 : Select .2 input for MUX17 11 : Select .3 input for MUX17 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Group Select Bits for MUX16: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX16 01 : Select .1 input for MUX16 10 : Select .2 input for MUX16 11 : Select .3 input for MUX16 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.9 AUXSIG4MUX0TO15CFG Register (Offset = 10h) [Reset = 0h]

AUXSIG4MUX0TO15CFG is shown in Figure 17-66 and described in Table 17-75.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-4

Figure 17-66. AUXSIG4MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-75. AUXSIG4MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Group Select Bits for MUX15: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX15 01 : Select .1 input for MUX15 10 : Select .2 input for MUX15 11 : Select .3 input for MUX15 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Group Select Bits for MUX14: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX14 01 : Select .1 input for MUX14 10 : Select .2 input for MUX14 11 : Select .3 input for MUX14 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Group Select Bits for MUX13: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX13 01 : Select .1 input for MUX13 10 : Select .2 input for MUX13 11 : Select .3 input for MUX13 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Group Select Bits for MUX12: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX12 01 : Select .1 input for MUX12 10 : Select .2 input for MUX12 11 : Select .3 input for MUX12 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Group Select Bits for MUX11: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX11 01 : Select .1 input for MUX11 10 : Select .2 input for MUX11 11 : Select .3 input for MUX11 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-75. AUXSIG4MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX10	R/W	0h	Group Select Bits for MUX10: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX10 01 : Select .1 input for MUX10 10 : Select .2 input for MUX10 11 : Select .3 input for MUX10 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX9	R/W	0h	Group Select Bits for MUX9: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX9 01 : Select .1 input for MUX9 10 : Select .2 input for MUX9 11 : Select .3 input for MUX9 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Group Select Bits for MUX8: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX8 01 : Select .1 input for MUX8 10 : Select .2 input for MUX8 11 : Select .3 input for MUX8 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Group Select Bits for MUX7: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX7 01 : Select .1 input for MUX7 10 : Select .2 input for MUX7 11 : Select .3 input for MUX7 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Group Select Bits for MUX6: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX6 01 : Select .1 input for MUX6 10 : Select .2 input for MUX6 11 : Select .3 input for MUX6 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Group Select Bits for MUX5: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX5 01 : Select .1 input for MUX5 10 : Select .2 input for MUX5 11 : Select .3 input for MUX5 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Group Select Bits for MUX4: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX4 01 : Select .1 input for MUX4 10 : Select .2 input for MUX4 11 : Select .3 input for MUX4 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-75. AUXSIG4MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX3	R/W	0h	Group Select Bits for MUX3: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX3 01 : Select .1 input for MUX3 10 : Select .2 input for MUX3 11 : Select .3 input for MUX3 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Group Select Bits for MUX2: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX2 01 : Select .1 input for MUX2 10 : Select .2 input for MUX2 11 : Select .3 input for MUX2 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX1	R/W	0h	Group Select Bits for MUX1: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX1 01 : Select .1 input for MUX1 10 : Select .2 input for MUX1 11 : Select .3 input for MUX1 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Group Select Bits for MUX0: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX0 01 : Select .1 input for MUX0 10 : Select .2 input for MUX0 11 : Select .3 input for MUX0 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.10 AUXSIG4MUX16TO31CFG Register (Offset = 12h) [Reset = 0h]

AUXSIG4MUX16TO31CFG is shown in Figure 17-67 and described in Table 17-76.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-4

Figure 17-67. AUXSIG4MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-76. AUXSIG4MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Group Select Bits for MUX31: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX31 01 : Select .1 input for MUX31 10 : Select .2 input for MUX31 11 : Select .3 input for MUX31 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Group Select Bits for MUX30: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX30 01 : Select .1 input for MUX30 10 : Select .2 input for MUX30 11 : Select .3 input for MUX30 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Group Select Bits for MUX29: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX29 01 : Select .1 input for MUX29 10 : Select .2 input for MUX29 11 : Select .3 input for MUX29 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Group Select Bits for MUX28: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX28 01 : Select .1 input for MUX28 10 : Select .2 input for MUX28 11 : Select .3 input for MUX28 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Group Select Bits for MUX27: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX27 01 : Select .1 input for MUX27 10 : Select .2 input for MUX27 11 : Select .3 input for MUX27 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-76. AUXSIG4MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX26	R/W	0h	Group Select Bits for MUX26: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX26 01 : Select .1 input for MUX26 10 : Select .2 input for MUX26 11 : Select .3 input for MUX26 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX25	R/W	0h	Group Select Bits for MUX25: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX25 01 : Select .1 input for MUX25 10 : Select .2 input for MUX25 11 : Select .3 input for MUX25 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Group Select Bits for MUX24: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX24 01 : Select .1 input for MUX24 10 : Select .2 input for MUX24 11 : Select .3 input for MUX24 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Group Select Bits for MUX23: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX23 01 : Select .1 input for MUX23 10 : Select .2 input for MUX23 11 : Select .3 input for MUX23 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Group Select Bits for MUX22: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX22 01 : Select .1 input for MUX22 10 : Select .2 input for MUX22 11 : Select .3 input for MUX22 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Group Select Bits for MUX21: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX21 01 : Select .1 input for MUX21 10 : Select .2 input for MUX21 11 : Select .3 input for MUX21 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Group Select Bits for MUX20: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX20 01 : Select .1 input for MUX20 10 : Select .2 input for MUX20 11 : Select .3 input for MUX20 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-76. AUXSIG4MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX19	R/W	0h	Group Select Bits for MUX19: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX19 01 : Select .1 input for MUX19 10 : Select .2 input for MUX19 11 : Select .3 input for MUX19 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Group Select Bits for MUX18: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX18 01 : Select .1 input for MUX18 10 : Select .2 input for MUX18 11 : Select .3 input for MUX18 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX17	R/W	0h	Group Select Bits for MUX17: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX17 01 : Select .1 input for MUX17 10 : Select .2 input for MUX17 11 : Select .3 input for MUX17 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Group Select Bits for MUX16: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX16 01 : Select .1 input for MUX16 10 : Select .2 input for MUX16 11 : Select .3 input for MUX16 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.11 AUXSIG5MUX0TO15CFG Register (Offset = 14h) [Reset = 0h]

AUXSIG5MUX0TO15CFG is shown in Figure 17-68 and described in Table 17-77.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-5

Figure 17-68. AUXSIG5MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-77. AUXSIG5MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Group Select Bits for MUX15: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX15 01 : Select .1 input for MUX15 10 : Select .2 input for MUX15 11 : Select .3 input for MUX15 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Group Select Bits for MUX14: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX14 01 : Select .1 input for MUX14 10 : Select .2 input for MUX14 11 : Select .3 input for MUX14 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Group Select Bits for MUX13: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX13 01 : Select .1 input for MUX13 10 : Select .2 input for MUX13 11 : Select .3 input for MUX13 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Group Select Bits for MUX12: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX12 01 : Select .1 input for MUX12 10 : Select .2 input for MUX12 11 : Select .3 input for MUX12 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Group Select Bits for MUX11: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX11 01 : Select .1 input for MUX11 10 : Select .2 input for MUX11 11 : Select .3 input for MUX11 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-77. AUXSIG5MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX10	R/W	0h	Group Select Bits for MUX10: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX10 01 : Select .1 input for MUX10 10 : Select .2 input for MUX10 11 : Select .3 input for MUX10 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX9	R/W	0h	Group Select Bits for MUX9: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX9 01 : Select .1 input for MUX9 10 : Select .2 input for MUX9 11 : Select .3 input for MUX9 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Group Select Bits for MUX8: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX8 01 : Select .1 input for MUX8 10 : Select .2 input for MUX8 11 : Select .3 input for MUX8 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Group Select Bits for MUX7: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX7 01 : Select .1 input for MUX7 10 : Select .2 input for MUX7 11 : Select .3 input for MUX7 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Group Select Bits for MUX6: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX6 01 : Select .1 input for MUX6 10 : Select .2 input for MUX6 11 : Select .3 input for MUX6 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Group Select Bits for MUX5: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX5 01 : Select .1 input for MUX5 10 : Select .2 input for MUX5 11 : Select .3 input for MUX5 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Group Select Bits for MUX4: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX4 01 : Select .1 input for MUX4 10 : Select .2 input for MUX4 11 : Select .3 input for MUX4 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-77. AUXSIG5MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX3	R/W	0h	Group Select Bits for MUX3: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX3 01 : Select .1 input for MUX3 10 : Select .2 input for MUX3 11 : Select .3 input for MUX3 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Group Select Bits for MUX2: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX2 01 : Select .1 input for MUX2 10 : Select .2 input for MUX2 11 : Select .3 input for MUX2 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX1	R/W	0h	Group Select Bits for MUX1: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX1 01 : Select .1 input for MUX1 10 : Select .2 input for MUX1 11 : Select .3 input for MUX1 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Group Select Bits for MUX0: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX0 01 : Select .1 input for MUX0 10 : Select .2 input for MUX0 11 : Select .3 input for MUX0 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.12 AUXSIG5MUX16TO31CFG Register (Offset = 16h) [Reset = 0h]

AUXSIG5MUX16TO31CFG is shown in Figure 17-69 and described in Table 17-78.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-5

Figure 17-69. AUXSIG5MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-78. AUXSIG5MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Group Select Bits for MUX31: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX31 01 : Select .1 input for MUX31 10 : Select .2 input for MUX31 11 : Select .3 input for MUX31 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Group Select Bits for MUX30: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX30 01 : Select .1 input for MUX30 10 : Select .2 input for MUX30 11 : Select .3 input for MUX30 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Group Select Bits for MUX29: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX29 01 : Select .1 input for MUX29 10 : Select .2 input for MUX29 11 : Select .3 input for MUX29 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Group Select Bits for MUX28: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX28 01 : Select .1 input for MUX28 10 : Select .2 input for MUX28 11 : Select .3 input for MUX28 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Group Select Bits for MUX27: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX27 01 : Select .1 input for MUX27 10 : Select .2 input for MUX27 11 : Select .3 input for MUX27 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-78. AUXSIG5MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX26	R/W	0h	Group Select Bits for MUX26: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX26 01 : Select .1 input for MUX26 10 : Select .2 input for MUX26 11 : Select .3 input for MUX26 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX25	R/W	0h	Group Select Bits for MUX25: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX25 01 : Select .1 input for MUX25 10 : Select .2 input for MUX25 11 : Select .3 input for MUX25 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Group Select Bits for MUX24: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX24 01 : Select .1 input for MUX24 10 : Select .2 input for MUX24 11 : Select .3 input for MUX24 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Group Select Bits for MUX23: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX23 01 : Select .1 input for MUX23 10 : Select .2 input for MUX23 11 : Select .3 input for MUX23 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Group Select Bits for MUX22: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX22 01 : Select .1 input for MUX22 10 : Select .2 input for MUX22 11 : Select .3 input for MUX22 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Group Select Bits for MUX21: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX21 01 : Select .1 input for MUX21 10 : Select .2 input for MUX21 11 : Select .3 input for MUX21 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Group Select Bits for MUX20: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX20 01 : Select .1 input for MUX20 10 : Select .2 input for MUX20 11 : Select .3 input for MUX20 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-78. AUXSIG5MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX19	R/W	0h	Group Select Bits for MUX19: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX19 01 : Select .1 input for MUX19 10 : Select .2 input for MUX19 11 : Select .3 input for MUX19 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Group Select Bits for MUX18: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX18 01 : Select .1 input for MUX18 10 : Select .2 input for MUX18 11 : Select .3 input for MUX18 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX17	R/W	0h	Group Select Bits for MUX17: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX17 01 : Select .1 input for MUX17 10 : Select .2 input for MUX17 11 : Select .3 input for MUX17 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Group Select Bits for MUX16: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX16 01 : Select .1 input for MUX16 10 : Select .2 input for MUX16 11 : Select .3 input for MUX16 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.13 AUXSIG6MUX0TO15CFG Register (Offset = 18h) [Reset = 0h]

AUXSIG6MUX0TO15CFG is shown in Figure 17-70 and described in Table 17-79.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-6

Figure 17-70. AUXSIG6MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-79. AUXSIG6MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Group Select Bits for MUX15: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX15 01 : Select .1 input for MUX15 10 : Select .2 input for MUX15 11 : Select .3 input for MUX15 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Group Select Bits for MUX14: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX14 01 : Select .1 input for MUX14 10 : Select .2 input for MUX14 11 : Select .3 input for MUX14 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Group Select Bits for MUX13: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX13 01 : Select .1 input for MUX13 10 : Select .2 input for MUX13 11 : Select .3 input for MUX13 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Group Select Bits for MUX12: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX12 01 : Select .1 input for MUX12 10 : Select .2 input for MUX12 11 : Select .3 input for MUX12 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Group Select Bits for MUX11: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX11 01 : Select .1 input for MUX11 10 : Select .2 input for MUX11 11 : Select .3 input for MUX11 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-79. AUXSIG6MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX10	R/W	0h	Group Select Bits for MUX10: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX10 01 : Select .1 input for MUX10 10 : Select .2 input for MUX10 11 : Select .3 input for MUX10 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX9	R/W	0h	Group Select Bits for MUX9: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX9 01 : Select .1 input for MUX9 10 : Select .2 input for MUX9 11 : Select .3 input for MUX9 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Group Select Bits for MUX8: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX8 01 : Select .1 input for MUX8 10 : Select .2 input for MUX8 11 : Select .3 input for MUX8 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Group Select Bits for MUX7: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX7 01 : Select .1 input for MUX7 10 : Select .2 input for MUX7 11 : Select .3 input for MUX7 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Group Select Bits for MUX6: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX6 01 : Select .1 input for MUX6 10 : Select .2 input for MUX6 11 : Select .3 input for MUX6 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Group Select Bits for MUX5: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX5 01 : Select .1 input for MUX5 10 : Select .2 input for MUX5 11 : Select .3 input for MUX5 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Group Select Bits for MUX4: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX4 01 : Select .1 input for MUX4 10 : Select .2 input for MUX4 11 : Select .3 input for MUX4 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-79. AUXSIG6MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX3	R/W	0h	Group Select Bits for MUX3: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX3 01 : Select .1 input for MUX3 10 : Select .2 input for MUX3 11 : Select .3 input for MUX3 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Group Select Bits for MUX2: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX2 01 : Select .1 input for MUX2 10 : Select .2 input for MUX2 11 : Select .3 input for MUX2 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX1	R/W	0h	Group Select Bits for MUX1: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX1 01 : Select .1 input for MUX1 10 : Select .2 input for MUX1 11 : Select .3 input for MUX1 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Group Select Bits for MUX0: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX0 01 : Select .1 input for MUX0 10 : Select .2 input for MUX0 11 : Select .3 input for MUX0 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.14 AUXSIG6MUX16TO31CFG Register (Offset = 1Ah) [Reset = 0h]

AUXSIG6MUX16TO31CFG is shown in Figure 17-71 and described in Table 17-80.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-6

Figure 17-71. AUXSIG6MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-80. AUXSIG6MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Group Select Bits for MUX31: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX31 01 : Select .1 input for MUX31 10 : Select .2 input for MUX31 11 : Select .3 input for MUX31 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Group Select Bits for MUX30: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX30 01 : Select .1 input for MUX30 10 : Select .2 input for MUX30 11 : Select .3 input for MUX30 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Group Select Bits for MUX29: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX29 01 : Select .1 input for MUX29 10 : Select .2 input for MUX29 11 : Select .3 input for MUX29 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Group Select Bits for MUX28: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX28 01 : Select .1 input for MUX28 10 : Select .2 input for MUX28 11 : Select .3 input for MUX28 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Group Select Bits for MUX27: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX27 01 : Select .1 input for MUX27 10 : Select .2 input for MUX27 11 : Select .3 input for MUX27 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-80. AUXSIG6MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX26	R/W	0h	Group Select Bits for MUX26: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX26 01 : Select .1 input for MUX26 10 : Select .2 input for MUX26 11 : Select .3 input for MUX26 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX25	R/W	0h	Group Select Bits for MUX25: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX25 01 : Select .1 input for MUX25 10 : Select .2 input for MUX25 11 : Select .3 input for MUX25 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Group Select Bits for MUX24: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX24 01 : Select .1 input for MUX24 10 : Select .2 input for MUX24 11 : Select .3 input for MUX24 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Group Select Bits for MUX23: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX23 01 : Select .1 input for MUX23 10 : Select .2 input for MUX23 11 : Select .3 input for MUX23 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Group Select Bits for MUX22: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX22 01 : Select .1 input for MUX22 10 : Select .2 input for MUX22 11 : Select .3 input for MUX22 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Group Select Bits for MUX21: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX21 01 : Select .1 input for MUX21 10 : Select .2 input for MUX21 11 : Select .3 input for MUX21 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Group Select Bits for MUX20: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX20 01 : Select .1 input for MUX20 10 : Select .2 input for MUX20 11 : Select .3 input for MUX20 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-80. AUXSIG6MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX19	R/W	0h	Group Select Bits for MUX19: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX19 01 : Select .1 input for MUX19 10 : Select .2 input for MUX19 11 : Select .3 input for MUX19 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Group Select Bits for MUX18: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX18 01 : Select .1 input for MUX18 10 : Select .2 input for MUX18 11 : Select .3 input for MUX18 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX17	R/W	0h	Group Select Bits for MUX17: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX17 01 : Select .1 input for MUX17 10 : Select .2 input for MUX17 11 : Select .3 input for MUX17 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Group Select Bits for MUX16: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX16 01 : Select .1 input for MUX16 10 : Select .2 input for MUX16 11 : Select .3 input for MUX16 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.15 AUXSIG7MUX0TO15CFG Register (Offset = 1Ch) [Reset = 0h]

AUXSIG7MUX0TO15CFG is shown in Figure 17-72 and described in Table 17-81.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-7

Figure 17-72. AUXSIG7MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-81. AUXSIG7MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Group Select Bits for MUX15: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX15 01 : Select .1 input for MUX15 10 : Select .2 input for MUX15 11 : Select .3 input for MUX15 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Group Select Bits for MUX14: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX14 01 : Select .1 input for MUX14 10 : Select .2 input for MUX14 11 : Select .3 input for MUX14 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Group Select Bits for MUX13: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX13 01 : Select .1 input for MUX13 10 : Select .2 input for MUX13 11 : Select .3 input for MUX13 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Group Select Bits for MUX12: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX12 01 : Select .1 input for MUX12 10 : Select .2 input for MUX12 11 : Select .3 input for MUX12 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Group Select Bits for MUX11: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX11 01 : Select .1 input for MUX11 10 : Select .2 input for MUX11 11 : Select .3 input for MUX11 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-81. AUXSIG7MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX10	R/W	0h	Group Select Bits for MUX10: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX10 01 : Select .1 input for MUX10 10 : Select .2 input for MUX10 11 : Select .3 input for MUX10 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX9	R/W	0h	Group Select Bits for MUX9: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX9 01 : Select .1 input for MUX9 10 : Select .2 input for MUX9 11 : Select .3 input for MUX9 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Group Select Bits for MUX8: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX8 01 : Select .1 input for MUX8 10 : Select .2 input for MUX8 11 : Select .3 input for MUX8 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Group Select Bits for MUX7: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX7 01 : Select .1 input for MUX7 10 : Select .2 input for MUX7 11 : Select .3 input for MUX7 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Group Select Bits for MUX6: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX6 01 : Select .1 input for MUX6 10 : Select .2 input for MUX6 11 : Select .3 input for MUX6 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Group Select Bits for MUX5: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX5 01 : Select .1 input for MUX5 10 : Select .2 input for MUX5 11 : Select .3 input for MUX5 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Group Select Bits for MUX4: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX4 01 : Select .1 input for MUX4 10 : Select .2 input for MUX4 11 : Select .3 input for MUX4 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-81. AUXSIG7MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX3	R/W	0h	Group Select Bits for MUX3: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX3 01 : Select .1 input for MUX3 10 : Select .2 input for MUX3 11 : Select .3 input for MUX3 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Group Select Bits for MUX2: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX2 01 : Select .1 input for MUX2 10 : Select .2 input for MUX2 11 : Select .3 input for MUX2 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX1	R/W	0h	Group Select Bits for MUX1: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX1 01 : Select .1 input for MUX1 10 : Select .2 input for MUX1 11 : Select .3 input for MUX1 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Group Select Bits for MUX0: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX0 01 : Select .1 input for MUX0 10 : Select .2 input for MUX0 11 : Select .3 input for MUX0 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.16 AUXSIG7MUX16TO31CFG Register (Offset = 1Eh) [Reset = 0h]

AUXSIG7MUX16TO31CFG is shown in Figure 17-73 and described in Table 17-82.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-7

Figure 17-73. AUXSIG7MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-82. AUXSIG7MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Group Select Bits for MUX31: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX31 01 : Select .1 input for MUX31 10 : Select .2 input for MUX31 11 : Select .3 input for MUX31 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Group Select Bits for MUX30: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX30 01 : Select .1 input for MUX30 10 : Select .2 input for MUX30 11 : Select .3 input for MUX30 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Group Select Bits for MUX29: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX29 01 : Select .1 input for MUX29 10 : Select .2 input for MUX29 11 : Select .3 input for MUX29 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Group Select Bits for MUX28: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX28 01 : Select .1 input for MUX28 10 : Select .2 input for MUX28 11 : Select .3 input for MUX28 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Group Select Bits for MUX27: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX27 01 : Select .1 input for MUX27 10 : Select .2 input for MUX27 11 : Select .3 input for MUX27 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-82. AUXSIG7MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX26	R/W	0h	Group Select Bits for MUX26: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX26 01 : Select .1 input for MUX26 10 : Select .2 input for MUX26 11 : Select .3 input for MUX26 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX25	R/W	0h	Group Select Bits for MUX25: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX25 01 : Select .1 input for MUX25 10 : Select .2 input for MUX25 11 : Select .3 input for MUX25 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Group Select Bits for MUX24: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX24 01 : Select .1 input for MUX24 10 : Select .2 input for MUX24 11 : Select .3 input for MUX24 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Group Select Bits for MUX23: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX23 01 : Select .1 input for MUX23 10 : Select .2 input for MUX23 11 : Select .3 input for MUX23 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Group Select Bits for MUX22: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX22 01 : Select .1 input for MUX22 10 : Select .2 input for MUX22 11 : Select .3 input for MUX22 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Group Select Bits for MUX21: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX21 01 : Select .1 input for MUX21 10 : Select .2 input for MUX21 11 : Select .3 input for MUX21 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Group Select Bits for MUX20: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX20 01 : Select .1 input for MUX20 10 : Select .2 input for MUX20 11 : Select .3 input for MUX20 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-82. AUXSIG7MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX19	R/W	0h	Group Select Bits for MUX19: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX19 01 : Select .1 input for MUX19 10 : Select .2 input for MUX19 11 : Select .3 input for MUX19 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Group Select Bits for MUX18: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX18 01 : Select .1 input for MUX18 10 : Select .2 input for MUX18 11 : Select .3 input for MUX18 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX17	R/W	0h	Group Select Bits for MUX17: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX17 01 : Select .1 input for MUX17 10 : Select .2 input for MUX17 11 : Select .3 input for MUX17 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Group Select Bits for MUX16: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX16 01 : Select .1 input for MUX16 10 : Select .2 input for MUX16 11 : Select .3 input for MUX16 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.17 AUXSIG0MUXENABLE Register (Offset = 20h) [Reset = 0h]

AUXSIG0MUXENABLE is shown in Figure 17-74 and described in Table 17-83.

Return to the Summary Table.

CLB XBAR Mux Enable Register for Output-0

Figure 17-74. AUXSIG0MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-83. AUXSIG0MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of MUX31 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX31 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX31 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of MUX30 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX30 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX30 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of MUX29 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX29 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX29 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of MUX28 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX28 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX28 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of MUX27 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX27 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX27 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-83. AUXSIG0MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of MUX26 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX26 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX26 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of MUX25 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX25 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX25 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of MUX24 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX24 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX24 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of MUX23 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX23 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX23 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of MUX22 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX22 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX22 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of MUX21 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX21 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX21 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of MUX20 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX20 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX20 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of MUX19 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX19 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX19 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-83. AUXSIG0MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of MUX18 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX18 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX18 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of MUX17 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX17 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX17 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of MUX16 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX16 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX16 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of MUX15 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX15 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX15 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of MUX14 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX14 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX14 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of MUX13 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX13 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX13 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of MUX12 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX12 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX12 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of MUX11 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX11 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX11 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-83. AUXSIG0MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of MUX10 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX10 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX10 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of MUX9 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX9 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX9 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of MUX8 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX8 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX8 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of MUX7 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX7 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX7 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of MUX6 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX6 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX6 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of MUX5 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX5 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX5 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of MUX4 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX4 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX4 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of MUX3 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX3 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX3 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-83. AUXSIG0MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of MUX2 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX2 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX2 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of MUX1 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX1 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX1 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX0 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX0 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.18 AUXSIG1MUXENABLE Register (Offset = 22h) [Reset = 0h]

AUXSIG1MUXENABLE is shown in Figure 17-75 and described in Table 17-84.

Return to the Summary Table.

CLB XBAR Mux Enable Register for Output-1

Figure 17-75. AUXSIG1MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-84. AUXSIG1MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of MUX31 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX31 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX31 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of MUX30 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX30 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX30 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of MUX29 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX29 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX29 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of MUX28 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX28 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX28 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of MUX27 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX27 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX27 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-84. AUXSIG1MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of MUX26 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX26 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX26 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of MUX25 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX25 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX25 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of MUX24 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX24 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX24 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of MUX23 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX23 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX23 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of MUX22 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX22 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX22 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of MUX21 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX21 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX21 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of MUX20 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX20 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX20 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of MUX19 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX19 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX19 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-84. AUXSIG1MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of MUX18 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX18 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX18 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of MUX17 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX17 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX17 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of MUX16 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX16 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX16 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of MUX15 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX15 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX15 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of MUX14 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX14 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX14 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of MUX13 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX13 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX13 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of MUX12 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX12 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX12 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of MUX11 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX11 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX11 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-84. AUXSIG1MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of MUX10 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX10 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX10 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of MUX9 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX9 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX9 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of MUX8 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX8 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX8 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of MUX7 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX7 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX7 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of MUX6 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX6 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX6 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of MUX5 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX5 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX5 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of MUX4 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX4 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX4 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of MUX3 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX3 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX3 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-84. AUXSIG1MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of MUX2 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX2 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX2 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of MUX1 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX1 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX1 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX0 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX0 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.19 AUXSIG2MUXENABLE Register (Offset = 24h) [Reset = 0h]

AUXSIG2MUXENABLE is shown in Figure 17-76 and described in Table 17-85.

Return to the Summary Table.

CLB XBAR Mux Enable Register for Output-2

Figure 17-76. AUXSIG2MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-85. AUXSIG2MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of MUX31 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX31 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX31 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of MUX30 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX30 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX30 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of MUX29 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX29 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX29 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of MUX28 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX28 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX28 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of MUX27 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX27 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX27 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-85. AUXSIG2MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of MUX26 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX26 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX26 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of MUX25 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX25 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX25 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of MUX24 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX24 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX24 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of MUX23 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX23 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX23 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of MUX22 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX22 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX22 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of MUX21 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX21 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX21 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of MUX20 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX20 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX20 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of MUX19 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX19 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX19 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-85. AUXSIG2MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of MUX18 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX18 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX18 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of MUX17 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX17 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX17 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of MUX16 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX16 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX16 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of MUX15 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX15 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX15 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of MUX14 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX14 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX14 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of MUX13 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX13 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX13 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of MUX12 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX12 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX12 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of MUX11 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX11 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX11 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-85. AUXSIG2MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of MUX10 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX10 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX10 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of MUX9 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX9 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX9 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of MUX8 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX8 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX8 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of MUX7 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX7 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX7 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of MUX6 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX6 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX6 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of MUX5 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX5 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX5 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of MUX4 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX4 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX4 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of MUX3 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX3 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX3 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-85. AUXSIG2MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of MUX2 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX2 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX2 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of MUX1 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX1 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX1 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX0 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX0 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.20 AUXSIG3MUXENABLE Register (Offset = 26h) [Reset = 0h]

AUXSIG3MUXENABLE is shown in Figure 17-77 and described in Table 17-86.

Return to the Summary Table.

CLB XBAR Mux Enable Register for Output-3

Figure 17-77. AUXSIG3MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-86. AUXSIG3MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of MUX31 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX31 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX31 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of MUX30 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX30 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX30 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of MUX29 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX29 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX29 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of MUX28 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX28 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX28 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of MUX27 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX27 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX27 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-86. AUXSIG3MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of MUX26 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX26 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX26 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of MUX25 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX25 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX25 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of MUX24 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX24 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX24 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of MUX23 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX23 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX23 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of MUX22 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX22 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX22 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of MUX21 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX21 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX21 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of MUX20 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX20 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX20 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of MUX19 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX19 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX19 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-86. AUXSIG3MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of MUX18 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX18 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX18 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of MUX17 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX17 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX17 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of MUX16 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX16 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX16 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of MUX15 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX15 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX15 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of MUX14 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX14 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX14 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of MUX13 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX13 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX13 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of MUX12 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX12 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX12 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of MUX11 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX11 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX11 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-86. AUXSIG3MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of MUX10 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX10 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX10 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of MUX9 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX9 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX9 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of MUX8 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX8 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX8 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of MUX7 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX7 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX7 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of MUX6 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX6 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX6 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of MUX5 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX5 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX5 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of MUX4 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX4 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX4 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of MUX3 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX3 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX3 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-86. AUXSIG3MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of MUX2 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX2 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX2 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of MUX1 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX1 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX1 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX0 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX0 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.21 AUXSIG4MUXENABLE Register (Offset = 28h) [Reset = 0h]

AUXSIG4MUXENABLE is shown in Figure 17-78 and described in Table 17-87.

Return to the Summary Table.

CLB XBAR Mux Enable Register for Output-4

Figure 17-78. AUXSIG4MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-87. AUXSIG4MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of MUX31 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX31 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX31 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of MUX30 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX30 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX30 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of MUX29 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX29 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX29 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of MUX28 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX28 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX28 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of MUX27 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX27 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX27 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-87. AUXSIG4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of MUX26 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX26 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX26 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of MUX25 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX25 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX25 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of MUX24 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX24 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX24 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of MUX23 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX23 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX23 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of MUX22 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX22 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX22 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of MUX21 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX21 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX21 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of MUX20 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX20 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX20 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of MUX19 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX19 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX19 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-87. AUXSIG4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of MUX18 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX18 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX18 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of MUX17 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX17 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX17 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of MUX16 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX16 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX16 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of MUX15 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX15 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX15 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of MUX14 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX14 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX14 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of MUX13 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX13 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX13 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of MUX12 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX12 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX12 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of MUX11 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX11 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX11 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-87. AUXSIG4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of MUX10 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX10 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX10 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of MUX9 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX9 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX9 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of MUX8 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX8 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX8 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of MUX7 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX7 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX7 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of MUX6 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX6 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX6 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of MUX5 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX5 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX5 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of MUX4 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX4 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX4 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of MUX3 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX3 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX3 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-87. AUXSIG4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of MUX2 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX2 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX2 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of MUX1 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX1 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX1 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX0 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX0 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.22 AUXSIG5MUXENABLE Register (Offset = 2Ah) [Reset = 0h]

AUXSIG5MUXENABLE is shown in Figure 17-79 and described in Table 17-88.

Return to the Summary Table.

CLB XBAR Mux Enable Register for Output-5

Figure 17-79. AUXSIG5MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-88. AUXSIG5MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of MUX31 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX31 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX31 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of MUX30 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX30 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX30 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of MUX29 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX29 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX29 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of MUX28 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX28 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX28 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of MUX27 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX27 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX27 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-88. AUXSIG5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of MUX26 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX26 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX26 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of MUX25 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX25 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX25 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of MUX24 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX24 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX24 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of MUX23 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX23 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX23 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of MUX22 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX22 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX22 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of MUX21 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX21 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX21 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of MUX20 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX20 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX20 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of MUX19 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX19 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX19 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-88. AUXSIG5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of MUX18 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX18 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX18 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of MUX17 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX17 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX17 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of MUX16 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX16 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX16 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of MUX15 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX15 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX15 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of MUX14 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX14 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX14 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of MUX13 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX13 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX13 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of MUX12 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX12 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX12 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of MUX11 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX11 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX11 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-88. AUXSIG5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of MUX10 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX10 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX10 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of MUX9 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX9 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX9 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of MUX8 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX8 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX8 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of MUX7 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX7 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX7 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of MUX6 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX6 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX6 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of MUX5 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX5 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX5 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of MUX4 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX4 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX4 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of MUX3 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX3 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX3 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-88. AUXSIG5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of MUX2 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX2 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX2 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of MUX1 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX1 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX1 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX0 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX0 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.23 AUXSIG6MUXENABLE Register (Offset = 2Ch) [Reset = 0h]

AUXSIG6MUXENABLE is shown in Figure 17-80 and described in Table 17-89.

Return to the Summary Table.

CLB XBAR Mux Enable Register for Output-6

Figure 17-80. AUXSIG6MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-89. AUXSIG6MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of MUX31 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX31 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX31 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of MUX30 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX30 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX30 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of MUX29 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX29 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX29 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of MUX28 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX28 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX28 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of MUX27 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX27 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX27 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-89. AUXSIG6MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of MUX26 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX26 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX26 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of MUX25 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX25 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX25 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of MUX24 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX24 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX24 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of MUX23 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX23 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX23 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of MUX22 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX22 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX22 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of MUX21 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX21 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX21 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of MUX20 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX20 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX20 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of MUX19 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX19 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX19 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-89. AUXSIG6MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of MUX18 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX18 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX18 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of MUX17 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX17 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX17 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of MUX16 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX16 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX16 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of MUX15 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX15 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX15 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of MUX14 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX14 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX14 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of MUX13 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX13 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX13 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of MUX12 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX12 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX12 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of MUX11 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX11 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX11 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-89. AUXSIG6MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of MUX10 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX10 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX10 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of MUX9 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX9 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX9 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of MUX8 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX8 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX8 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of MUX7 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX7 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX7 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of MUX6 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX6 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX6 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of MUX5 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX5 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX5 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of MUX4 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX4 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX4 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of MUX3 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX3 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX3 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-89. AUXSIG6MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of MUX2 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX2 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX2 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of MUX1 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX1 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX1 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX0 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX0 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.24 AUXSIG7MUXENABLE Register (Offset = 2Eh) [Reset = 0h]

AUXSIG7MUXENABLE is shown in Figure 17-81 and described in Table 17-90.

Return to the Summary Table.

CLB XBAR Mux Enable Register for Output-7

Figure 17-81. AUXSIG7MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-90. AUXSIG7MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of MUX31 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX31 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX31 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of MUX30 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX30 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX30 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of MUX29 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX29 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX29 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of MUX28 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX28 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX28 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of MUX27 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX27 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX27 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-90. AUXSIG7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of MUX26 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX26 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX26 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of MUX25 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX25 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX25 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of MUX24 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX24 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX24 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of MUX23 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX23 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX23 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of MUX22 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX22 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX22 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of MUX21 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX21 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX21 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of MUX20 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX20 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX20 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of MUX19 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX19 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX19 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-90. AUXSIG7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of MUX18 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX18 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX18 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of MUX17 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX17 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX17 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of MUX16 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX16 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX16 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of MUX15 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX15 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX15 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of MUX14 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX14 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX14 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of MUX13 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX13 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX13 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of MUX12 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX12 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX12 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of MUX11 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX11 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX11 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-90. AUXSIG7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of MUX10 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX10 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX10 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of MUX9 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX9 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX9 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of MUX8 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX8 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX8 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of MUX7 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX7 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX7 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of MUX6 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX6 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX6 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of MUX5 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX5 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX5 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of MUX4 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX4 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX4 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of MUX3 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX3 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX3 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-90. AUXSIG7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of MUX2 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX2 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX2 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of MUX1 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX1 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX1 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX0 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX0 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.25 AUXSIGOUTINV Register (Offset = 38h) [Reset = 0h]

AUXSIGOUTINV is shown in Figure 17-82 and described in Table 17-91.

Return to the Summary Table.

CLB XBAR Output Inversion Register

Figure 17-82. AUXSIGOUTINV Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
OUT7	OUT6	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-91. AUXSIGOUTINV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	RESERVED	R-0	0h	Reserved
7	OUT7	R/W	0h	Selects polarity for AUXSIG7 of CLB-XBAR 0: drives active high output 1: drives active-low output Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	OUT6	R/W	0h	Selects polarity for AUXSIG6 of CLB-XBAR 0: drives active high output 1: drives active-low output Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	OUT5	R/W	0h	Selects polarity for AUXSIG5 of CLB-XBAR 0: drives active high output 1: drives active-low output Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	OUT4	R/W	0h	Selects polarity for AUXSIG4 of CLB-XBAR 0: drives active high output 1: drives active-low output Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	OUT3	R/W	0h	Selects polarity for AUXSIG3 of CLB-XBAR 0: drives active high output 1: drives active-low output Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
2	OUT2	R/W	0h	Selects polarity for AUXSIG2 of CLB-XBAR 0: drives active high output 1: drives active-low output Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-91. AUXSIGOUTINV Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	OUT1	R/W	0h	Selects polarity for AUXSIG1 of CLB-XBAR 0: drives active high output 1: drives active-low output Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	OUT0	R/W	0h	Selects polarity for AUXSIG0 of CLB-XBAR 0: drives active high output 1: drives active-low output Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.5.26 AUXSIGLOCK Register (Offset = 3Eh) [Reset = 0h]

AUXSIGLOCK is shown in Figure 17-83 and described in Table 17-92.

Return to the Summary Table.

ClbXbar Configuration Lock register

Figure 17-83. AUXSIGLOCK Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK
R-0-0h							R/WSonce-0h

Table 17-92. AUXSIGLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Bit-0 of this register can be set only if KEY= 0x5a5a Reset type: CPU1.SYSRSn
15-1	RESERVED	R-0	0h	Reserved
0	LOCK	R/WSonce	0h	Locks the configuration for CLB-XBAR. Once the configuration is locked, writes to the below registers for CLB-XBAR is blocked. Registers Affected by the LOCK mechanism: CLB-XBAROUTyMUX0TO15CFG CLB-XBAROUTyMUX16TO31CFG CLB-XBAROUTyMUXENABLE CLB-XBAROUTLATEN CLB-XBAROUTINV 0: Writes to the above registers are allowed 1: Writes to the above registers are blocked Note: [1] LOCK mechanism only apples to writes. Reads are never blocked. Reset type: CPU1.SYSRSn

17.3.6 OUTPUT_XBAR_REGS Registers

Table 17-93 lists the memory-mapped registers for the OUTPUT_XBAR_REGS registers. All register offset addresses not listed in Table 17-93 should be considered as reserved locations and the register contents should not be modified.

Table 17-93. OUTPUT_XBAR_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	OUTPUT1MUX0TO15CFG	Output X-BAR Mux Configuration for Output 1	EALLOW	Go
2h	OUTPUT1MUX16TO31CFG	Output X-BAR Mux Configuration for Output 1	EALLOW	Go
4h	OUTPUT2MUX0TO15CFG	Output X-BAR Mux Configuration for Output 2	EALLOW	Go
6h	OUTPUT2MUX16TO31CFG	Output X-BAR Mux Configuration for Output 2	EALLOW	Go
8h	OUTPUT3MUX0TO15CFG	Output X-BAR Mux Configuration for Output 3	EALLOW	Go
Ah	OUTPUT3MUX16TO31CFG	Output X-BAR Mux Configuration for Output 3	EALLOW	Go
Ch	OUTPUT4MUX0TO15CFG	Output X-BAR Mux Configuration for Output 4	EALLOW	Go
Eh	OUTPUT4MUX16TO31CFG	Output X-BAR Mux Configuration for Output 4	EALLOW	Go
10h	OUTPUT5MUX0TO15CFG	Output X-BAR Mux Configuration for Output 5	EALLOW	Go
12h	OUTPUT5MUX16TO31CFG	Output X-BAR Mux Configuration for Output 5	EALLOW	Go
14h	OUTPUT6MUX0TO15CFG	Output X-BAR Mux Configuration for Output 6	EALLOW	Go
16h	OUTPUT6MUX16TO31CFG	Output X-BAR Mux Configuration for Output 6	EALLOW	Go
18h	OUTPUT7MUX0TO15CFG	Output X-BAR Mux Configuration for Output 7	EALLOW	Go
1Ah	OUTPUT7MUX16TO31CFG	Output X-BAR Mux Configuration for Output 7	EALLOW	Go
1Ch	OUTPUT8MUX0TO15CFG	Output X-BAR Mux Configuration for Output 8	EALLOW	Go
1Eh	OUTPUT8MUX16TO31CFG	Output X-BAR Mux Configuration for Output 8	EALLOW	Go
20h	OUTPUT1MUXENABLE	Output X-BAR Mux Enable for Output 1	EALLOW	Go
22h	OUTPUT2MUXENABLE	Output X-BAR Mux Enable for Output 2	EALLOW	Go
24h	OUTPUT3MUXENABLE	Output X-BAR Mux Enable for Output 3	EALLOW	Go
26h	OUTPUT4MUXENABLE	Output X-BAR Mux Enable for Output 4	EALLOW	Go
28h	OUTPUT5MUXENABLE	Output X-BAR Mux Enable for Output 5	EALLOW	Go
2Ah	OUTPUT6MUXENABLE	Output X-BAR Mux Enable for Output 6	EALLOW	Go
2Ch	OUTPUT7MUXENABLE	Output X-BAR Mux Enable for Output 7	EALLOW	Go
2Eh	OUTPUT8MUXENABLE	Output X-BAR Mux Enable for Output 8	EALLOW	Go
30h	OUTPUTLATCH	Output X-BAR Output Latch		Go
32h	OUTPUTLATCHCLR	Output X-BAR Output Latch Clear		Go
34h	OUTPUTLATCHFRC	Output X-BAR Output Latch Clear		Go
36h	OUTPUTLATCHENABLE	Output X-BAR Output Latch Enable	EALLOW	Go
38h	OUTPUTINV	Output X-BAR Output Inversion	EALLOW	Go
3Eh	OUTPUTLOCK	Output X-BAR Configuration Lock register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 17-94 shows the codes that are used for access types in this section.

Table 17-94. OUTPUT_XBAR_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		

Table 17-94. OUTPUT_XBAR_REGS Access Type Codes (continued)

Access Type	Code	Description
W	W	Write
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

17.3.6.1 OUTPUT1MUX0TO15CFG Register (Offset = 0h) [Reset = 0h]

OUTPUT1MUX0TO15CFG is shown in Figure 17-84 and described in Table 17-95.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 1

Figure 17-84. OUTPUT1MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-95. OUTPUT1MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for OUTPUT1 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for OUTPUT1 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for OUTPUT1 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for OUTPUT1 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for OUTPUT1 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for OUTPUT1 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-95. OUTPUT1MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for OUTPUT1 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for OUTPUT1 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for OUTPUT1 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for OUTPUT1 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for OUTPUT1 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for OUTPUT1 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for OUTPUT1 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for OUTPUT1 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-95. OUTPUT1MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for OUTPUT1 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for OUTPUT1 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.2 OUTPUT1MUX16TO31CFG Register (Offset = 2h) [Reset = 0h]

OUTPUT1MUX16TO31CFG is shown in Figure 17-85 and described in Table 17-96.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 1

Figure 17-85. OUTPUT1MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-96. OUTPUT1MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for OUTPUT1 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for OUTPUT1 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for OUTPUT1 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for OUTPUT1 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for OUTPUT1 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for OUTPUT1 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-96. OUTPUT1MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for OUTPUT1 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for OUTPUT1 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for OUTPUT1 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for OUTPUT1 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for OUTPUT1 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for OUTPUT1 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for OUTPUT1 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for OUTPUT1 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-96. OUTPUT1MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for OUTPUT1 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for OUTPUT1 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.3 OUTPUT2MUX0TO15CFG Register (Offset = 4h) [Reset = 0h]

OUTPUT2MUX0TO15CFG is shown in Figure 17-86 and described in Table 17-97.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 2

Figure 17-86. OUTPUT2MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-97. OUTPUT2MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for OUTPUT2 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for OUTPUT2 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for OUTPUT2 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for OUTPUT2 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for OUTPUT2 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for OUTPUT2 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-97. OUTPUT2MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for OUTPUT2 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for OUTPUT2 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for OUTPUT2 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for OUTPUT2 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for OUTPUT2 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for OUTPUT2 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for OUTPUT2 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for OUTPUT2 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-97. OUTPUT2MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for OUTPUT2 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for OUTPUT2 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.4 OUTPUT2MUX16TO31CFG Register (Offset = 6h) [Reset = 0h]

OUTPUT2MUX16TO31CFG is shown in Figure 17-87 and described in Table 17-98.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 2

Figure 17-87. OUTPUT2MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-98. OUTPUT2MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for OUTPUT2 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for OUTPUT2 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for OUTPUT2 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for OUTPUT2 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for OUTPUT2 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for OUTPUT2 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-98. OUTPUT2MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for OUTPUT2 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for OUTPUT2 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for OUTPUT2 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for OUTPUT2 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for OUTPUT2 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for OUTPUT2 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for OUTPUT2 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for OUTPUT2 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-98. OUTPUT2MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for OUTPUT2 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for OUTPUT2 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.5 OUTPUT3MUX0TO15CFG Register (Offset = 8h) [Reset = 0h]

OUTPUT3MUX0TO15CFG is shown in Figure 17-88 and described in Table 17-99.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 3

Figure 17-88. OUTPUT3MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-99. OUTPUT3MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for OUTPUT3 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for OUTPUT3 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for OUTPUT3 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for OUTPUT3 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for OUTPUT3 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for OUTPUT3 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-99. OUTPUT3MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for OUTPUT3 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for OUTPUT3 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for OUTPUT3 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for OUTPUT3 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for OUTPUT3 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for OUTPUT3 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for OUTPUT3 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for OUTPUT3 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-99. OUTPUT3MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for OUTPUT3 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for OUTPUT3 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.6 OUTPUT3MUX16TO31CFG Register (Offset = Ah) [Reset = 0h]

OUTPUT3MUX16TO31CFG is shown in Figure 17-89 and described in Table 17-100.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 3

Figure 17-89. OUTPUT3MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-100. OUTPUT3MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for OUTPUT3 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for OUTPUT3 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for OUTPUT3 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for OUTPUT3 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for OUTPUT3 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for OUTPUT3 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-100. OUTPUT3MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for OUTPUT3 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for OUTPUT3 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for OUTPUT3 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for OUTPUT3 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for OUTPUT3 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for OUTPUT3 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for OUTPUT3 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for OUTPUT3 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-100. OUTPUT3MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for OUTPUT3 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for OUTPUT3 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.7 OUTPUT4MUX0TO15CFG Register (Offset = Ch) [Reset = 0h]

OUTPUT4MUX0TO15CFG is shown in Figure 17-90 and described in Table 17-101.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 4

Figure 17-90. OUTPUT4MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-101. OUTPUT4MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for OUTPUT4 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for OUTPUT4 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for OUTPUT4 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for OUTPUT4 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for OUTPUT4 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for OUTPUT4 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-101. OUTPUT4MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for OUTPUT4 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for OUTPUT4 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for OUTPUT4 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for OUTPUT4 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for OUTPUT4 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for OUTPUT4 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for OUTPUT4 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for OUTPUT4 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-101. OUTPUT4MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for OUTPUT4 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for OUTPUT4 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.8 OUTPUT4MUX16TO31CFG Register (Offset = Eh) [Reset = 0h]

OUTPUT4MUX16TO31CFG is shown in Figure 17-91 and described in Table 17-102.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 4

Figure 17-91. OUTPUT4MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-102. OUTPUT4MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for OUTPUT4 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for OUTPUT4 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for OUTPUT4 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for OUTPUT4 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for OUTPUT4 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for OUTPUT4 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-102. OUTPUT4MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for OUTPUT4 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for OUTPUT4 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for OUTPUT4 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for OUTPUT4 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for OUTPUT4 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for OUTPUT4 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for OUTPUT4 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for OUTPUT4 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-102. OUTPUT4MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for OUTPUT4 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for OUTPUT4 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.9 OUTPUT5MUX0TO15CFG Register (Offset = 10h) [Reset = 0h]

OUTPUT5MUX0TO15CFG is shown in Figure 17-92 and described in Table 17-103.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 5

Figure 17-92. OUTPUT5MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-103. OUTPUT5MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for OUTPUT5 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for OUTPUT5 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for OUTPUT5 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for OUTPUT5 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for OUTPUT5 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for OUTPUT5 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-103. OUTPUT5MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for OUTPUT5 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for OUTPUT5 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for OUTPUT5 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for OUTPUT5 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for OUTPUT5 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for OUTPUT5 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for OUTPUT5 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for OUTPUT5 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-103. OUTPUT5MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for OUTPUT5 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for OUTPUT5 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.10 OUTPUT5MUX16TO31CFG Register (Offset = 12h) [Reset = 0h]

OUTPUT5MUX16TO31CFG is shown in Figure 17-93 and described in Table 17-104.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 5

Figure 17-93. OUTPUT5MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-104. OUTPUT5MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for OUTPUT5 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for OUTPUT5 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for OUTPUT5 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for OUTPUT5 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for OUTPUT5 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for OUTPUT5 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-104. OUTPUT5MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for OUTPUT5 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for OUTPUT5 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for OUTPUT5 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for OUTPUT5 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for OUTPUT5 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for OUTPUT5 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for OUTPUT5 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for OUTPUT5 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-104. OUTPUT5MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for OUTPUT5 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for OUTPUT5 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.11 OUTPUT6MUX0TO15CFG Register (Offset = 14h) [Reset = 0h]

OUTPUT6MUX0TO15CFG is shown in Figure 17-94 and described in Table 17-105.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 6

Figure 17-94. OUTPUT6MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-105. OUTPUT6MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for OUTPUT6 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for OUTPUT6 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for OUTPUT6 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for OUTPUT6 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for OUTPUT6 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for OUTPUT6 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-105. OUTPUT6MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for OUTPUT6 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for OUTPUT6 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for OUTPUT6 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for OUTPUT6 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for OUTPUT6 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for OUTPUT6 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for OUTPUT6 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for OUTPUT6 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-105. OUTPUT6MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for OUTPUT6 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for OUTPUT6 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.12 OUTPUT6MUX16TO31CFG Register (Offset = 16h) [Reset = 0h]

OUTPUT6MUX16TO31CFG is shown in Figure 17-95 and described in Table 17-106.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 6

Figure 17-95. OUTPUT6MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-106. OUTPUT6MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for OUTPUT6 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for OUTPUT6 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for OUTPUT6 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for OUTPUT6 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for OUTPUT6 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for OUTPUT6 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-106. OUTPUT6MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for OUTPUT6 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for OUTPUT6 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for OUTPUT6 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for OUTPUT6 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for OUTPUT6 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for OUTPUT6 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for OUTPUT6 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for OUTPUT6 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-106. OUTPUT6MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for OUTPUT6 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for OUTPUT6 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.13 OUTPUT7MUX0TO15CFG Register (Offset = 18h) [Reset = 0h]

OUTPUT7MUX0TO15CFG is shown in Figure 17-96 and described in Table 17-107.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 7

Figure 17-96. OUTPUT7MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-107. OUTPUT7MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for OUTPUT7 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for OUTPUT7 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for OUTPUT7 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for OUTPUT7 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for OUTPUT7 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for OUTPUT7 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-107. OUTPUT7MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for OUTPUT7 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for OUTPUT7 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for OUTPUT7 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for OUTPUT7 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for OUTPUT7 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for OUTPUT7 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for OUTPUT7 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for OUTPUT7 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-107. OUTPUT7MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for OUTPUT7 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for OUTPUT7 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.14 OUTPUT7MUX16TO31CFG Register (Offset = 1Ah) [Reset = 0h]

OUTPUT7MUX16TO31CFG is shown in Figure 17-97 and described in Table 17-108.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 7

Figure 17-97. OUTPUT7MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-108. OUTPUT7MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for OUTPUT7 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for OUTPUT7 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for OUTPUT7 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for OUTPUT7 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for OUTPUT7 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for OUTPUT7 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-108. OUTPUT7MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for OUTPUT7 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for OUTPUT7 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for OUTPUT7 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for OUTPUT7 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for OUTPUT7 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for OUTPUT7 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for OUTPUT7 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for OUTPUT7 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-108. OUTPUT7MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for OUTPUT7 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for OUTPUT7 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.15 OUTPUT8MUX0TO15CFG Register (Offset = 1Ch) [Reset = 0h]

OUTPUT8MUX0TO15CFG is shown in Figure 17-98 and described in Table 17-109.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 8

Figure 17-98. OUTPUT8MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-109. OUTPUT8MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for OUTPUT8 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for OUTPUT8 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for OUTPUT8 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for OUTPUT8 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for OUTPUT8 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for OUTPUT8 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-109. OUTPUT8MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for OUTPUT8 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for OUTPUT8 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for OUTPUT8 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for OUTPUT8 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for OUTPUT8 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for OUTPUT8 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for OUTPUT8 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for OUTPUT8 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-109. OUTPUT8MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for OUTPUT8 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for OUTPUT8 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.16 OUTPUT8MUX16TO31CFG Register (Offset = 1Eh) [Reset = 0h]

OUTPUT8MUX16TO31CFG is shown in Figure 17-99 and described in Table 17-110.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 8

Figure 17-99. OUTPUT8MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 17-110. OUTPUT8MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for OUTPUT8 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for OUTPUT8 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for OUTPUT8 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for OUTPUT8 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for OUTPUT8 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for OUTPUT8 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-110. OUTPUT8MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for OUTPUT8 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for OUTPUT8 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for OUTPUT8 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for OUTPUT8 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for OUTPUT8 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for OUTPUT8 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for OUTPUT8 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for OUTPUT8 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-110. OUTPUT8MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for OUTPUT8 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for OUTPUT8 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.17 OUTPUT1MUXENABLE Register (Offset = 20h) [Reset = 0h]

OUTPUT1MUXENABLE is shown in Figure 17-100 and described in Table 17-111.

Return to the Summary Table.

Output X-BAR Mux Enable for Output 1

Figure 17-100. OUTPUT1MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-111. OUTPUT1MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux31 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux31 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux30 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux30 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux29 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux29 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux28 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux28 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux27 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux27 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-111. OUTPUT1MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux26 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux26 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux25 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux25 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux24 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux24 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux23 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux23 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux22 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux22 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux21 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux21 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux20 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux20 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux19 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux19 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-111. OUTPUT1MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux18 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux18 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux17 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux17 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux16 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux16 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux15 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux15 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux14 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux14 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux13 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux13 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux12 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux12 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux11 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux11 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-111. OUTPUT1MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux10 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux10 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux9 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux9 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux8 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux8 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux7 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux7 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux6 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux6 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux5 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux5 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux4 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux4 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux3 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux3 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-111. OUTPUT1MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux2 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux2 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux1 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux1 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux0 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux0 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.18 OUTPUT2MUXENABLE Register (Offset = 22h) [Reset = 0h]

OUTPUT2MUXENABLE is shown in Figure 17-101 and described in Table 17-112.

Return to the Summary Table.

Output X-BAR Mux Enable for Output 2

Figure 17-101. OUTPUT2MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-112. OUTPUT2MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux31 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux31 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux30 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux30 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux29 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux29 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux28 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux28 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux27 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux27 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-112. OUTPUT2MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux26 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux26 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux25 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux25 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux24 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux24 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux23 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux23 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux22 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux22 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux21 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux21 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux20 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux20 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux19 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux19 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-112. OUTPUT2MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux18 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux18 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux17 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux17 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux16 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux16 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux15 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux15 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux14 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux14 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux13 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux13 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux12 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux12 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux11 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux11 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-112. OUTPUT2MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux10 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux10 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux9 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux9 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux8 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux8 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux7 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux7 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux6 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux6 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux5 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux5 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux4 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux4 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux3 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux3 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-112. OUTPUT2MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux2 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux2 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux1 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux1 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux0 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux0 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.19 OUTPUT3MUXENABLE Register (Offset = 24h) [Reset = 0h]

OUTPUT3MUXENABLE is shown in Figure 17-102 and described in Table 17-113.

Return to the Summary Table.

Output X-BAR Mux Enable for Output 3

Figure 17-102. OUTPUT3MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-113. OUTPUT3MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux31 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux31 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux30 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux30 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux29 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux29 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux28 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux28 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux27 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux27 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-113. OUTPUT3MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux26 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux26 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux25 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux25 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux24 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux24 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux23 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux23 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux22 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux22 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux21 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux21 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux20 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux20 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux19 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux19 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-113. OUTPUT3MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux18 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux18 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux17 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux17 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux16 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux16 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux15 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux15 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux14 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux14 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux13 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux13 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux12 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux12 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux11 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux11 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-113. OUTPUT3MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux10 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux10 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux9 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux9 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux8 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux8 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux7 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux7 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux6 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux6 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux5 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux5 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux4 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux4 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux3 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux3 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-113. OUTPUT3MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux2 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux2 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux1 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux1 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux0 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux0 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.20 OUTPUT4MUXENABLE Register (Offset = 26h) [Reset = 0h]

OUTPUT4MUXENABLE is shown in Figure 17-103 and described in Table 17-114.

Return to the Summary Table.

Output X-BAR Mux Enable for Output 4

Figure 17-103. OUTPUT4MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-114. OUTPUT4MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux31 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux31 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux30 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux30 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux29 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux29 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux28 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux28 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux27 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux27 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-114. OUTPUT4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux26 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux26 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux25 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux25 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux24 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux24 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux23 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux23 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux22 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux22 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux21 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux21 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux20 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux20 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux19 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux19 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-114. OUTPUT4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux18 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux18 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux17 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux17 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux16 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux16 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux15 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux15 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux14 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux14 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux13 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux13 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux12 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux12 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux11 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux11 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-114. OUTPUT4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux10 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux10 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux9 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux9 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux8 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux8 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux7 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux7 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux6 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux6 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux5 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux5 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux4 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux4 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux3 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux3 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-114. OUTPUT4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux2 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux2 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux1 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux1 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux0 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux0 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.21 OUTPUT5MUXENABLE Register (Offset = 28h) [Reset = 0h]

OUTPUT5MUXENABLE is shown in Figure 17-104 and described in Table 17-115.

Return to the Summary Table.

Output X-BAR Mux Enable for Output 5

Figure 17-104. OUTPUT5MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-115. OUTPUT5MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux31 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux31 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux30 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux30 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux29 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux29 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux28 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux28 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux27 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux27 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-115. OUTPUT5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux26 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux26 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux25 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux25 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux24 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux24 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux23 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux23 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux22 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux22 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux21 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux21 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux20 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux20 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux19 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux19 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-115. OUTPUT5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux18 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux18 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux17 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux17 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux16 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux16 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux15 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux15 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux14 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux14 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux13 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux13 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux12 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux12 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux11 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux11 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-115. OUTPUT5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux10 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux10 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux9 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux9 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux8 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux8 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux7 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux7 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux6 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux6 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux5 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux5 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux4 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux4 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux3 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux3 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-115. OUTPUT5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux2 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux2 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux1 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux1 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux0 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux0 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.22 OUTPUT6MUXENABLE Register (Offset = 2Ah) [Reset = 0h]

OUTPUT6MUXENABLE is shown in Figure 17-105 and described in Table 17-116.

Return to the Summary Table.

Output X-BAR Mux Enable for Output 6

Figure 17-105. OUTPUT6MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-116. OUTPUT6MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux31 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux31 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux30 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux30 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux29 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux29 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux28 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux28 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux27 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux27 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-116. OUTPUT6MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux26 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux26 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux25 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux25 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux24 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux24 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux23 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux23 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux22 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux22 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux21 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux21 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux20 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux20 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux19 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux19 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-116. OUTPUT6MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux18 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux18 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux17 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux17 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux16 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux16 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux15 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux15 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux14 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux14 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux13 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux13 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux12 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux12 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux11 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux11 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-116. OUTPUT6MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux10 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux10 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux9 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux9 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux8 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux8 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux7 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux7 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux6 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux6 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux5 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux5 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux4 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux4 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux3 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux3 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-116. OUTPUT6MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux2 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux2 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux1 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux1 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux0 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux0 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.23 OUTPUT7MUXENABLE Register (Offset = 2Ch) [Reset = 0h]

OUTPUT7MUXENABLE is shown in Figure 17-106 and described in Table 17-117.

Return to the Summary Table.

Output X-BAR Mux Enable for Output 7

Figure 17-106. OUTPUT7MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-117. OUTPUT7MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux31 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux31 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux30 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux30 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux29 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux29 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux28 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux28 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux27 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux27 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-117. OUTPUT7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux26 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux26 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux25 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux25 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux24 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux24 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux23 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux23 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux22 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux22 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux21 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux21 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux20 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux20 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux19 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux19 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-117. OUTPUT7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux18 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux18 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux17 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux17 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux16 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux16 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux15 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux15 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux14 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux14 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux13 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux13 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux12 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux12 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux11 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux11 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-117. OUTPUT7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux10 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux10 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux9 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux9 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux8 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux8 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux7 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux7 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux6 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux6 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux5 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux5 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux4 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux4 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux3 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux3 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-117. OUTPUT7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux2 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux2 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux1 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux1 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux0 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux0 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.24 OUTPUT8MUXENABLE Register (Offset = 2Eh) [Reset = 0h]

OUTPUT8MUXENABLE is shown in Figure 17-107 and described in Table 17-118.

Return to the Summary Table.

Output X-BAR Mux Enable for Output 8

Figure 17-107. OUTPUT8MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-118. OUTPUT8MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux31 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux31 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux30 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux30 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux29 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux29 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux28 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux28 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux27 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux27 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-118. OUTPUT8MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux26 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux26 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux25 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux25 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux24 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux24 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux23 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux23 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux22 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux22 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux21 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux21 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux20 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux20 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux19 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux19 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-118. OUTPUT8MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux18 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux18 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux17 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux17 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux16 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux16 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux15 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux15 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux14 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux14 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux13 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux13 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux12 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux12 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux11 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux11 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-118. OUTPUT8MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux10 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux10 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux9 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux9 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux8 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux8 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux7 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux7 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux6 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux6 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux5 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux5 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux4 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux4 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux3 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux3 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-118. OUTPUT8MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux2 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux2 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux1 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux1 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux0 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux0 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.25 OUTPUTLATCH Register (Offset = 30h) [Reset = 0h]

OUTPUTLATCH is shown in Figure 17-108 and described in Table 17-119.

Return to the Summary Table.

Output X-BAR Output Latch

Figure 17-108. OUTPUTLATCH Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
OUTPUT8	OUTPUT7	OUTPUT6	OUTPUT5	OUTPUT4	OUTPUT3	OUTPUT2	OUTPUT1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 17-119. OUTPUTLATCH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	RESERVED	R-0	0h	Reserved
7	OUTPUT8	R	0h	Records the OUTPUT8 of OUTPUT-XBAR. 0: Respective output has not been triggered 1: Respective output is triggered Refer to the Output X-BAR section of this chapter for more details. Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
6	OUTPUT7	R	0h	Records the OUTPUT7 of OUTPUT-XBAR. 0: Respective output has not been triggered 1: Respective output is triggered Refer to the Output X-BAR section of this chapter for more details. Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
5	OUTPUT6	R	0h	Records the OUTPUT6 of OUTPUT-XBAR. 0: Respective output has not been triggered 1: Respective output is triggered Refer to the Output X-BAR section of this chapter for more details. Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
4	OUTPUT5	R	0h	Records the OUTPUT5 of OUTPUT-XBAR. 0: Respective output has not been triggered 1: Respective output is triggered Refer to the Output X-BAR section of this chapter for more details. Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 17-119. OUTPUTLATCH Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	OUTPUT4	R	0h	Records the OUTPUT4 of OUTPUT-XBAR. 0: Respective output has not been triggered 1: Respective output is triggered Refer to the Output X-BAR section of this chapter for more details. Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
2	OUTPUT3	R	0h	Records the OUTPUT3 of OUTPUT-XBAR. 0: Respective output has not been triggered 1: Respective output is triggered Refer to the Output X-BAR section of this chapter for more details. Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
1	OUTPUT2	R	0h	Records the OUTPUT2 of OUTPUT-XBAR. 0: Respective output has not been triggered 1: Respective output is triggered Refer to the Output X-BAR section of this chapter for more details. Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
0	OUTPUT1	R	0h	Records the OUTPUT1 of OUTPUT-XBAR. 0: Respective output has not been triggered 1: Respective output is triggered Refer to the Output X-BAR section of this chapter for more details. Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

17.3.6.26 OUTPUTLATCHCLR Register (Offset = 32h) [Reset = 0h]

OUTPUTLATCHCLR is shown in Figure 17-109 and described in Table 17-120.

Return to the Summary Table.

Output X-BAR Output Latch Clear

Figure 17-109. OUTPUTLATCHCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
OUTPUT8	OUTPUT7	OUTPUT6	OUTPUT5	OUTPUT4	OUTPUT3	OUTPUT2	OUTPUT1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 17-120. OUTPUTLATCHCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	RESERVED	R-0	0h	Reserved
7	OUTPUT8	R-0/W1S	0h	Clears the Output-Latch for OUTPUT8 of OUTPUT-XBAR Writing 1 clears the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	OUTPUT7	R-0/W1S	0h	Clears the Output-Latch for OUTPUT7 of OUTPUT-XBAR Writing 1 clears the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	OUTPUT6	R-0/W1S	0h	Clears the Output-Latch for OUTPUT6 of OUTPUT-XBAR Writing 1 clears the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	OUTPUT5	R-0/W1S	0h	Clears the Output-Latch for OUTPUT5 of OUTPUT-XBAR Writing 1 clears the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	OUTPUT4	R-0/W1S	0h	Clears the Output-Latch for OUTPUT4 of OUTPUT-XBAR Writing 1 clears the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-120. OUTPUTLATCHCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	OUTPUT3	R-0/W1S	0h	Clears the Output-Latch for OUTPUT3 of OUTPUT-XBAR Writing 1 clears the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	OUTPUT2	R-0/W1S	0h	Clears the Output-Latch for OUTPUT2 of OUTPUT-XBAR Writing 1 clears the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	OUTPUT1	R-0/W1S	0h	Clears the Output-Latch for OUTPUT1 of OUTPUT-XBAR Writing 1 clears the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.27 OUTPUTLATCHFRC Register (Offset = 34h) [Reset = 0h]

OUTPUTLATCHFRC is shown in Figure 17-110 and described in Table 17-121.

Return to the Summary Table.

Output X-BAR Output Latch Clear

Figure 17-110. OUTPUTLATCHFRC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
OUTPUT8	OUTPUT7	OUTPUT6	OUTPUT5	OUTPUT4	OUTPUT3	OUTPUT2	OUTPUT1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 17-121. OUTPUTLATCHFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	RESERVED	R-0	0h	Reserved
7	OUTPUT8	R-0/W1S	0h	Sets the Output-Latch for OUTPUT8 of OUTPUT-XBAR Writing 1 sets the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	OUTPUT7	R-0/W1S	0h	Sets the Output-Latch for OUTPUT7 of OUTPUT-XBAR Writing 1 sets the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	OUTPUT6	R-0/W1S	0h	Sets the Output-Latch for OUTPUT6 of OUTPUT-XBAR Writing 1 sets the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	OUTPUT5	R-0/W1S	0h	Sets the Output-Latch for OUTPUT5 of OUTPUT-XBAR Writing 1 sets the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	OUTPUT4	R-0/W1S	0h	Sets the Output-Latch for OUTPUT4 of OUTPUT-XBAR Writing 1 sets the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-121. OUTPUTLATCHFRC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	OUTPUT3	R-0/W1S	0h	Sets the Output-Latch for OUTPUT3 of OUTPUT-XBAR Writing 1 sets the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	OUTPUT2	R-0/W1S	0h	Sets the Output-Latch for OUTPUT2 of OUTPUT-XBAR Writing 1 sets the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	OUTPUT1	R-0/W1S	0h	Sets the Output-Latch for OUTPUT1 of OUTPUT-XBAR Writing 1 sets the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.28 OUTPUTLATCHENABLE Register (Offset = 36h) [Reset = 0h]

OUTPUTLATCHENABLE is shown in Figure 17-111 and described in Table 17-122.

Return to the Summary Table.

Output X-BAR Output Latch Enable

Figure 17-111. OUTPUTLATCHENABLE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
OUTPUT8	OUTPUT7	OUTPUT6	OUTPUT5	OUTPUT4	OUTPUT3	OUTPUT2	OUTPUT1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-122. OUTPUTLATCHENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	RESERVED	R-0	0h	Reserved
7	OUTPUT8	R/W	0h	Selects the output latch to drive OUTPUT8 for OUTPUT-XBAR 0: Output Latch is not selected to driven the respective output 1: Output Latch is selected to drive the respective output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	OUTPUT7	R/W	0h	Selects the output latch to drive OUTPUT7 for OUTPUT-XBAR 0: Output Latch is not selected to driven the respective output 1: Output Latch is selected to drive the respective output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	OUTPUT6	R/W	0h	Selects the output latch to drive OUTPUT6 for OUTPUT-XBAR 0: Output Latch is not selected to driven the respective output 1: Output Latch is selected to drive the respective output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	OUTPUT5	R/W	0h	Selects the output latch to drive OUTPUT5 for OUTPUT-XBAR 0: Output Latch is not selected to driven the respective output 1: Output Latch is selected to drive the respective output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	OUTPUT4	R/W	0h	Selects the output latch to drive OUTPUT4 for OUTPUT-XBAR 0: Output Latch is not selected to driven the respective output 1: Output Latch is selected to drive the respective output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
2	OUTPUT3	R/W	0h	Selects the output latch to drive OUTPUT3 for OUTPUT-XBAR 0: Output Latch is not selected to driven the respective output 1: Output Latch is selected to drive the respective output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-122. OUTPUTLATCHENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	OUTPUT2	R/W	0h	Selects the output latch to drive OUTPUT2 for OUTPUT-XBAR 0: Output Latch is not selected to driven the respective output 1: Output Latch is selected to drive the respective output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	OUTPUT1	R/W	0h	Selects the output latch to drive OUTPUT1 for OUTPUT-XBAR 0: Output Latch is not selected to driven the respective output 1: Output Latch is selected to drive the respective output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.29 OUTPUTINV Register (Offset = 38h) [Reset = 0h]

OUTPUTINV is shown in Figure 17-112 and described in Table 17-123.

Return to the Summary Table.

Output X-BAR Output Inversion

Figure 17-112. OUTPUTINV Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
OUTPUT8	OUTPUT7	OUTPUT6	OUTPUT5	OUTPUT4	OUTPUT3	OUTPUT2	OUTPUT1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-123. OUTPUTINV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	RESERVED	R-0	0h	Reserved
7	OUTPUT8	R/W	0h	Selects polarity for OUTPUT8 of OUTPUT-XBAR 0: drives active high output 1: drives active-low output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	OUTPUT7	R/W	0h	Selects polarity for OUTPUT7 of OUTPUT-XBAR 0: drives active high output 1: drives active-low output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	OUTPUT6	R/W	0h	Selects polarity for OUTPUT6 of OUTPUT-XBAR 0: drives active high output 1: drives active-low output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	OUTPUT5	R/W	0h	Selects polarity for OUTPUT5 of OUTPUT-XBAR 0: drives active high output 1: drives active-low output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	OUTPUT4	R/W	0h	Selects polarity for OUTPUT4 of OUTPUT-XBAR 0: drives active high output 1: drives active-low output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
2	OUTPUT3	R/W	0h	Selects polarity for OUTPUT3 of OUTPUT-XBAR 0: drives active high output 1: drives active-low output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 17-123. OUTPUTINV Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	OUTPUT2	R/W	0h	Selects polarity for OUTPUT2 of OUTPUT-XBAR 0: drives active high output 1: drives active-low output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	OUTPUT1	R/W	0h	Selects polarity for OUTPUT1 of OUTPUT-XBAR 0: drives active high output 1: drives active-low output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

17.3.6.30 OUTPUTLOCK Register (Offset = 3Eh) [Reset = 0h]

OUTPUTLOCK is shown in Figure 17-113 and described in Table 17-124.

Return to the Summary Table.

Output X-BAR Configuration Lock register

Figure 17-113. OUTPUTLOCK Register

31	30	29	28	27	26	25	24
KEY							
R-0/W1S-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W1S-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK
R-0-0h							R/WSonce-0h

Table 17-124. OUTPUTLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W1S	0h	Bit-0 of this register can be set only if KEY= 0x5a5a Reset type: CPU1.SYSRSn
15-1	RESERVED	R-0	0h	Reserved
0	LOCK	R/WSonce	0h	Locks the configuration for OUTPUT-XBAR. Once the configuration is locked, writes to the below registers for OUTPUT-XBAR is blocked. Registers Affected by the LOCK mechanism: OUTPUT-XBAROUTyMUX0TO15CFG OUTPUT-XBAROUTyMUX16TO31CFG OUTPUT-XBAROUTyMUXENABLE OUTPUT-XBAROUTLATENABLE OUTPUT-XBAROUTINV 0: Writes to the above registers are allowed 1: Writes to the above registers are blocked Note: [1] LOCK mechanism only apples to writes. Reads are never blocked. Reset type: CPU1.SYSRSn

17.3.7 Register to Driverlib Function Mapping

17.3.7.1 INPUTXBAR Registers to Driverlib Functions

Table 17-125. INPUTXBAR Registers to Driverlib Functions

File	Driverlib Function
INPUT1SELECT	
xbar.h	XBAR_setInputPin
INPUT2SELECT	
-	See INPUT1SELECT
INPUT3SELECT	
-	See INPUT1SELECT
INPUT4SELECT	
-	See INPUT1SELECT
INPUT5SELECT	
-	See INPUT1SELECT
INPUT6SELECT	
-	See INPUT1SELECT
INPUT7SELECT	
-	See INPUT1SELECT
INPUT8SELECT	
-	See INPUT1SELECT
INPUT9SELECT	
-	See INPUT1SELECT
INPUT10SELECT	
-	See INPUT1SELECT
INPUT11SELECT	
-	See INPUT1SELECT
INPUT12SELECT	
-	See INPUT1SELECT
INPUT13SELECT	
-	See INPUT1SELECT
INPUT14SELECT	
-	See INPUT1SELECT
INPUT15SELECT	
-	See INPUT1SELECT
INPUT16SELECT	
-	See INPUT1SELECT
INPUTSELECTLOCK	
xbar.h	XBAR_lockInput

17.3.7.2 XBAR Registers to Driverlib Functions

Table 17-126. XBAR Registers to Driverlib Functions

File	Driverlib Function
FLG1	
xbar.c	XBAR_getInputFlagStatus
FLG2	
xbar.c	XBAR_getInputFlagStatus

Table 17-126. XBAR Registers to Driverlib Functions (continued)

File	Driverlib Function
FLG3	
xbar.c	XBAR_getInputFlagStatus
FLG4	
xbar.c	XBAR_getInputFlagStatus
CLR1	
xbar.c	XBAR_clearInputFlag
CLR2	
xbar.c	XBAR_clearInputFlag
CLR3	
xbar.c	XBAR_clearInputFlag
CLR4	
xbar.c	XBAR_clearInputFlag

17.3.7.3 EPWMXBAR Registers to Driverlib Functions

Table 17-127. EPWMXBAR Registers to Driverlib Functions

File	Driverlib Function
TRIP4MUX0TO15CFG	
xbar.c	XBAR_setEPWMMuxConfig
TRIP4MUX16TO31CFG	
xbar.c	XBAR_setEPWMMuxConfig
TRIP5MUX0TO15CFG	
-	See TRIP4MUX0TO15CFG
TRIP5MUX16TO31CFG	
-	See TRIP4MUX0TO15CFG
TRIP7MUX0TO15CFG	
-	See TRIP4MUX0TO15CFG
TRIP7MUX16TO31CFG	
-	See TRIP4MUX0TO15CFG
TRIP8MUX0TO15CFG	
-	See TRIP4MUX0TO15CFG
TRIP8MUX16TO31CFG	
-	See TRIP4MUX0TO15CFG
TRIP9MUX0TO15CFG	
-	See TRIP4MUX0TO15CFG
TRIP9MUX16TO31CFG	
-	See TRIP4MUX0TO15CFG
TRIP10MUX0TO15CFG	
-	See TRIP4MUX0TO15CFG
TRIP10MUX16TO31CFG	
-	See TRIP4MUX0TO15CFG
TRIP11MUX0TO15CFG	
-	See TRIP4MUX0TO15CFG
TRIP11MUX16TO31CFG	
-	See TRIP4MUX0TO15CFG
TRIP12MUX0TO15CFG	

Table 17-127. EPWMXBAR Registers to Driverlib Functions (continued)

File	Driverlib Function
-	See TRIP4MUX0TO15CFG
TRIP12MUX16TO31CFG	
-	See TRIP4MUX0TO15CFG
TRIP4MUXENABLE	
xbar.h	XBAR_enableEPWMMux
xbar.h	XBAR_disableEPWMMux
TRIP5MUXENABLE	
-	See TRIP4MUXENABLE
TRIP7MUXENABLE	
-	See TRIP4MUXENABLE
TRIP8MUXENABLE	
-	See TRIP4MUXENABLE
TRIP9MUXENABLE	
-	See TRIP4MUXENABLE
TRIP10MUXENABLE	
-	See TRIP4MUXENABLE
TRIP11MUXENABLE	
-	See TRIP4MUXENABLE
TRIP12MUXENABLE	
-	See TRIP4MUXENABLE
TRIPOUTINV	
xbar.h	XBAR_invertEPWMSignal
TRIPLOCK	
xbar.h	XBAR_lockEPWM

17.3.7.4 CLBXBAR Registers to Driverlib Functions

Table 17-128. CLBXBAR Registers to Driverlib Functions

File	Driverlib Function
AUXSIG0MUX0TO15CFG	
xbar.c	XBAR_setCLBMuxConfig
AUXSIG0MUX16TO31CFG	
xbar.c	XBAR_setCLBMuxConfig
AUXSIG1MUX0TO15CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG1MUX16TO31CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG2MUX0TO15CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG2MUX16TO31CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG3MUX0TO15CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG3MUX16TO31CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG4MUX0TO15CFG	

Table 17-128. CLBXBAR Registers to Driverlib Functions (continued)

File	Driverlib Function
-	See AUXSIG0MUX0TO15CFG
AUXSIG4MUX16TO31CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG5MUX0TO15CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG5MUX16TO31CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG6MUX0TO15CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG6MUX16TO31CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG7MUX0TO15CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG7MUX16TO31CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG0MUXENABLE	
xbar.h	XBAR_enableCLBMux
xbar.h	XBAR_disableCLBMux
AUXSIG1MUXENABLE	
-	See AUXSIG0MUXENABLE
AUXSIG2MUXENABLE	
-	See AUXSIG0MUXENABLE
AUXSIG3MUXENABLE	
-	See AUXSIG0MUXENABLE
AUXSIG4MUXENABLE	
-	See AUXSIG0MUXENABLE
AUXSIG5MUXENABLE	
-	See AUXSIG0MUXENABLE
AUXSIG6MUXENABLE	
-	See AUXSIG0MUXENABLE
AUXSIG7MUXENABLE	
-	See AUXSIG0MUXENABLE
AUXSIGOUTINV	
xbar.h	XBAR_invertCLBSignal
AUXSIGLOCK	
-	

17.3.7.5 OUTPUTXBAR Registers to Driverlib Functions

Table 17-129. OUTPUTXBAR Registers to Driverlib Functions

File	Driverlib Function
OUTPUT1MUX0TO15CFG	
xbar.c	XBAR_setOutputMuxConfig
OUTPUT1MUX16TO31CFG	
-	
OUTPUT2MUX0TO15CFG	

Table 17-129. OUTPUTXBAR Registers to Driverlib Functions (continued)

File	Driverlib Function
-	See OUTPUT1MUX0TO15CFG
OUTPUT2MUX16TO31CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT3MUX0TO15CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT3MUX16TO31CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT4MUX0TO15CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT4MUX16TO31CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT5MUX0TO15CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT5MUX16TO31CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT6MUX0TO15CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT6MUX16TO31CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT7MUX0TO15CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT7MUX16TO31CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT8MUX0TO15CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT8MUX16TO31CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT1MUXENABLE	
xbar.h	XBAR_enableOutputMux
xbar.h	XBAR_disableOutputMux
OUTPUT2MUXENABLE	
-	See OUTPUT1MUXENABLE
OUTPUT3MUXENABLE	
-	See OUTPUT1MUXENABLE
OUTPUT4MUXENABLE	
-	See OUTPUT1MUXENABLE
OUTPUT5MUXENABLE	
-	See OUTPUT1MUXENABLE
OUTPUT6MUXENABLE	
-	See OUTPUT1MUXENABLE
OUTPUT7MUXENABLE	
-	See OUTPUT1MUXENABLE
OUTPUT8MUXENABLE	
-	See OUTPUT1MUXENABLE
OUTPUTLATCH	

Table 17-129. OUTPUTXBAR Registers to Driverlib Functions (continued)

File	Driverlib Function
xbar.h	XBAR_setOutputLatchMode
xbar.h	XBAR_getOutputLatchStatus
xbar.h	XBAR_clearOutputLatch
xbar.h	XBAR_forceOutputLatch
OUTPUTLATCHCLR	
xbar.h	XBAR_clearOutputLatch
OUTPUTLATCHFRC	
xbar.h	XBAR_forceOutputLatch
OUTPUTLATCHENABLE	
xbar.h	XBAR_setOutputLatchMode
OUTPUTINV	
xbar.h	XBAR_invertOutputSignal
OUTPUTLOCK	
xbar.h	XBAR_lockOutput

Chapter 18

► ANALOG PERIPHERALS

The following chapters describe the analog peripherals.

18.1 Technical Reference Manual Overview

The block diagram for the F2838x device is shown in Figure 18-1. This Technical Reference Manual is organized into five major sections:

- C28x SYSTEM RESOURCES

  These chapters describe the C28x CPU subsystem, C28x Boot ROM, device configuration, and other system peripherals.

- ANALOG PERIPHERALS

  These chapters describe the Analog-to-Digital Converter (ADC), Buffered Digital-to-Analog Converter (DAC), Comparator Subsystem (CMPSS), and general analog subsystem configuration.

- CONTROL PERIPHERALS

  These chapters describe the Enhanced Capture (eCAP), High Resolution Capture (HRCAP), Enhanced Pulse Width Modulator (ePWM), Enhanced Quadrature Encoder Pulse (eQEP), and Sigma Delta Filter Module (SDFM) peripherals.

- COMMUNICATION PERIPHERALS

  These chapters describe the communication peripherals available to the C28x subsystem such as the I2C, SCI, FSI, McBSP, PMBUS, and SPI. The CAN, EtherCAT, and USB peripherals are also described in this section and can be assigned to the CM subsystem.

- CONNECTIVITY MANAGER (CM)

  These chapters describe the Connectivity Manager (CM) subsystem as well as the AES, GCRC, CM-I2C, CM-UART, SSI, and Ethernet peripherals.

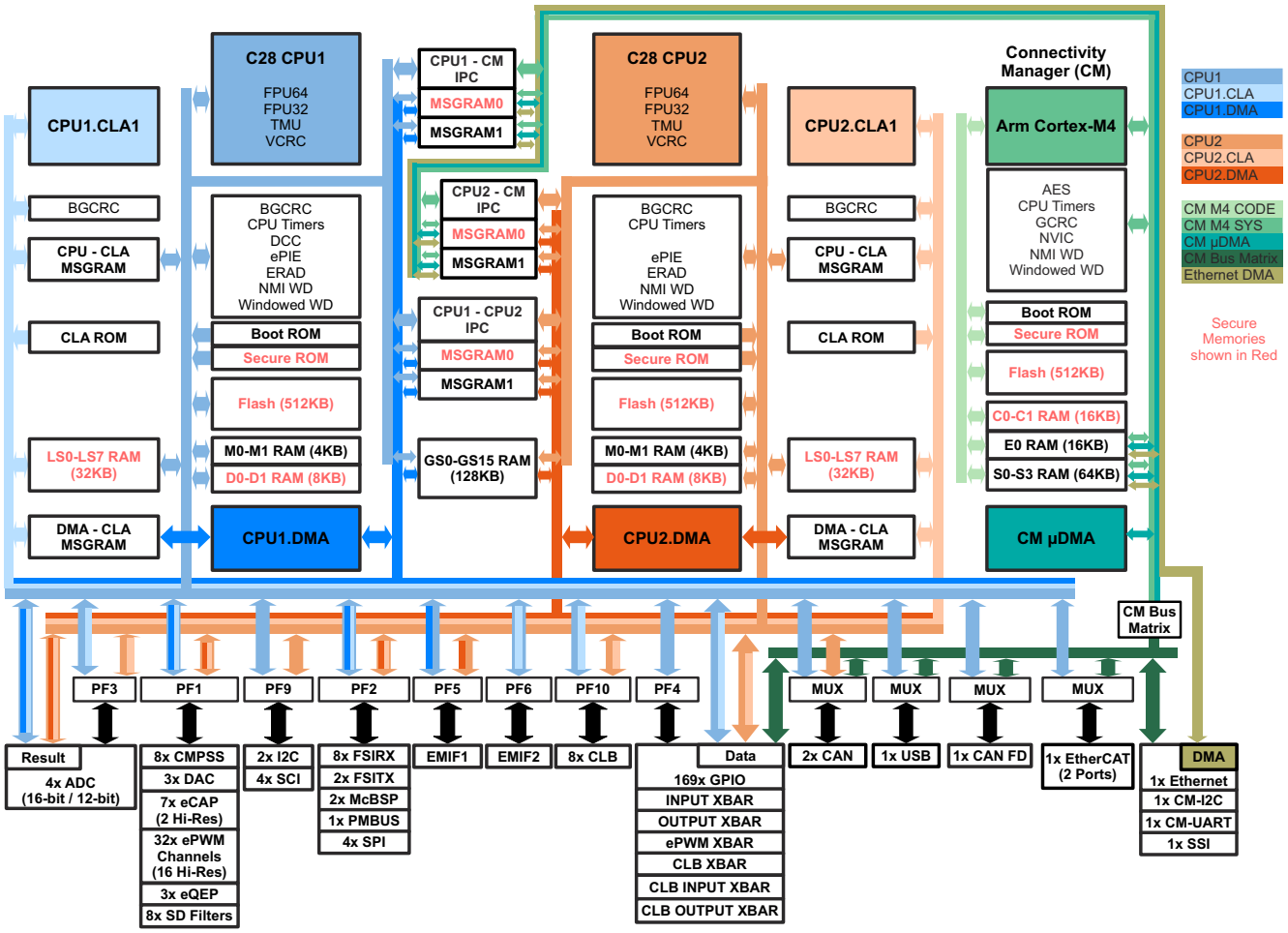


Figure 18-1. F2838x Block Diagram

Chapter 19

Analog Subsystem

The analog subsystem module is described in this chapter.

19.1 Introduction

The analog modules on this device include the Analog-to-Digital Converter (ADC), Temperature Sensor, Buffered Digital-to-Analog Converter (DAC), and Comparator Subsystem (CMPSS).

19.1.1 Features

The analog subsystem has the following features:

- Flexible voltage references
  - The ADCs are referenced to VREFHIx and VREFLOx pins
    - VREFHIx pin voltage must be driven in externally
- The buffered DACs are referenced to VREFHIx and VSSA
  - Alternately, these DACs can be referenced to the VDAC pin and VSSA
- The comparator DACs are referenced to VDDA and VSSA
  - Alternately, these DACs can be referenced to the VDAC pin and VSSA
- Flexible pin usage
  - Buffered DAC and comparator subsystem functions multiplexed with ADC inputs
- Internal connection to VREFLO on all ADCs for offset self-calibration

19.1.2 Block Diagram

The following analog subsystem block diagrams show the connections between the different integrated analog modules to the device pins. These pins fall into two categories: analog module inputs/outputs and reference pins.

The reference pins, VREFHIA to VREFHID and VREFLOA to VREFLOD, can be used to externally supply the reference to the ADC. VREFHIA can also be used to supply the reference voltage to DAC A and DAC B and VREFHIB can be used to supply the reference to DAC C.

The analog module input and outputs are all ADC inputs by default. The pins that connect to CMPSS inputs can be used for the CMPSS without further action and without preventing use as an ADC input simultaneously. DAC outputs must be enabled; this prevents the channel from simultaneously being used as an ADC input (but the ADC can be used to sample the DAC output voltage, if desired).

The VDAC reference pin can be used to set an alternate range for DAC A, DAC B, and DAC C and for the DACs inside the CMPSS modules (the CMPSS DACs are referenced to VDDA and VSSA by default). Using this pin as a reference prevents the channel from being used as an ADC input (but the ADC can be used to sample the VDAC voltage, if desired). The choice of reference is configurable per-module for each CMPSS or buffered DAC, and the selection is made using the module's configuration registers.

The following notes apply to all packages:

- Not all analog pins are available on all devices. See the device data sheet to determine which pins are available.
- See the device data sheet to determine the allowable voltage range for VREFHI and VREFLO.
- An external capacitor is required on the VREFHI pins. See the device data sheet for the specific value required.
- For buffered DAC modules, VSSA is the low reference whether VREFHIx or VDAC is selected as the high reference.
- For CMPSS modules, VSSA is the low reference whether VDAC or VDDA is selected as the high reference.

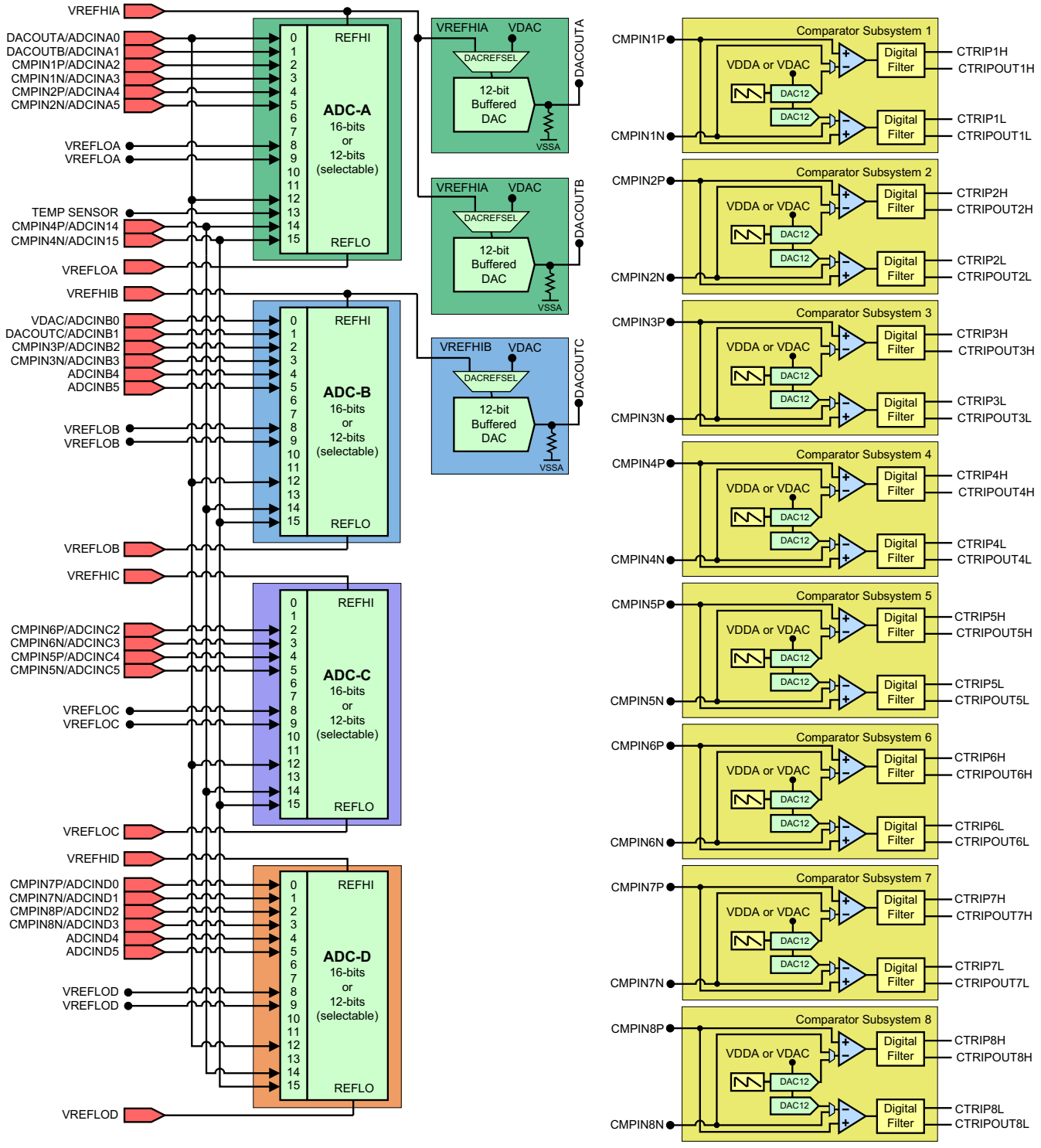


Figure 19-1. Analog Subsystem Block Diagram (337-Ball ZWT)

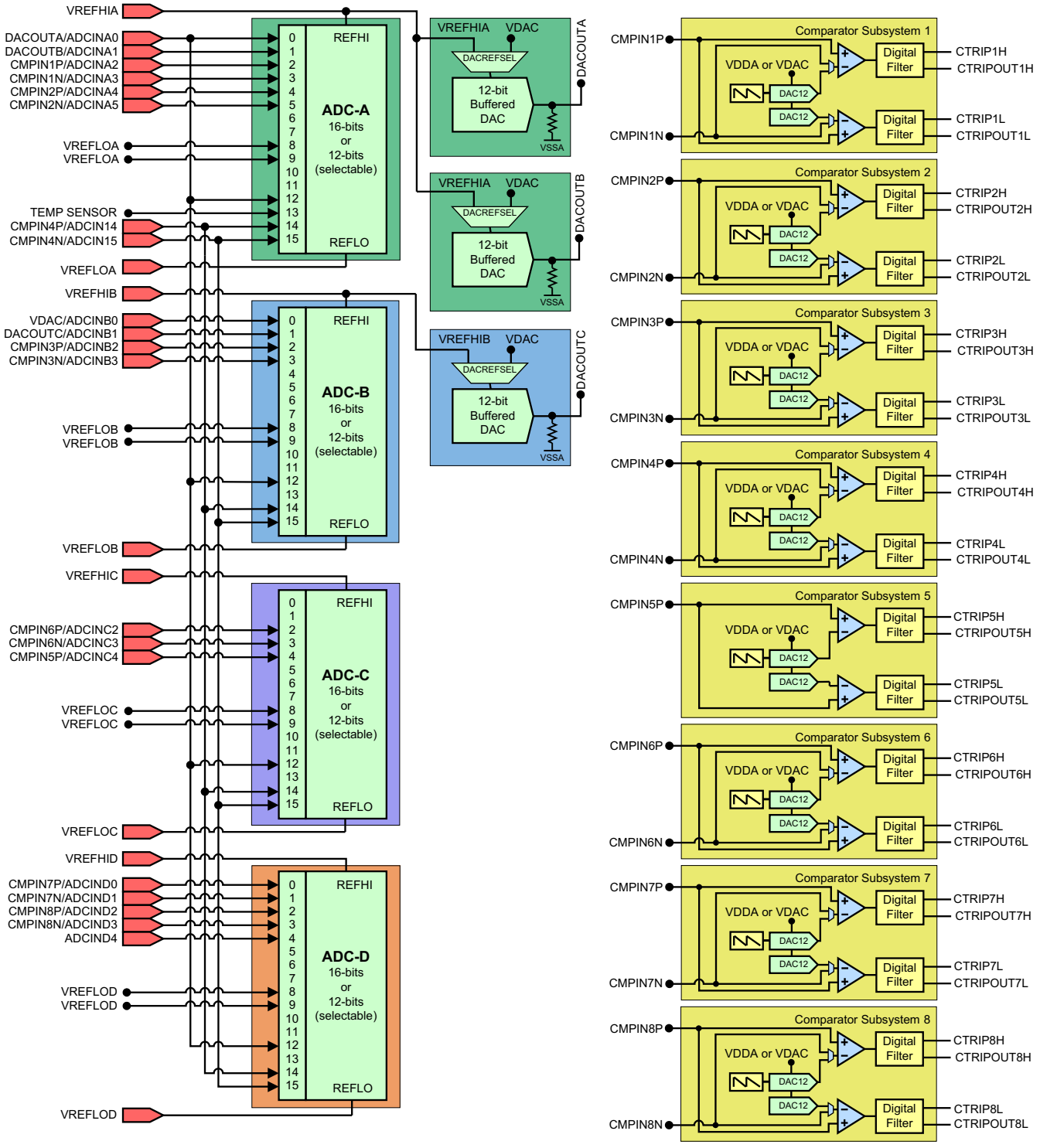


Figure 19-2. Analog Subsystem Block Diagram (176-Pin PTP)

Table 19-1. Analog Signal Descriptions

Signal Name	Description
ADCINAx	ADC A Input
ADCINBx	ADC B Input
ADCINCx	ADC C Input
ADCINDx	ADC D Input
CMPINxP	Comparator subsystem positive input
CMPINxN	Comparator subsystem negative input
DACOUTx	Buffered DAC Output
TEMP SENSOR	Internal temperature sensor
VDAC	Optional external reference voltage for on-chip DACs. There is a 100-pF capacitor to VSSA on this pin whether used for ADC input or DAC reference that cannot be disabled. If this pin is being used as a reference for the on-chip DACs, place at least a 1-µF capacitor on this pin.

Table 19-2. Reference Summary

Module	Reference Option	Configured Where?	Register	Driverlib Function	Notes
ADC	External or Internal	Not Configurable	N/A	N/A	VREFHI must always be driven externally on this device.
Buffered DAC	VREFHI or VDAC	DAC Module	DacxRegs.DACCTL.bit.DACREFSEL	DAC_CTL_DACREFSEL	
	External or Internal	Not Configurable	N/A	N/A	VREFHI must always be driven externally on this device.
CMPSS DACs	VDDA or VDAC	CMPSS Module	CmpssxRegs.COMPDACCTL.bit.SELREF	CMPSS_COMPDACCTL_SELREF	

19.2 Optimizing Power-Up Time

The analog-to-digital converters (ADC) and buffered digital-to-analog converters (DAC) share a common reference circuit. If needed, an application using one or more of these modules can optimize power-up time by taking advantage of the shared reference. Once one of the modules using the shared reference has been initialized in internal reference mode, the power-up time for subsequent modules can be optimized by subtracting the reference power-up time from the minimum power-up time requirement.

For instance, if ADCA requires $t_{ADCPUINT}$ to power up in internal reference mode, and $t_{ADCPUEXT}$ to power up in external reference mode, the application does not need to wait $t_{ADCPUINT}$ to power up a second ADC instance such as ADCC in internal reference mode. In this case, the application can simply wait for $t_{ADCPUEXT}$ after powering up ADCC, even though both ADCs are used in internal reference mode. In the same scenario, if the application wished to use DACA in internal reference mode, the required wait time after power-up is $t_{DACPUEXT}$, not the longer $t_{DACPUINT}$.

There is also a wait time associated with power-up in internal reference mode when switching between 2.5-V and 3.3-V range. See the device data sheet for wait time values.

19.3 Analog Subsystem Registers

This section describes the Analog Subsystem Registers. These registers are only accessible in CPU1.

19.3.1 ASBSYS Base Address Table (C28)

Table 19-3. ASBSYS Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
AnalogSubsysRegs	ANALOG_SUBSYS_REGS	ANALOGSUBSYS_BASE	0x0005_D700	YES	-	-	-	YES

19.3.2 ANALOG_SUBSYS_REGS Registers

Table 19-4 lists the memory-mapped registers for the ANALOG_SUBSYS_REGS registers. All register offset addresses not listed in Table 19-4 should be considered as reserved locations and the register contents should not be modified.

Table 19-4. ANALOG_SUBSYS_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
20h	INTOSC1TRIM	Internal Oscillator 1 Trim Register	EALLOW	Go
22h	INTOSC2TRIM	Internal Oscillator 2 Trim Register	EALLOW	Go
26h	TSNSCTL	Temperature Sensor Control Register	EALLOW	Go
2Eh	LOCK	Lock Register	EALLOW	Go
36h	ANAREFTRIMA	Analog Reference Trim A Register	EALLOW	Go
38h	ANAREFTRIMB	Analog Reference Trim B Register	EALLOW	Go
3Ah	ANAREFTRIMC	Analog Reference Trim C Register	EALLOW	Go
3Ch	ANAREFTRIMD	Analog Reference Trim D Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 19-5 shows the codes that are used for access types in this section.

Table 19-5. ANALOG_SUBSYS_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

19.3.2.1 INTOSC1TRIM Register (Offset = 20h) [Reset = 0h]

INTOSC1TRIM is shown in Figure 19-3 and described in Table 19-6.

Return to the Summary Table.

Internal Oscillator 1 Trim Register

Figure 19-3. INTOSC1TRIM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								RESERVED							
R-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				VALFINETRIM											
R-0h				R/W-0h											

Table 19-6. INTOSC1TRIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	RESERVED	R/W	0h	Reserved
15-12	RESERVED	R	0h	Reserved
11-0	VALFINETRIM	R/W	0h	Oscillator Value Fine Trim Bits. This register should not be modified unless specifically indicated by TI Errata or other documentation. Modifying the contents of this register could cause this module to operate outside of datasheet specifications. Reset type: XRSn

19.3.2.2 INTOSC2TRIM Register (Offset = 22h) [Reset = 0h]

INTOSC2TRIM is shown in Figure 19-4 and described in Table 19-7.

Return to the Summary Table.

Internal Oscillator 2 Trim Register

Figure 19-4. INTOSC2TRIM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								RESERVED							
R-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				VALFINETRIM											
R-0h				R/W-0h											

Table 19-7. INTOSC2TRIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	RESERVED	R/W	0h	Reserved
15-12	RESERVED	R	0h	Reserved
11-0	VALFINETRIM	R/W	0h	Oscillator Value Fine Trim Bits. This register should not be modified unless specifically indicated by TI Errata or other documentation. Modifying the contents of this register could cause this module to operate outside of datasheet specifications. Reset type: XRSn

19.3.2.3 TSNSCTL Register (Offset = 26h) [Reset = 0h]

TSNSCTL is shown in Figure 19-5 and described in Table 19-8.

Return to the Summary Table.

Temperature Sensor Control Register

Figure 19-5. TSNSCTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ENABLE
R-0h							R/W-0h

Table 19-8. TSNSCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	ENABLE	R/W	0h	Temperature Sensor Enable. This bit enables the temperature sensor output to the ADC. 0 Disabled 1 Enabled Reset type: CPU1.SYSRSn

19.3.2.4 LOCK Register (Offset = 2Eh) [Reset = 0h]

LOCK is shown in Figure 19-6 and described in Table 19-9.

Return to the Summary Table.

Lock Register

Figure 19-6. LOCK Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
23	22	21	20	19	18	17	16
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED		
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R-0h		
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	TSNSCTL	RESERVED	RESERVED	RESERVED
R-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 19-9. LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30	RESERVED	R/WSonce	0h	Reserved
29	RESERVED	R/WSonce	0h	Reserved
28	RESERVED	R/WSonce	0h	Reserved
27	RESERVED	R/WSonce	0h	Reserved
26	RESERVED	R/WSonce	0h	Reserved
25	RESERVED	R/WSonce	0h	Reserved
24	RESERVED	R/WSonce	0h	Reserved
23	RESERVED	R/WSonce	0h	Reserved
22	RESERVED	R/WSonce	0h	Reserved
21	RESERVED	R/WSonce	0h	Reserved
20	RESERVED	R/WSonce	0h	Reserved
19	RESERVED	R/WSonce	0h	Reserved
18-7	RESERVED	R	0h	Reserved
6	RESERVED	R/WSonce	0h	Reserved
5	RESERVED	R/WSonce	0h	Reserved
4	RESERVED	R/WSonce	0h	Reserved
3	TSNSCTL	R/WSonce	0h	Temperature Sensor Control Register Lock. Setting this bit will disable any future write to the respective register. This bit can only be cleared by a reset. Reset type: CPU1.SYSRSn
2	RESERVED	R/WSonce	0h	Reserved
1	RESERVED	R/WSonce	0h	Reserved
0	RESERVED	R/WSonce	0h	Reserved

19.3.2.5 ANAREFTRIMA Register (Offset = 36h) [Reset = 0h]

ANAREFTRIMA is shown in Figure 19-7 and described in Table 19-10.

Return to the Summary Table.

Analog Reference Trim A Register

Figure 19-7. ANAREFTRIMA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								RESERVED							
R-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IREFTRIM					BGSLOPETRIM					BGVALTRIM					
R/W-0h					R/W-0h					R/W-0h					

Table 19-10. ANAREFTRIMA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	RESERVED	R/W	0h	Reserved
15-11	IREFTRIM	R/W	0h	Reference Current Trim. This bit field defines the reference current trim value. 0x0 - Untrimmed all other values - Trimmed Reset type: XRSn
10-6	BGSLOPETRIM	R/W	0h	Bandgap Slope Trim. This bit field defines the bandgap slope trim value. 0x0 - Untrimmed all other values - Trimmed Reset type: XRSn
5-0	BGVALTRIM	R/W	0h	Bandgap Value Trim. This bit field defines the bandgap voltage offset trim value. 0x0 - Untrimmed all other values - Trimmed Reset type: XRSn

19.3.2.6 ANAREFTRIMB Register (Offset = 38h) [Reset = 0h]

ANAREFTRIMB is shown in Figure 19-8 and described in Table 19-11.

Return to the Summary Table.

Analog Reference Trim B Register

Figure 19-8. ANAREFTRIMB Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								RESERVED							
R-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IREFTRIM					BGSLOPETRIM					BGVALTRIM					
R/W-0h					R/W-0h					R/W-0h					

Table 19-11. ANAREFTRIMB Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	RESERVED	R/W	0h	Reserved
15-11	IREFTRIM	R/W	0h	Reference Current Trim. This bit field defines the reference current trim value. 0x0 - Untrimmed all other values - Trimmed Reset type: XRSn
10-6	BGSLOPETRIM	R/W	0h	Bandgap Slope Trim. This bit field defines the bandgap slope trim value. 0x0 - Untrimmed all other values - Trimmed Reset type: XRSn
5-0	BGVALTRIM	R/W	0h	Bandgap Value Trim. This bit field defines the bandgap voltage offset trim value. 0x0 - Untrimmed all other values - Trimmed Reset type: XRSn

19.3.2.7 ANAREFTRIMC Register (Offset = 3Ah) [Reset = 0h]

ANAREFTRIMC is shown in Figure 19-9 and described in Table 19-12.

Return to the Summary Table.

Analog Reference Trim C Register

Figure 19-9. ANAREFTRIMC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								RESERVED							
R-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IREFTRIM					BGSLOPETRIM					BGVALTRIM					
R/W-0h					R/W-0h					R/W-0h					

Table 19-12. ANAREFTRIMC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	RESERVED	R/W	0h	Reserved
15-11	IREFTRIM	R/W	0h	Reference Current Trim. This bit field defines the reference current trim value. 0x0 - Untrimmed all other values - Trimmed Reset type: XRSn
10-6	BGSLOPETRIM	R/W	0h	Bandgap Slope Trim. This bit field defines the bandgap slope trim value. 0x0 - Untrimmed all other values - Trimmed Reset type: XRSn
5-0	BGVALTRIM	R/W	0h	Bandgap Value Trim. This bit field defines the bandgap voltage offset trim value. 0x0 - Untrimmed all other values - Trimmed Reset type: XRSn

19.3.2.8 ANAREFTRIMD Register (Offset = 3Ch) [Reset = 0h]

ANAREFTRIMD is shown in Figure 19-10 and described in Table 19-13.

Return to the Summary Table.

Analog Reference Trim D Register

Figure 19-10. ANAREFTRIMD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								RESERVED							
R-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IREFTRIM					BGSLOPETRIM					BGVALTRIM					
R/W-0h					R/W-0h					R/W-0h					

Table 19-13. ANAREFTRIMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	RESERVED	R/W	0h	Reserved
15-11	IREFTRIM	R/W	0h	Reference Current Trim. This bit field defines the reference current trim value. 0x0 - Untrimmed all other values - Trimmed Reset type: XRSn
10-6	BGSLOPETRIM	R/W	0h	Bandgap Slope Trim. This bit field defines the bandgap slope trim value. 0x0 - Untrimmed all other values - Trimmed Reset type: XRSn
5-0	BGVALTRIM	R/W	0h	Bandgap Value Trim. This bit field defines the bandgap voltage offset trim value. 0x0 - Untrimmed all other values - Trimmed Reset type: XRSn

This page intentionally left blank.

Chapter 20

Analog-to-Digital Converter (ADC)

The analog-to-digital converter (ADC) module described in this chapter is a Type 4 ADC. See the *C2000 Real-Time Control Peripheral Reference Guide* for a list of all devices with modules of the same type, to determine the differences between the types, and for a list of device-specific differences within a type.

20.1 Introduction

The ADC module is a successive approximation (SAR) style ADC with selectable resolution of either 16 bits or 12 bits. The ADC is composed of a core and a wrapper. The core is composed of the analog circuits which include the channel select MUX, the sample-and-hold (S/H) circuit, the successive approximation circuits, voltage reference circuits, and other analog support circuits. The wrapper is composed of the digital circuits that configure and control the ADC. These circuits include the logic for programmable conversions, result registers, interfaces to analog circuits, interfaces to the peripheral buses, post-processing circuits, and interfaces to other on-chip modules.

Each ADC module consists of a single sample-and-hold (S/H) circuit. The ADC module is designed to be duplicated multiple times on the same chip, allowing simultaneous sampling or independent operation of multiple ADCs. The ADC wrapper is start-of-conversion (SOC) based (see Section 20.3).

20.1.1 ADC Related Collateral

Foundational Materials

- *ADC Input Circuit Evaluation for C2000 MCUs (TINA-TI) Application Report*
- C2000 Academy - ADC
- PSpice for TI design and simulation tool
- *Real-Time Control Reference Guide*
  - Refer to the ADC section
- TI Precision Labs - ADCs
- TI Precision Labs: Driving the reference input on a SAR ADC (Video)
- TI Precision Labs: Introduction to analog-to-digital converters (ADCs) (Video)
- TI Precision Labs: SAR ADC input driver design (Video)
- TI e2e: Connecting VDDA to VREFHI
- TI e2e: Topologies for ADC Input Protection
- TI e2e: Why does the ADC Input Voltage drop with sampling?
  - Sampling a high impedance voltage divider with ADC
- *Understanding Data Converters Application Report*

Getting Started Materials

- ADC-PWM Synchronization Using ADC Interrupt
  - NOTE: This is a non-TI (third party) site.
- Analog-to-Digital Converter (ADC) Training for C2000 MCUs (Video)
- *Hardware Design Guide for F2800x C2000 Real-Time MCU Series*

Expert Materials

- Analog Engineer's Calculator
- *Analog Engineer's Pocket Reference*
- *Charge-Sharing Driving Circuits for C2000 ADCs (using PSPICE-FOR-TI) Application Report*
- *Charge-Sharing Driving Circuits for C2000 ADCs (using TINA-TI) Application Report*
- Debugging an integrated ADC in a microcontroller using an oscilloscope
- *Methods for Mitigating ADC Memory Cross-Talk Application Report*
- TI Precision Labs: ADC AC specifications (Video)
- TI Precision Labs: ADC Error sources (Video)
- TI Precision Labs: ADC Noise (Video)
- TI Precision Labs: Analog-to-digital converter (ADC) drive topologies (Video)
- TI Precision Labs: Electrical overstress on data converters (Video)
- TI Precision Labs: High-speed ADC fundamentals (Video)
- TI Precision Labs: SAR & Delta-Sigma: Understanding the Difference (Video)
- TI e2e: ADC Bandwidth Clarification
- TI e2e: ADC Calibration and Total Unadjusted Error

- TI e2e: ADC Reference Driver Options
- TI e2e: ADC Resolution with Oversampling
- TI e2e: ADC configuration for interleaved mode
- TI e2e: Simultaneous Sampling with Single ADC

20.1.2 Features

Each ADC has the following features:

- Selectable resolution of 12 bits or 16 bits
- Ratiometric external reference set by VREFHI and VREFLO pins
- Differential signal conversions (16-bit mode only)
- Single-ended signal conversions
- Input multiplexer with up to 16 channels (single-ended) or 8 channels (differential)
- 16 configurable SOCs
- 16 individually addressable result registers
- Multiple trigger sources
  - S/W - software immediate start
  - All ePWMs - ADCSOC A or B
  - GPIO XINT2
  - CPU Timers 0/1/2 (from each C28x core present)
  - ADCINT1/2
- Four flexible PIE interrupts
- Configurable interrupt placement
- Burst mode
- Four post-processing blocks, each with:
  - Saturating offset calibration
  - Error from set-point calculation
  - High, low, and zero-crossing compare, with interrupt and ePWM trip capability
  - Trigger-to-sample delay capture

Note

Not every channel is pinned out from all ADCs. Check the device data manual to determine which channels are available.

20.1.3 Block Diagram

Figure 20-1 shows the block diagram for the ADC core and ADC wrapper.

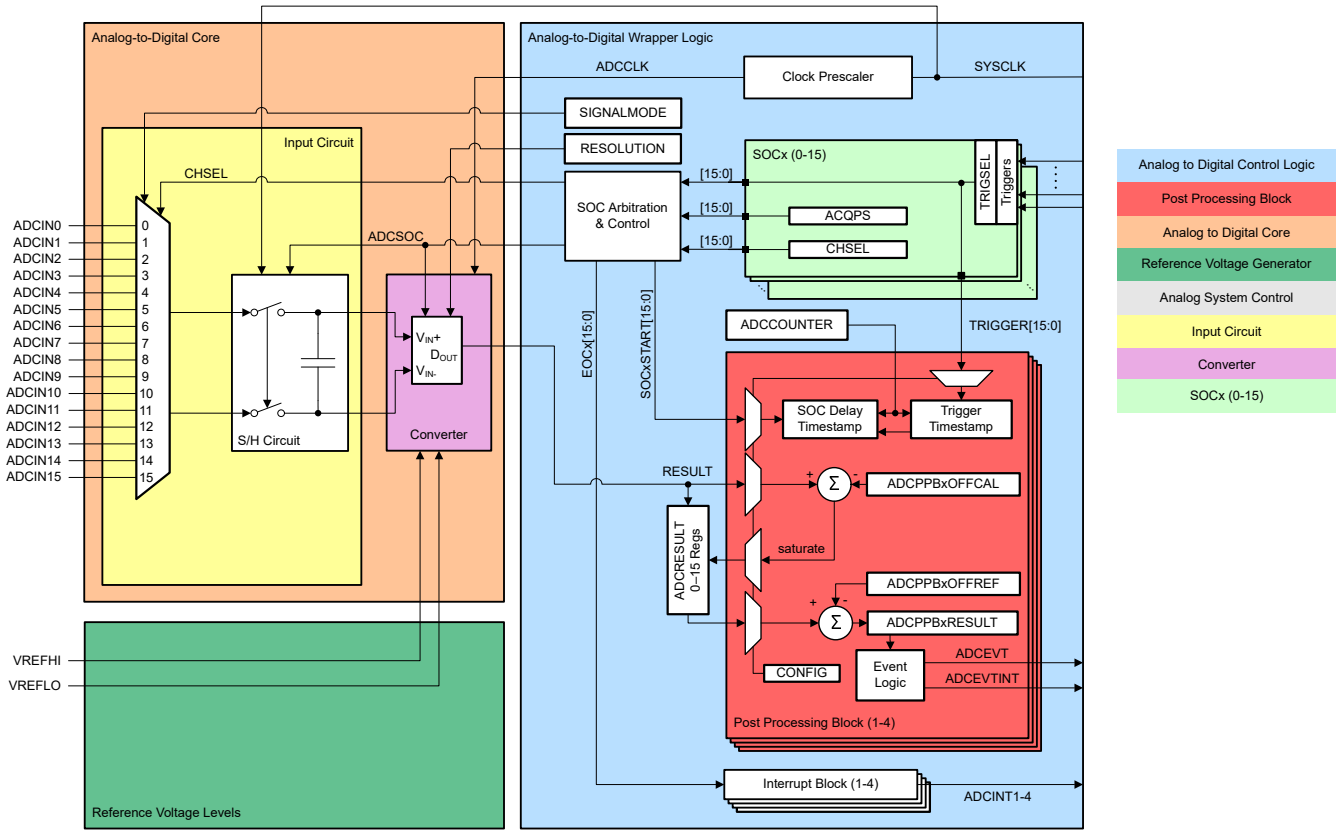


Figure 20-1. ADC Module Block Diagram

Note

The ADC block diagram reflects the number of ADC channels internally configurable on the device. The actual number of available external ADC inputs varies depending on part and package.

20.2 ADC Configurability

Some ADC configurations are individually controlled by the SOCs, while others are globally controlled per ADC module. Table 20-1 summarizes the basic ADC options and their level of configurability. The subsequent sections discuss these configurations.

Table 20-1. ADC Options and Configuration Levels

Options	Configurability
Clock	Per module(1)
Resolution	Per module(1)
Signal mode	Per module
Reference voltage source	Not configurable (external reference only)
Trigger source	Per SOC(1)
Converted channel	Per SOC
Acquisition window duration	Per SOC(1)
EOC location	Per module
Burst Mode	Per module(1)

(1) Writing these values differently to different ADC modules could cause the ADCs to operate asynchronously. See Section 20.13.1 for guidance on when the ADCs are operating synchronously or asynchronously.

20.2.1 Clock Configuration

The base ADC clock is provided directly by the system clock (SYSCLK). SYSCLK is used to generate the ADC acquisition window. The register ADCCTL2 has a PRESCALE field that determines the ADCCLK. ADCCLK is used to clock the converter, and is only active during the conversion phase. At all other times, including during the sample-and-hold window, the ADCCLK signal is gated off.

In 16-bit mode, the core requires approximately 29.5 ADCCLK cycles to process a voltage into a conversion result, while in 12-bit mode, this process requires approximately 10.5 ADCCLK cycles. The choice of resolution will also determine the necessary duration of the acquisition window, see Section 20.13.2.

Note

To determine an appropriate value for ADCCTL2.PRESCALE, see the device data manual to determine the maximum SYSCLK and ADCCLK frequency.

20.2.2 Resolution

The resolution of the ADC determines how finely the analog range is quantized into digital values. This ADC supports a configurable resolution of 16 bits or 12 bits.

The resolution must be configured by using either the AdcSetMode() or ADC_setMode() functions, depending on the header files used, provided in C2000ware in F2838x_Adc.c. These functions make sure that the correct trim is loaded into the ADC trim registers, and must be called at least once after a device reset. Do not configure the resolution by directly writing to the ADCCTL2 register.

The resolution can be changed at any time when the ADC is idle (no active or pending SOCs). No wait time is necessary after changing the resolution before conversions can be initiated. If SOCs are active or pending when the resolution is changed, those SOCs can produce incorrect conversion results.

20.2.3 Voltage Reference

20.2.3.1 External Reference Mode

Each ADC has a VREFHI input and a VREFLO input. In external reference mode, these pins are used as a ratiometric reference to determine the ADC conversion input range.

See Section 20.13.6 for information on how to supply the reference voltage.

Note

- On devices with no external VREFLO pin, VREFLO is internally tied to device analog ground, VSSA.
- See the device data manual to determine the allowable voltage range for VREFHI and VREFLO.
- The external reference mode requires an external capacitor on the VREFHI pin. See the device data manual for the specific value required.

20.2.4 Signal Mode

The ADC supports two signal modes: single-ended and differential.

In single-ended mode, the input voltage to the converter is sampled through a single pin (ADCINx), referenced to VREFLO.

In differential signaling mode, the input voltage to the converter is sampled through a pair of input pins, one of which is the positive input (ADCINxP) and the other is the negative input (ADCINxN). The actual input voltage is the difference between the two (ADCINxP – ADCINxN).

NOTES:

- In differential signal mode, VREFLO must be connected to VSSA.
- In differential signal mode, the common mode voltage is V_{CM} = (ADCINxP + ADCINxN)/2

  The data sheet for a particular device places some requirements on how close this voltage needs to be to: (VREFHI + VREFLO)/2

  Note: The above condition is not met by connecting the negative input to VSSA or VREFLO.
- Differential signaling mode is advantageous because noise encountered on both inputs is largely canceled. The effect can be maximized by routing the positive and negative traces for the same differential input as close together as possible and keeping them symmetrical with respect to the signal reference.

The signal mode must be configured by using either the ADC_setMode() or AdcSetMode() functions, depending on the header files used, provided in C2000warein F2838x_Adc.c. These functions make sure that the correct trim is loaded into the ADC trim registers. These functions must be called at least once after a device reset. The signal mode must not be configured by writing to the ADCCTL2 register directly.

20.2.5 Expected Conversion Results

Based on a given analog input voltage, the ideal expected digital conversion is given in Table 20-2 and Table 20-3. Fractional values are truncated.

Table 20-2. Analog to 12-bit Digital Formulas

	Analog Input	Digital Result
Single-Ended	when ADCINy ≤ VREFLO	ADCRESULTx = 0
	when VREFLO < ADCINy < VREFHI	$ADCRESULTx = 4096 \left(\frac{ADCINy - VREFLO}{VREFHI - VREFLO} \right)$
	when ADCINy ≥ VREFHI	ADCRESULTx = 4095
Differential	when ADCINyP - ADCINyN ≤ -VREFHI	ADCRESULTx = 0
	when -VREFHI < ADCINyP - ADCINyN ≤ VREFHI	$ADCRESULTx = 4096 \left(\frac{ADCINyP - ADCINyN + VREFHI}{2\ VREFHI} \right)$
	when ADCINyP - ADCINyN ≥ VREFHI	ADCRESULTx = 4095

Table 20-3. Analog to 16-bit Digital Formulas

	Analog Input	Digital Result
Single-Ended	when ADCINy ≤ VREFLO	ADCRESULTx=0
	when VREFLO < ADCINy < VREFHI	$ADCRESULTx = 65536 \left(\frac{ADCINy - VREFLO}{VREFHI - VREFLO} \right)$
	when ADCINy ≥VREFHI	ADCRESULTx=65535
Differential	when ADCINyP - ADCINyN ≤ -VREFHI	ADCRESULTx = 0
	when -VREFHI < ADCINyP - ADCINyN ≤ VREFHI	$ADCRESULTx = 65536 \left(\frac{ADCINyP - ADCINyN + VREFHI}{2\ VREFHI} \right)$
	when ADCINyP - ADCINyN ≥ VREFHI	ADCRESULTx = 65535

20.2.6 Interpreting Conversion Results

Based on a given ADC conversion result, the ideal corresponding analog input is given in Table 20-4 and Table 20-5. This corresponds to the center of the possible range of analog voltages that can produce this conversion result.

Table 20-4. 12-Bit Digital-to-Analog Formulas

	Digital Value	Analog Equivalent
Single-Ended	when ADCRESULTy = 0	$\text{ADCINx} \leq \text{VREFLO}$
	when 0 < ADCRESULTy < 4095	$\text{ADCINx} = (\text{VREFHI} - \text{VREFLO})\left(\frac{\text{ADCRESULTy}}{4096}\right) + \text{VREFLO}$
	when ADCRESULTy = 4095	$\text{ADCINx} \geq \text{VREFHI}$
Differential	when ADCRESULTy = 0	$\text{ADCINxP} - \text{ADCINxN} \leq \text{VREFHI}$
	when 0 < ADCRESULTy < 4095	$\text{ADCINxP} - \text{ADCINxN} = \text{VREFHI} \times \left(\frac{2 \times \text{ADCRESULTy}}{4096} - 1\right)$
	when ADCRESULTy = 4095	$\text{ADCINxP} - \text{ADCINxN} \geq \text{VREFHI}$

Table 20-5. 16-Bit Digital-to-Analog Formulas

	Digital Value	Analog Equivalent
Single-Ended	when ADCRESULTy = 0	$\text{ADCINx} \leq \text{VREFLO}$
	when 0 < ADCRESULTy < 65535	$\text{ADCINx} = (\text{VREFHI} - \text{VREFLO})\left(\frac{\text{ADCRESULTy}}{65536}\right) + \text{VREFLO}$
	when ADCRESULTy = 65535	$\text{ADCINx} \geq \text{VREFHI}$
Differential	when ADCRESULTy = 0	$\text{ADCINxP} - \text{ADCINxN} \leq \text{VREFHI}$
	when 0 < ADCRESULTy < 65535	$\text{ADCINxP} - \text{ADCINxN} = \text{VREFHI} \times \left(\frac{2 \times \text{ADCRESULTy}}{65536} - 1\right)$
	when ADCRESULTy = 65535	$\text{ADCINxP} - \text{ADCINxN} \geq \text{VREFHI}$

20.3 SOC Principle of Operation

The ADC triggering and conversion sequencing is accomplished through configurable start-of-conversions (SOCs). Each SOC is a configuration set defining the single conversion of a single channel. In that set, there are three configurations: the trigger source that starts the conversion, the channel to convert, and the acquisition (sample) window duration. Upon receiving the trigger configured for a SOC, the wrapper makes sure that the specified channel is captured using the specified acquisition window duration.

Multiple SOCs can be configured for the same trigger, channel, and acquisition window as desired. Configuring multiple SOCs to use the same trigger allows the trigger to generate a sequence of conversions. Configuring multiple SOCs to use the same trigger and channel allows for oversampling.

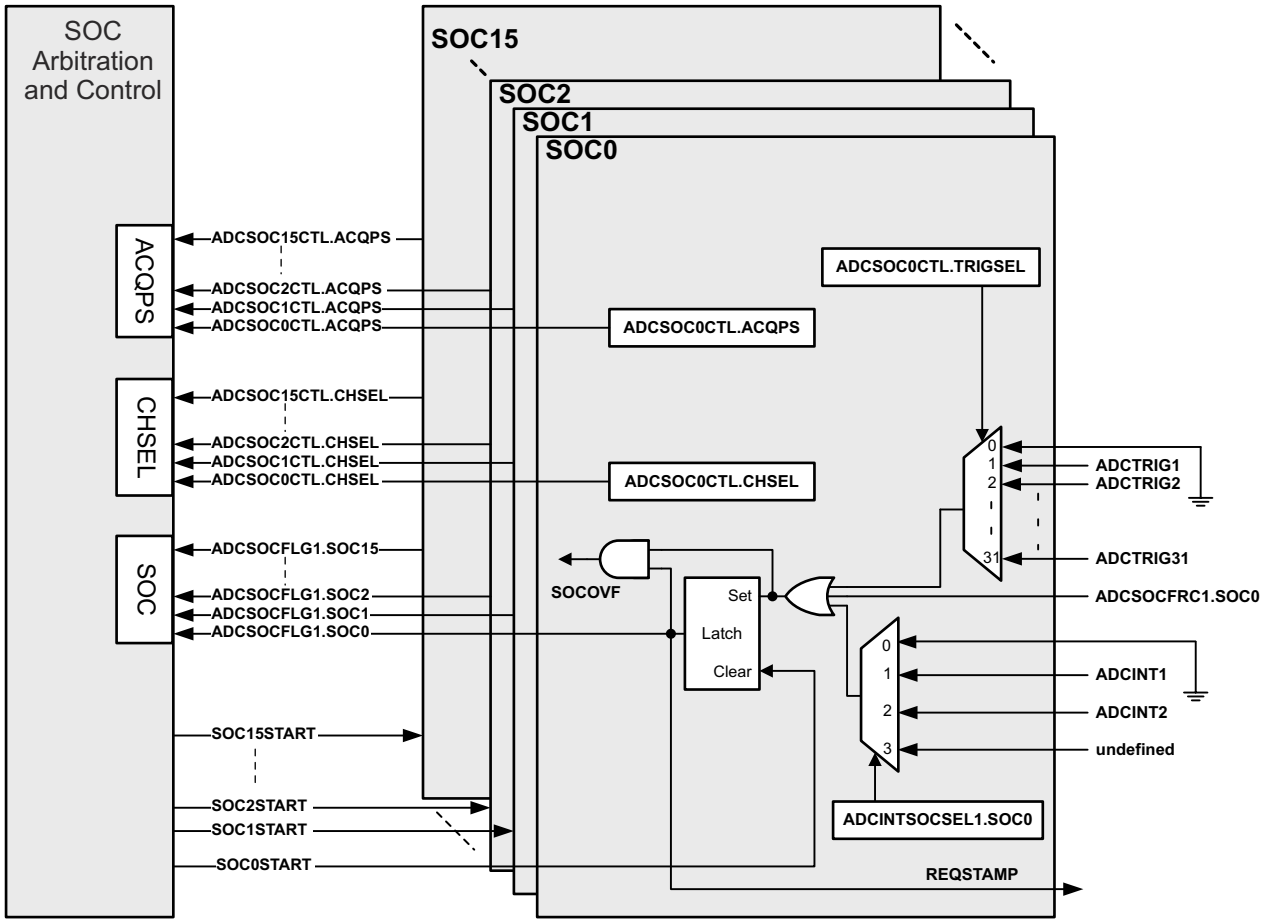


Figure 20-2. SOC Block Diagram

20.3.1 SOC Configuration

Each SOC has its own configuration register, ADCSOCxCTL. Within this register, SOCx can be configured for trigger source, channel to convert, and acquisition (sample) window duration.

20.3.2 Trigger Operation

Each SOC can be configured to start on one of many input triggers. The primary trigger select for SOCx is in the ADCSOCxCTL.TRIGSEL register, which can select between:

- Disabled (software only)
- CPU Timers 0/1/2 (from each C28x core present)
- GPIO: Input X-Bar INPUT5
- ADCSOCA or ADCSOCB from each ePWM module

In addition, each SOC can also be triggered when the ADCINT1 flag or ADCINT2 flag is set. This is achieved by configuring the ADCINTSOCSEL1 register (for SOC0 to SOC7) or the ADCINTSOCSEL2 register (for SOC8 to SOC15). This is useful for creating continuous conversions.

20.3.3 ADC Acquisition (Sample and Hold) Window

External signal sources vary in their ability to drive an analog signal quickly and effectively. To achieve rated resolution, the signal source needs to charge the sampling capacitor in the ADC core to within 0.5 LSBs of the signal voltage. The acquisition window is the amount of time the sampling capacitor is allowed to charge and is configurable for SOCx by the ADCSOCxCTL.ACQPS register.

ACQPS is a 9-bit register that can be set to a value between 0 and 511, resulting in an acquisition window duration of:

Acquisition window = (ACQPS + 1)·(System Clock (SYSCLK) cycle time)

- The acquisition window duration is based on the System Clock (SYSCLK), not the ADC clock (ADCCLK).
- The selected acquisition window duration must be at least as long as one ADCCLK cycle.
- The data manual specifies a minimum acquisition window duration (in nanoseconds). The user is responsible for selecting an acquisition window duration that meets this requirement.

20.3.4 ADC Input Models

For single-ended operation, the ADC input characteristics for values in the single-ended input model (see Figure 20-3) can be found in the device data manual.

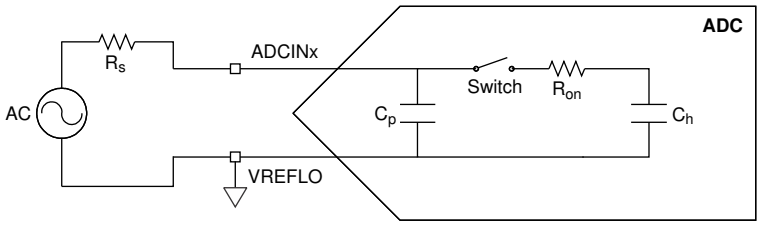


Figure 20-3. Single-Ended Input Model

For differential operation, the ADC input characteristics for values in the differential input model (see Figure 20-4) can be found in the device data manual.

These input models must be used along with actual signal source impedance to determine the acquisition window duration. See Section 20.13.2 for more information.

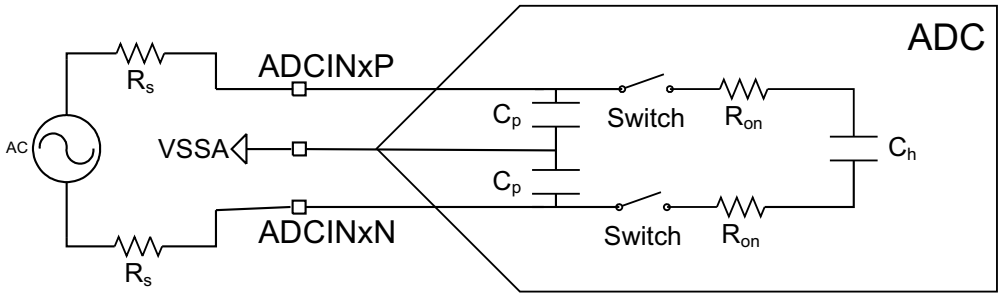


Figure 20-4. Differential Input Model

20.3.5 Channel Selection

Each SOC can be configured to convert any of the ADC channels. This behavior is selected for SOCx by the ADCSOCxCTL.CHSEL register. Depending on the signal mode, the selection is different. For single-ended signal mode, the value in CHSEL selects a single pin as the input. For differential signal mode, the value in CHSEL selects an even-odd pin pair to be the positive and negative inputs. This is summarized in Table 20-6.

Note

Regardless of configured resolution and signal mode, channel 13 on ADC-A (temperature sensor) and channel 12 on all ADCs are sampled in 12-bit single-ended mode.

Table 20-6. Channel Selection of Input Pins

Input Mode	CHSEL	Input	
Single-Ended	0	ADCIN0	
	1	ADCIN1	
	2	ADCIN2	
	3	ADCIN3	
	4	ADCIN4	
	5	ADCIN5	
	6	ADCIN6	
	7	ADCIN7	
	8	ADCIN8	
	9	ADCIN9	
	10	ADCIN10	
	11	ADCIN11	
	12	ADCIN12	
	13	ADCIN13	
	14	ADCIN14	
	15	ADCIN15	
	CHSEL	**Positive Input**	**Negative Input**
Differential	0 or 1	ADCIN0	ADCIN1
	2 or 3	ADCIN2	ADCIN3
	4 or 5	ADCIN4	ADCIN5
	6 or 7	ADCIN6	ADCIN7
	8 or 9	ADCIN8	ADCIN9
	10 or 11	ADCIN10	ADCIN11
	12 or 13	ADCIN12	ADCIN13
	14 or 15	ADCIN14	ADCIN15

20.4 SOC Configuration Examples

The following sections provide some specific examples of how to configure the SOCs to produce some conversions.

20.4.1 Single Conversion from ePWM Trigger

SOC5 is chosen arbitrarily. Any of the 16 SOCs can be used.

Assuming a 100 ns sample window is desired with a SYSCLK frequency of 200 MHz, then the acquisition window duration must be 100 ns/5 ns = 20 SYSCLK cycles. The ACQPS field must therefore be set to 20 - 1 = 19.

```
AdcaRegs.ADCSOC5CTL.bit.CHSEL = 1;      //SOC5 converts ADCINA1
AdcaRegs.ADCSOC5CTL.bit.ACQPS = 19;     //SOC5 uses a sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC5CTL.bit.TRIGSEL = 10;   //SOC5 begins conversion on ePWM3 SOCB
```

As configured, when ePWM3 matches the period and generates the SOCB signal, the ADC begins sampling channel ADCINA1 (SOC5) immediately if the ADC is idle. If the ADC is busy, ADCINA1 begins sampling when SOC5 gains priority (see Section 20.5). The ADC control logic samples ADCINA1 with the specified acquisition window width of 100 ns. Immediately after the acquisition is complete, the ADC begins converting the sampled voltage to a digital value. When the ADC conversion is complete, the results are available in the ADCRESULT5 register (see Section 20.12 for exact sample, conversion, and result latch timings).

20.4.2 Oversampled Conversion from ePWM Trigger

To configure the ADC to oversample ADCINA1 4 times, we use the same configurations as the previous example, but apply them to SOC5, SOC6, SOC7, and SOC8.

```
AdcaRegs.ADCSOC5CTL.bit.CHSEL = 1;        //SOC5 converts ADCINA1
AdcaRegs.ADCSOC5CTL.bit.ACQPS = 19;       //SOC5 uses a sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC5CTL.bit.TRIGSEL = 10;     //SOC5 begins conversion on ePWM3 SOCB
AdcaRegs.ADCSOC6CTL.bit.CHSEL = 1;        //SOC6 converts ADCINA1
AdcaRegs.ADCSOC6CTL.bit.ACQPS = 19;       //SOC6 uses a sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC6CTL.bit.TRIGSEL = 10;     //SOC6 begins conversion on ePWM3 SOCB
AdcaRegs.ADCSOC7CTL.bit.CHSEL = 1;        //SOC7 converts ADCINA1
AdcaRegs.ADCSOC7CTL.bit.ACQPS = 19;       //SOC7 uses a sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC7CTL.bit.TRIGSEL = 10;     //SOC7 begins conversion on ePWM3 SOCB
AdcaRegs.ADCSOC8CTL.bit.CHSEL = 1;        //SOC8 converts ADCINA1
AdcaRegs.ADCSOC8CTL.bit.ACQPS = 19;       //SOC8 uses a sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC8CTL.bit.TRIGSEL = 10;     //SOC8 begins conversion on ePWM3 SOCB
```

As configured, when ePWM3 matches the period and generates the SOCB signal, the ADC begins sampling channel ADCINA1 (SOC5) immediately if the ADC is idle. If the ADC is busy, ADCINA1 begins sampling when SOC5 gains priority (see Section 20.5). Once the conversion is complete for SOC5, SOC6 begins converting ADCINA1 and the results for SOC5 are placed in the ADCRESULT5 register. All four conversions eventually are completed sequentially, with the results in ADCRESULT5, ADCRESULT6, ADCRESULT7, and ADCRESULT8 for SOC5, SOC6, SOC7, and SOC8, respectively.

Note

It is possible, but unlikely, that the ADC can begin converting SOC6, SOC7, or SOC8 before SOC5 depending on the position of the round-robin pointer when the ePWM trigger is received. See Section 20.5 to understand how the next SOC to be converted is chosen.

20.4.3 Multiple Conversions from CPU Timer Trigger

This example shows how to sample multiple signals with different acquisition window requirements. CPU1 Timer 2 is used to generate the trigger. To see how to configure the CPU timer, see the *System Control and Interrupts* chapter.

A good first step when designing a sampling scheme with many signals is to list out the signals and their required acquisition window. From this, calculate the necessary number of SYSCLK cycles for each signal, then the ACQPS register setting. This is shown in Table 20-7, where a SYCLK of 200 MHz is assumed (5 ns cycle time).

Table 20-7. Example Requirements for Multiple Signal Sampling

Signal Name	Acquisition Window Requirement	Acquisition Window (SYSCLK Cycles)	ACQPS Register Value
Signal 1	>120 ns	120 ns/5 ns = 24	24 – 1 = 23
Signal 2	>444 ns	444 ns/5 ns = 89 (round up)	89 – 1 = 88
Signal 3	>110 ns	110 ns/5 ns = 22	22 – 1 = 21
Signal 4	>291 ns	291 ns/5 ns = 59 (round up)	59 – 1 = 58

Next decide which ADC pins to connect to each signal. This is highly dependent on the application board layout. Once the pins are selected, determining the value of CHSEL is straightforward (see Table 20-8).

Table 20-8. Example Connections for Multiple Signal Sampling

Signal Name	ADC Pin	CHSEL Register Value
Signal 1	ADCINA5	5
Signal 2	ADCINA0	0
Signal 3	ADCINA3	3
Signal 4	ADCINA2	2

With the information tabulated, generate the SOC configurations:

```
AdcaRegs.ADCSOC0CTL.bit.CHSEL = 5;        //SOC0 converts ADCINA5
AdcaRegs.ADCSOC0CTL.bit.ACQPS = 23;       //SOC0 uses a sample duration of 24 SYSCLK cycles
AdcaRegs.ADCSOC0CTL.bit.TRIGSEL = 3;      //SOC0 begins conversion on CPU1 Timer 2
AdcaRegs.ADCSOC1CTL.bit.CHSEL = 0;        //SOC1 converts ADCINA0
AdcaRegs.ADCSOC1CTL.bit.ACQPS = 88;       //SOC1 uses a sample duration of 89 SYSCLK cycles
AdcaRegs.ADCSOC1CTL.bit.TRIGSEL = 3;      //SOC1 begins conversion on CPU1 Timer 2
AdcaRegs.ADCSOC2CTL.bit.CHSEL = 3;        //SOC2 converts ADCINA3
AdcaRegs.ADCSOC2CTL.bit.ACQPS = 21;       //SOC2 uses a sample duration of 22 SYSCLK cycles
AdcaRegs.ADCSOC2CTL.bit.TRIGSEL = 3;      //SOC2 begins conversion on CPU1 Timer 2
AdcaRegs.ADCSOC3CTL.bit.CHSEL = 2;        //SOC3 converts ADCINA2
AdcaRegs.ADCSOC3CTL.bit.ACQPS = 58;       //SOC3 uses a sample duration of 59 SYSCLK cycles
AdcaRegs.ADCSOC3CTL.bit.TRIGSEL = 3;      //SOC3 begins conversion on CPU1 Timer 2
```

As configured, when CPU1 Timer 2 generates an event, SOC0, SOC1, SOC2, and SOC3 eventually is sampled and converted, in that order. The conversion results for ACINA5 (Signal 1) are in ADCRESULT0. Similarly, The results for ADCINA0 (Signal 2), ADCINA3 (Signal 3), and ADCINA2 (Signal 4) are in ADCRESULT1, ADCRESULT2, and ADCRESULT3, respectively.

Note

It is possible, but unlikely, that the ADC can begin converting SOC1, SOC2, or SOC3 before SOC0 depending on the position of the round-robin pointer when the CPU Timer trigger is received. See Section 20.5 to understand how the next SOC to be converted is chosen.

20.4.4 Software Triggering of SOCs

At any point, whether or not the SOCs have been configured to accept a specific trigger, a software trigger can set the SOCs to be converted. This is accomplished by writing bits in the ADCSOCFRC1 register.

Software triggering of the previous example without waiting for the CPU1 Timer 2 to generate the trigger could be accomplished by the statement:

```
AdcaRegs.ADCSOCFRC1.all = 0x000F;        //set SOC flags for SOC0 to SOC3
```

20.5 ADC Conversion Priority

When multiple SOC flags are set at the same time, one of two forms of priority determines the converted order. The default priority method is round-robin. In this scheme, no SOC has an inherent higher priority than another. Priority depends on the round-robin pointer (RRPOINTER). The RRPOINTER reflected in the ADCSOCPRIORITYCTL register points to the last SOC converted. The highest priority SOC is given to the next value greater than the RRPOINTER value, wrapping around back to SOC0 after SOC15. At reset the value is 16 since 0 indicates a conversion has already occurred. When RRPOINTER equals 16 the highest priority is given to SOC0. The RRPOINTER is reset when the ADC module is reset or when the reset value is written to the SOCPRICTL register. The ADC module is reset by writing and clearing the SOFTPRES bit corresponding to the ADC instance.

An example of the round-robin priority method is given in Figure 20-5.

The SOCPRIORITY field in the ADCSOCPRIORITYCTL register can be used to assign high priority from a single to all of the SOCs. When configured as high priority, an SOC interrupts the round-robin wheel after any current conversion completes and inserts in as the next conversion. After the conversion completes, the round-robin wheel continues where the conversion was interrupted. If two high priority SOCs are triggered at the same time, the SOC with the lower number takes precedence.

High priority mode is assigned first to SOC0, then in increasing numerical order. The value written in the SOCPRIORITY field defines the first SOC that is not high priority. In other words, if a value of 4 is written into SOCPRIORITY, then SOC0, SOC1, SOC2, and SOC3 are defined as high priority, with SOC0 the highest.

An example using high priority SOC's is given in Figure 20-6.

A After reset, SOC0 is highest priority SOC; SOC7 receives trigger; SOC7 configured channel is converted immediately.

B RRPOINTER changes to point to SOC 7; SOC8 is now highest priority SOC.

C SOC2 & SOC12 triggers rcvd. simultaneously; SOC12 is first on round robin wheel; SOC12 configured channel is converted while SOC2 stays pending.

D RRPOINTER changes to point to SOC 12; SOC2 configured channel is now converted.

E RRPOINTER changes to point to SOC 2; SOC3 is now highest priority SOC.

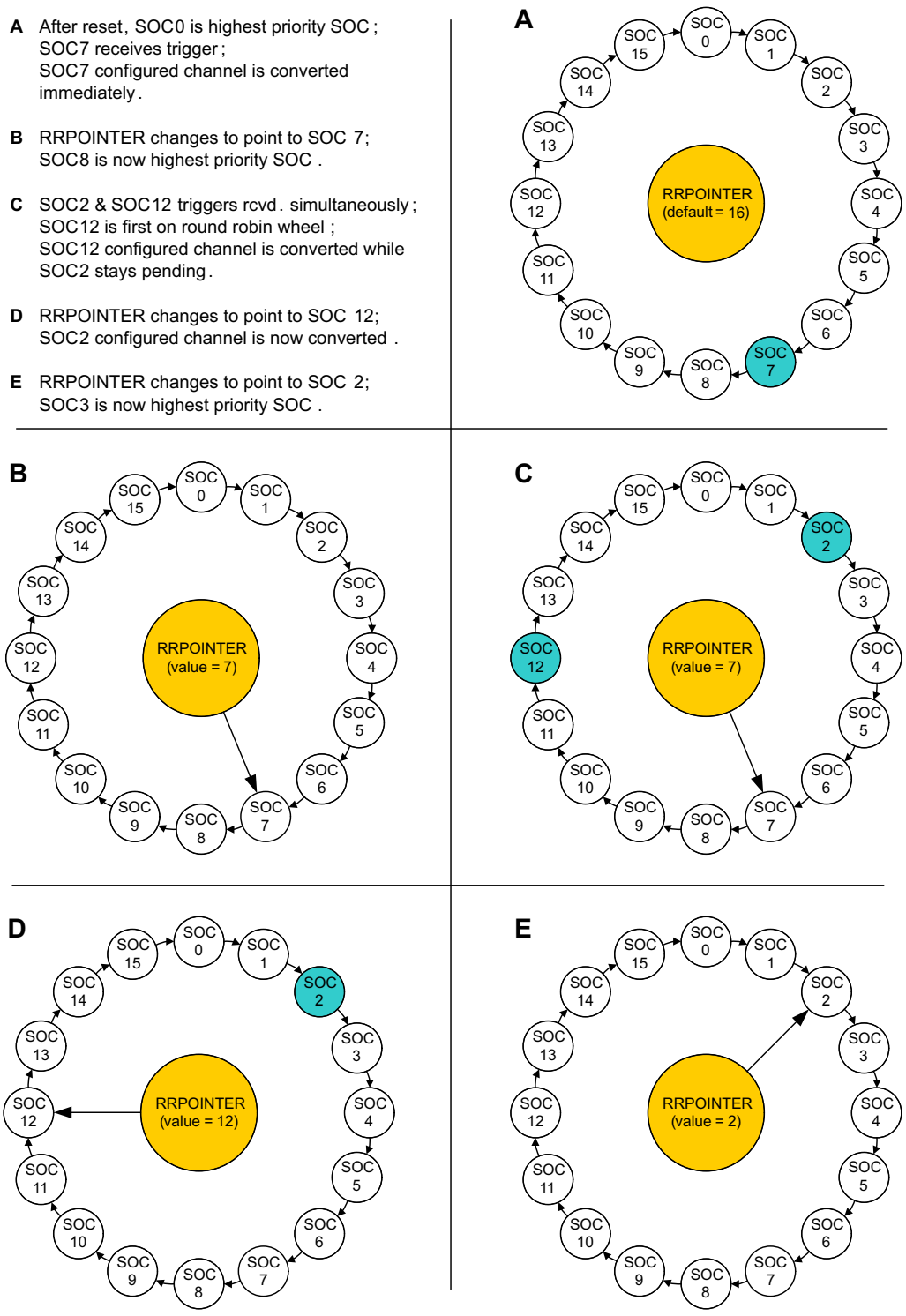


Figure 20-5. Round Robin Priority Example

Example when SOCPRIORITY = 4

A After reset, SOC4 is 1st on round robin wheel ;
SOC7 receives trigger ;
SOC7 configured channel is converted immediately .

B RRPOINTER changes to point to SOC 7;
SOC8 is now 1st on round robin wheel .

C SOC2 & SOC12 triggers rcvd . simultaneously ;
SOC2 interrupts round robin wheel and SOC 2 configured channel is converted while SOC 12 stays pending .

D RRPOINTER stays pointing to 7;
SOC12 configured channel is now converted .

E RRPOINTER changes to point to SOC 12;
SOC13 is now 1st on round robin wheel .

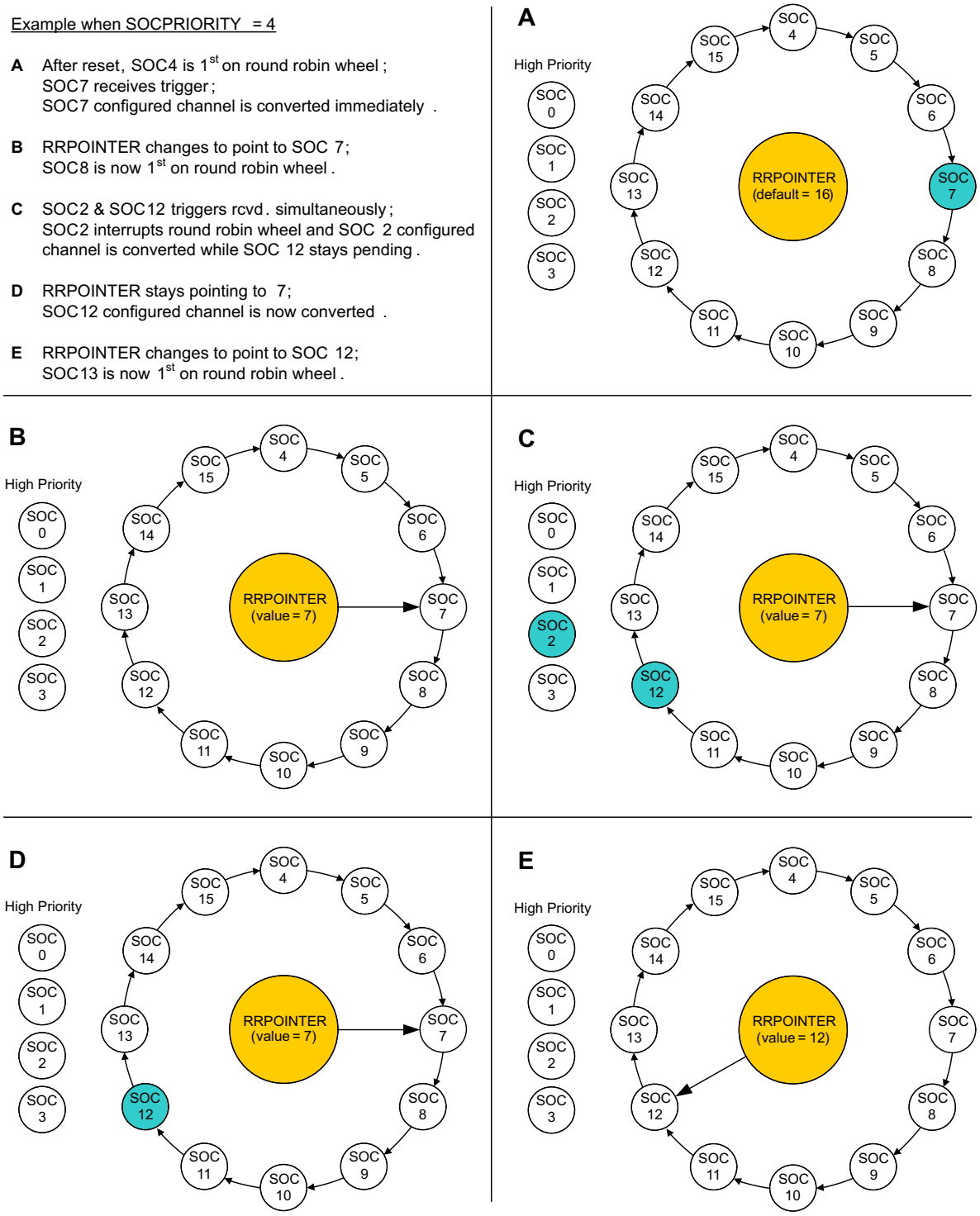


Figure 20-6. High Priority Example

20.6 Burst Mode

Burst mode allows a single trigger to walk through the round-robin SOCs one or more at a time. Setting the bit BURSTEN in the ADCBURSTCTL register configures the ADC wrapper for burst mode. This causes the TRIGSEL field to be ignored, but only for SOCs that are configured for round-robin operation (not high priority). Instead of the TRIGSEL field, all round-robin SOCs are triggered based on the BURSTTRIG field in the ADCBURSTCTL register. Upon reception of the burst trigger, the ADC wrapper does not set all round-robin SOCs to be converted, but only (ADCBURSTCTL.BURSTSIZE + 1) SOCs. The first SOC to be set is the SOC with the highest priority based on the round-robin pointer, and subsequent SOCs are set until BURSTSIZE SOCs have been set.

Note

When configuring the ADC for burst mode, the user is responsible for ensuring that each burst of conversions is allowed to complete before the next burst trigger is received. The value of (ADCBURSTCTL.BURSTSIZE + 1) must be less than or equal to the number of SOCs configured for round-robin priority.

For example, if SOCPRIORITY = 12, that is, SOC12, SOC13, SOC14, and SOC15 are in round-robin, ADCBURSTCTL.BURSTSIZE setting must be ≤3 for burst mode to operate correctly.

20.6.1 Burst Mode Example

Burst mode can be used to sample a different set of signals on every other trigger. In the following example, ADCIN7 and ADCIN5 are converted on the first trigger from CPU1 Timer 2 and every other trigger thereafter. ADCIN2 and ACIN3 are converted on the second trigger from CPU1 Timer 2 and every other trigger thereafter. All signals are converted with 20 SYSCLK cycle wide acquisition windows, but different durations can be configured for each SOC as desired.

```
AdcaRegs.BURSTCTL.BURSTEN = 1;              //Enable ADC burst mode
AdcaRegs.BURSTCTL.BURSTTRIG = 3;            //CPU1 Timer 2 will trigger burst of conversions
AdcaRegs.BURSTCTL.BURSTSIZE = 1;            //conversion bursts are 1 + 1 = 2 conversions long
AdcaRegs.SOCPRICTL.bit.SOCPRIORITY = 12;    //SOC0 to SOC11 are high priority
AdcaRegs.ADCSOC12CTL.bit.CHSEL = 7;         //SOC12 will convert ADCINA7
AdcaRegs.ADCSOC12CTL.bit.ACQPS = 19;        //SOC12 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC13CTL.bit.CHSEL = 5;         //SOC13 will convert ADCINA5
AdcaRegs.ADCSOC13CTL.bit.ACQPS = 19;        //SOC13 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC14CTL.bit.CHSEL = 2;         //SOC14 will convert ADCINA2
AdcaRegs.ADCSOC14CTL.bit.ACQPS = 19;        //SOC14 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC15CTL.bit.CHSEL = 3;         //SOC15 will convert ADCINA3
AdcaRegs.ADCSOC15CTL.bit.ACQPS = 19;        //SOC15 will use sample duration of 20 SYSCLK cycles
```

When the first CPU1 Timer 2 trigger is received, SOC12 and SOC13 are converted immediately if the ADC is idle. If the ADC is busy, SOC12 and SOC13 are converted once their SOCs gain priority. The results for SOC12 and SOC13 are in ADCRESULT12 and ADCRESULT13, respectively. After SOC13 completes, the round-robin pointer gives the highest priority to SOC14. Because of this, when the next CPU1 Timer 2 trigger is received, SOC14 and SOC15 is set as pending and eventually converted. The results for SOC14 and SOC15 are in ADCRESULT14 and ADCRESULT15, respectively. Subsequent triggers continue to toggle between converting SOC12 and SOC13, and converting SOC14 and SOC15.

While the above example toggles between two sets of conversions, three or more different sets of conversions can be achieved using a similar approach.

20.6.2 Burst Mode Priority Example

An example of priority resolution using burst mode and high-priority SOCs is presented in Figure 20-7.

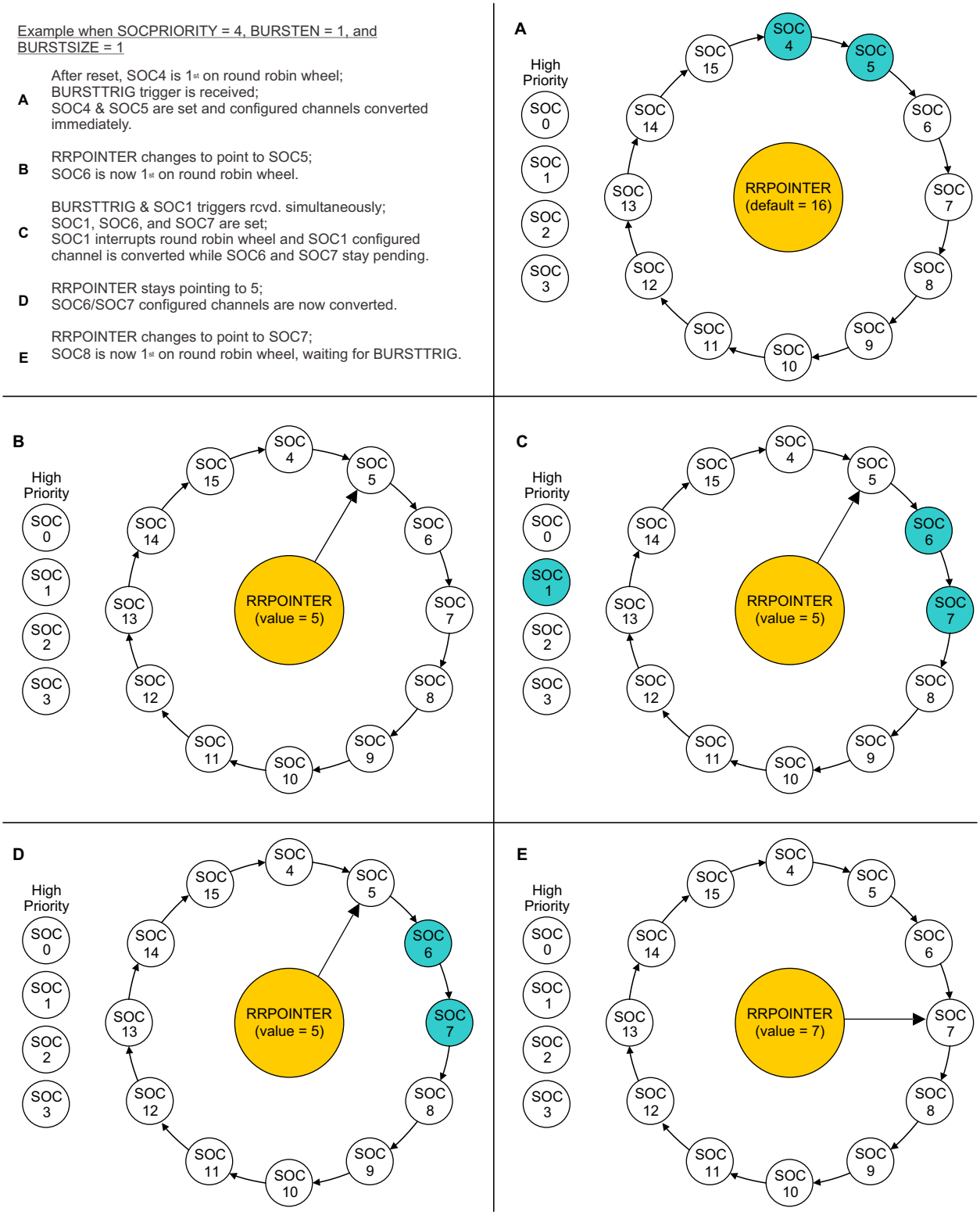


Figure 20-7. Burst Priority Example

20.7 EOC and Interrupt Operation

Each SOC has a corresponding end-of-conversion (EOC) signal. This EOC signal can be used to trigger an ADC interrupt. The ADC can be configured to generate the EOC pulse at either the end of the acquisition window or at the end of the voltage conversion. This is configured using the bit INTPULSEPOS in the ADCCTL1 register. See Section 20.12 for exact EOC pulse location.

The flag bit for each ADCINT can be read directly to determine if the associated SOC is complete or the interrupt can be passed on to the PIE.

Note

The ADCCTL1.ADCBSY bit being clear does not indicate that all conversions in a set of SOCs have completed, only that the ADC is ready to process the next conversion. To determine if a sequence of SOCs is complete, link an ADCINT flag to the last SOC in the sequence and monitor that ADCINT flag.

Figure 20-8 shows a block diagram of the ADC interrupt structure.

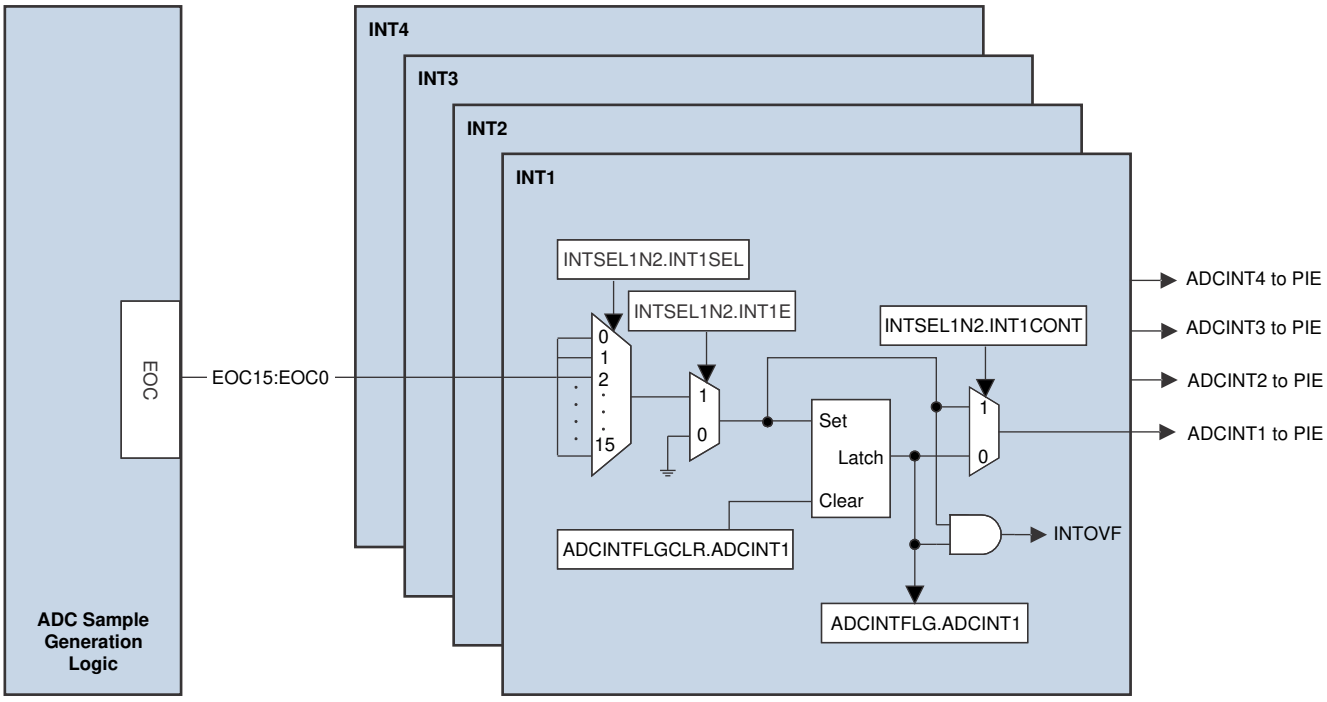


Figure 20-8. ADC EOC Interrupts

20.7.1 Interrupt Overflow

If the EOC signal sets a flag in the ADCINTFLG register, but that flag is already set, an interrupt overflow occurs. By default, overflow interrupts are not passed on to the PIE module. When an overflow occurs on a given flag in the ADCINTFLG register, the corresponding flag in the ADCINOVF register is set. This overflow flag is only used to detect that an overflow has occurred; the flag does not block further interrupts from propagating to the PIE module.

When an ADC interrupt overflow occurs, the application must check the appropriate ADCINTOVF flag inside the ISR or in the background loop and take appropriate action when an overflow is detected. The following code snippets demonstrate how to check the ADCINOVF flag inside the ISR after attempting to clear the ADCINT flag.

```
// Clear the interrupt flag
AdcaRegs.ADCINTFLGCLR.bit.ADCINT1 = 1;      //clear INT1 flag for ADC-A

// Check if an overflow has occurred
if(1 == AdcaRegs.ADCINTOVF.bit.ADCINT1)     //ADCINT overflow occurred
{
    AdcaRegs.ADCINTOVFCLR.bit.ADCINT1 = 1   //Clear overflow flag
    AdcaRegs.ADCINTFLGCLR.bit.ADCINT1 = 1   //Re-clear ADCINT flag
}
```

```
//
// Clear the interrupt flag
//
ADC_clearInterruptStatus(ADCA_BASE, ADC_INT_NUMBER1);

//
// Check if an overflow has occurred
//
if(true == ADC_getInterruptOverflowStatus(ADCA_BASE, ADC_INT_NUMBER1))
{
    ADC_clearInterruptOverflowStatus(ADCA_BASE, ADC_INT_NUMBER1);
    ADC_clearInterruptStatus(ADCA_BASE, ADC_INT_NUMBER1);
}
```

20.7.2 Continue to Interrupt Mode

The INTxCONT bits in the ADCINTSEL1N2 and ADCINTSEL3N4 registers configure how interrupts are handled when an ADCINTFLG has not yet been cleared from a prior interrupt. This mode is disabled by default and additional overlapping interrupts are not issued to the PIE. By activating this mode, ADC interrupts always reach the PIE. If interrupts occur while ADCINTFLG is set, the ADCINTOVF register remains set regardless of the configuration of the INTxCONT bits.

20.7.3 Early Interrupt Configuration Mode

Enabling early interrupt mode can allow the application to enter the ADC interrupt service routine before the ADC results are ready. This allows the application to do any necessary pre-work so that the application can act on the ADC results immediately when the ADC results become available. If the timing of the early interrupt is too early, then the application needs to waste time until the updated ADC results become available. To prevent this situation, the time the ADC interrupt is entered in early interrupt mode is configurable by way of the DELAY field in the ADCINTCYCLE register.

- To use the configurable interrupt time, the ADC must be in early interrupt mode. To achieve this, clear the bit INTPULSEPOS to 0 in ADCCTL1.
- The DELAY value in the ADCINTCYCLE register sets the number of additional SYSCLK cycles after the falling edge of the SOC pulse before the ADCINT flag is set.
- If the value of DELAY goes beyond EOC, the ADC interrupt is generated along with EOC.
- Writing values to DELAY when INTPULSEPOS is set to 1 does not have any effect on the interrupt generation.

20.8 Post-Processing Blocks

Each ADC module contains four post-processing blocks (PPB). These blocks can be associated with any of the RESULT registers using the ADCPPBxCONFIG.CONFIG bit field. The post-processing blocks have the ability to:

- Remove an offset associated with the ADCIN channel
- Subtract out a reference value
- Flag a zero-crossing point, with the option to trip a PWM and generate an interrupt
- Flag a high or low compare limit, with the option to trip a PWM and generate an interrupt
- Record the delay between the associated SOC trigger and when sampling actually begins

Figure 20-9 presents the structure of each PPB. Subsequent sections explain the use of each submodule.

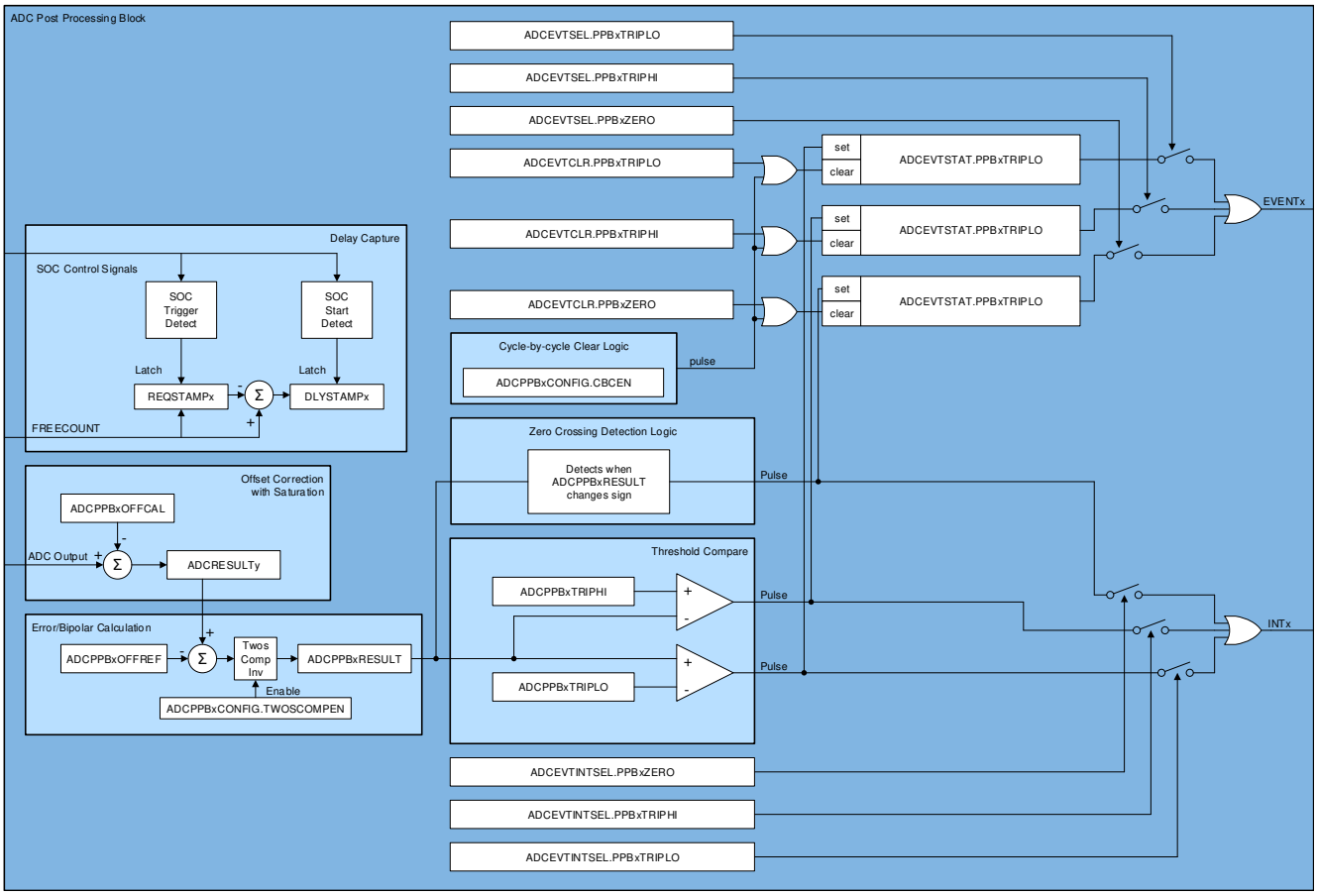


Figure 20-9. ADC PPB Block Diagram

20.8.1 PPB Offset Correction

In many applications, external sensors and signal sources produce an offset. A global trimming of the ADC offset is not enough to compensate for these offsets, which vary from channel to channel. The post-processing block can remove these offsets with zero overhead, saving numerous cycles in tight control loops.

Offset correction is accomplished by first pointing the ADCPPBxCONFIG.CONFIG to the desired SOC, then writing an offset correction value to the ADCPPBxOFFCAL.OFFCAL register. The post-processing block automatically adds or subtracts the value in the OFFCAL register from the raw conversion result and stores the value in the ADCRESULT register. This addition/subtraction saturates at 0 on the low end and either 4095 or 65535 on the high end for 12-bit or 16-bit mode, respectively.

Note

- Writing a 0 to the OFFCAL register effectively disables the offset correction feature, passing the raw result unchanged to the ADCRESULT register.
- It is possible to point multiple PPBs to the same SOC. In this case, the OFFCAL value that is actually applied comes from the PPB with the highest number.
- In particular, care needs to be taken when using the PPB on SOC0, as all PPBs point to this SOC by default. This can cause unintentional overwriting of offset correction of a lower numbered PPB by a higher numbered PPB.

20.8.2 PPB Error Calculation

In many applications, an error from a set point or expected value must be computed from the digital output of an ADC conversion. In other cases, a bipolar signal is necessary or convenient for control calculations. The PPB can perform these functions automatically, reducing the sample to output latency and reducing software overhead.

Error calculation is accomplished by first pointing the ADCPPBxCONFIG.CONFIG to the desired SOC, then writing a value to the ADCPPBxOFFCAL.OFFREF register. The post-processing block automatically subtracts the value in the OFFREF register from the ADCRESULT value and stores the value in the ADCPPBxRESULT register. This subtraction produces a sign-extended 32-bit result. It is also possible to selectively invert the calculated value before storing in the ADCPPBxRESULT register by setting the TWOSCOMPEN bit in the ADCPPBxCONFIG register.

Note

- In 12-bit mode, do not write a value larger than 12 bits to the ADCPPBxOFFREF register.
- Since the ADCPPBxRESULT register is unique for each PPB, it is possible to point multiple PPBs to the same SOC and get different results for each PPB.
- Writing a 0 to the ADCPPBxOFFREF register effectively disables the error calculation feature, passing the ADCRESULT value unchanged to the ADCPPBxRESULT register.
- Writing a new value to ADCPPBxOFFREF causes an immediate update to the ADCPPBxRESULT register. However, the flags coming out of the PPB do not change until the next end-of-conversion (EOC). For instance, if changing the ADCPPBxOFFREF register causes ADCPPBxRESULT to change signs, but the next conversion brings the result back to the same sign as before the OFFREF change, no ADCPPBxZERO flag is set.

20.8.3 PPB Limit Detection and Zero-Crossing Detection

Many applications perform a limit check against the ADC conversion results. The PPB can automatically perform a check against high and low limits, or whenever ADCPPBxRESULT changes sign. Based on these comparisons, the PPB can generate a trip to the PWM and an interrupt automatically, lowering the sample to

ePWM latency and reducing software overhead. This functionality also enables safety-conscious applications to trip the ePWM based on an out-of-range ADC conversion without any CPU intervention.

To enable this functionality, first point the ADCPPBxCONFIG.CONFIG to the desired SOC, then write a value to one or both of the registers ADCPPBxTRIPHI.LIMITHI and ADCPPBxTRIPLO.LIMITLO (zero-crossing detection does not require further configuration). Whenever these limits are exceeded, the PPBxTRIPHI bit or PPBxTRIPLO bit is set in the ADCEVTSTAT register. Note that the PPBxZERO bit in the ADCEVTSTAT register is gated by end-of-conversion (EOC), not by the sign change in the ADCPPBxRESULT register. The ADCEVTCLR register has corresponding bits to clear these event flags. The ADCEVTSEL register has corresponding bits which allow the events to propagate through to the PWM. The ADCEVTINTSEL register has corresponding bits that allow the events to propagate through to the PIE.

One PIE interrupt is shared between all the PPBs for a given ADC module as shown in Figure 20-10.

Note

- If different actions need to be taken for different PPB events from the same ADC module, then the ADCEVTINT ISR has to read the PPB event flags in the ADCEVTSTAT register to determine which event caused the interrupt.
- If different ePWM trips need to be generated separately for high compare, low compare, and zero-crossing, this can be achieved by pointing multiple PPBs to the same SOC.
- The zero-crossing detect circuit considers a result of zero to be positive.

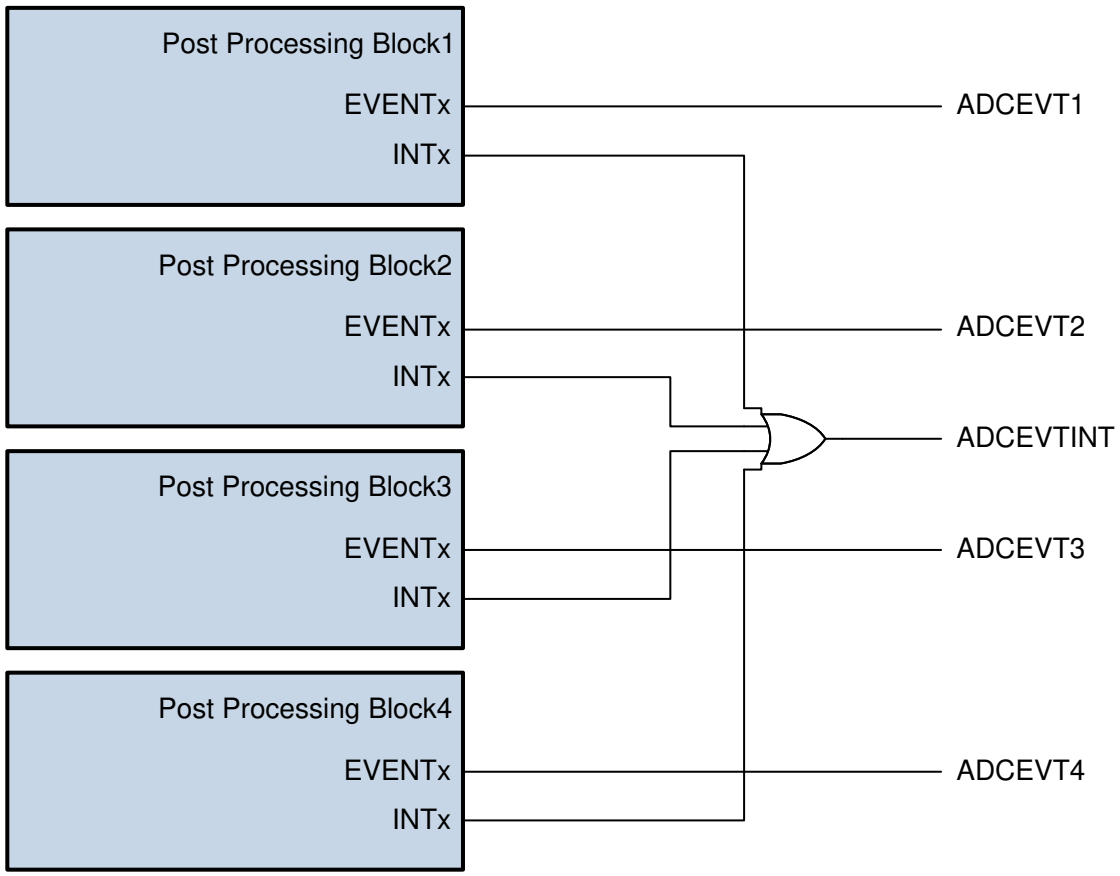


Figure 20-10. ADC PPB Interrupt Event

20.8.4 PPB Sample Delay Capture

When multiple control loops are running asynchronously on the same ADC, there is a chance that an ADC request from two or more loops collide, causing one of the samples to be delayed. This shows up as a measurement error in the system. By knowing when this delay occurs and the amount of delay that has occurred, software can employ extrapolation techniques to reduce the error.

To this effect, each PPB has the field DLYSTAMP in the ADCPPBxSTAMP register. This field contains the number of SYSCLK cycles between when the associate SOC was triggered and when the SOC began converting.

This is achieved by having a global 12-bit free running counter based off of SYSCLK, which is in the field FREECOUNT in the ADCCOUNTER register. When the trigger for the associated SOC arrives, the value of this counter is loaded into the bit field ADCPPBxTRIPLO.REQSTAMP. When the actual sample window for that SOC begins, the value in REQSTAMP is subtracted from the current FREECOUNT value and stored in DLYSTAMP.

Note

If more than 4096 SYSCLK cycles elapse between the SOC trigger and the actual start of the SOC acquisition, the FREECOUNT register can overflow more than once, leading to incorrect DLYSTAMP value. Be cautious when using very slow conversions to prevent this from happening.

The sample delay capture does not function, if the associated SOC is triggered using software. The sample delay capture, however, correctly records the delay, if the software triggering of a different SOC causes the SOC associated with the PPB to be delayed

20.9 Opens/Shorts Detection Circuit (OSDETECT)

The opens/shorts detection circuit (OSDETECT) can be used to detect pin faults in the system. The circuit connects to the ADC input after the channel select multiplexer but before the S+H circuit as shown in Figure 20-11.

Note

- The divider resistance tolerances can vary widely; hence, this feature must not be used to check for conversion accuracy.
- See the data sheet for implementation and availability of analog input channels.
- Due to high drive impedance, a S+H duration much longer than the ADC minimum is needed.

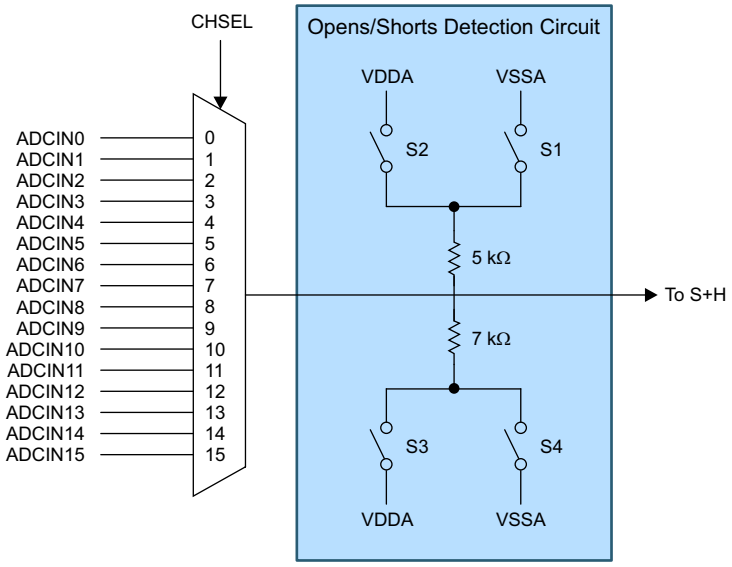


Figure 20-11. Opens/Shorts Detection Circuit

The circuit can be operated by writing a value to the DETECTCFG field in the ADCOSDETECT register. This causes the circuit to source a voltage onto the input during the S+H phase of any conversion. The voltage and drive strength of the OSDETECT circuit for different DETECTCFG settings is given in Table 20-9.

Table 20-9. DETECTCFG Settings

ADCOSDETECT. DETECTCFG	Source Voltage	S4	S3	S2	S1	Drive Impedance
0	Off	Open	Open	Open	Open	Open
1	Zero Scale	Closed	Open	Open	Closed	5K || 7K
2	Full Scale	Open	Closed	Closed	Open	5K || 7K
3	5/12 VDDA	Open	Closed	Open	Closed	5K || 7K
4	7/12 VDDA	Closed	Open	Closed	Open	5K || 7K
5	Zero Scale	Open	Open	Open	Closed	5K
6	Full Scale	Open	Open	Closed	Open	5K
7	Zero Scale	Closed	Open	Open	Open	7K

20.9.1 Implementation

A representative circuit with the OSDETECT implementation consists of the signal source with series resistance R_S, shunt capacitor C_P, the equivalent OSDETECT resistance $R_{OSDETECT}$ and voltage $V_{OSDETECT}$ is shown in Figure 20-12 and can be used as a basis to calculate the signal level going in to the sampling capacitor. $R_{OSDETECT}$ and $V_{OSDETECT}$ are the equivalent input resistance and voltage source contributed by the OSDETECT circuit with values shown in Table 20-9 for the different configuration settings. Refer to Figure 20-12 when deriving the input signal to S/H if signal source V_S is driving while the OSDETECT feature is enabled.

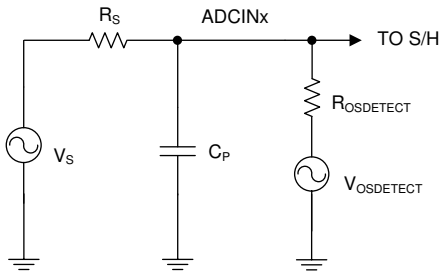


Figure 20-12. Input Circuit Equivalent with OSDETECT Enabled

The input impedance R_S and C_P are integral parts of the signal source or can have been implemented in the design to precondition the signal or to control signal settling time to meet S/H requirements. The input path has to be considered when using the OSDETECT feature, as this affects the conversion results. For instance, driving an input signal when this feature is enabled connects signal V_S to the OSDETECT circuit through R_S and affects the ADC results. Larger C_P values (in the order greater than hundreds of pF) require using higher ACQPS to make sure the signal at the input has settled prior to conversion.

To enable the circuit:

1. Configure the ADC for conversion (for example, channel, SOC, ACQPS, prescaler, trigger, and so on).
2. Set up the ADCOSDETECT register for the desired voltage divider connection as shown in Table 20-9.
3. Initiate a conversion and inspect the conversion result.

Interpret the results based on what is driving on the input side and what are the values of R_S and C_P. If the V_S signal can be disconnected from the input pin, the circuit can be used to detect open and shorted input pins as described in the following sections.

20.9.2 Detecting an Open Input Pin

By cycling through the various OSDETECT settings, the input signal is pulled towards the sourced voltages. An input with good drive strength (pin not open) is minimally affected. However, if the pin is open, the sampled voltages is close to the source voltages specified in Table 20-9.

20.9.3 Detecting a Shorted Input Pin

By cycling through the various OSDETECT settings, the input signal is pulled towards the sourced voltages. An input with finite drive strength (pin not shorted) is pulled toward each sourced voltage. However, if the pin is shorted, the signal remains at the same voltage.

20.10 Power-Up Sequence

Upon device power-up or system level reset, the ADC is powered down and disabled. When powering up the ADC, use the following sequence:

1. Set the bit to enable the desired ADC clock in the PCLKCR13 register.
2. Set the desired ADC clock divider in the PRESCALE field of ADCCTL2.
3. Power up the ADC by setting the ADCPWDNZ bit in ADCCTL1.
4. Allow a delay before sampling. See the data manual for the necessary time.

If multiple ADCs are powered up simultaneously, steps 1 and step 3 can each be done for all ADCs in one write instruction. Also, only one delay is necessary as long as the delay occurs after all the ADCs have begun powering up.

20.11 ADC Calibration

During the fabrication and test process, Texas Instruments calibrates the gain, offset, and linearity of the ADCs and the offset of the buffered DACs. These trim settings are embedded into TI reserved OTP memory and can be loaded using C-callable functions.

- The Device_cal() function copies the trim values for ADC and DAC offset from OTP memory to their respective trim registers.
- The CalAdcINL() function copies the trim values for linearity from OTP memory to the respective trim registers for the specified ADC.
- A different offset trim is required for each possible combination of resolution and signal mode. The GetAdcOffsetTrimOTP(Uint16) function takes an input value corresponding to the ADC, resolution, and signal mode and returns the corresponding offset trim value from OTP memory, which the user then moves into the ADC offset trim register.
- The trim functions in Device_cal() are callable in C2000ware as ADC_setOFFSETTRIM(), ADC_setINLTRIM(), and DAC_setDACTRIM(). These functions fetch trim values from the TI reserved OTP memory source locations where the values are stored during test process as well as the analog module register destinations where the trim values are copied to.

Until the appropriate factory trim is loaded, the ADC and other analog modules are not specified to operate within the data manual specifications. Similarly, if trim values other than the factory settings are placed into the trim registers, the ADC (and other modules) is not specified to operate within the data manual specifications.

The boot ROM calls the calibration functions so trim values must be initially populated without user intervention. However, if the trims are cleared due to a module reset or modified for some other reason, then the user can call the calibration functions (defined in the header files).

Refer to Section 20.11.2 for ADC registers being populated during ADC calibration.

20.11.1 ADC Zero Offset Calibration

Zero offset error is defined as the difference from 0 that occurs when converting a voltage at VREFLO (single-ended operation) or the difference from (maximum code/2) when converting ADCINxP = ADCINxN (differential mode). The zero offset error can be positive or negative. To correct this error, an adjustment of equal magnitude and opposite polarity is written into the ADCOFFTRIM register. The value contained in this register is applied before the results are available in the ADC result registers. This operation is fully contained within the ADC core, so the timing of the results is not affected and the full dynamic range of the ADC is maintained for any trim value.

Using the GetAdcOffsetTrimOTP(Uint16) function, the ADCOFFTRIM register can be loaded with the factory calibrated offset error correction. The user can modify the ADCOFFTRIM register to compensate for additional offset error induced by the application environment if desired, but this is not typically necessary to achieve data manual specified performance.

Note

Regardless of the converter resolution, the size of each ADCOFFTRIM step is (VREFHI-VREFLO)/65536.

Use the following procedure to re-calibrate the ADC offset in 12-bit, single-ended mode:

1. Set ADCOFFTRIM to +112 steps (0x70). This adds an artificial offset to account for negative offset that can reside in the ADC core.
2. Perform some multiple of 16 conversions on VREFLO (internal connection), accumulating the results (for example, 32*16 conversions = 512 conversions).
3. Divide the accumulated result by the multiple of 16 (for example, for 512 conversions, divide by 32).
4. Set ADCOFFTRIM to 112 – result from step 3.

Use the following procedure to recalibrate the ADC offset in 16-bit, differential mode:

1. Set ADCOFFTRIM to no adjustment (0x00).
2. Short ADCINxP and ADCINyN together (external connection) to a voltage near Vrefcm and accumulate some multiple of 16 conversions (for example, 32*16 conversions = 512 conversions).
3. Divide the accumulated result by the number of conversions (for example, for 512 conversions, divide by 512).
4. Set ADCOFFTRIM to 0 – result from step 3).

20.11.2 ADC Calibration Routines in OTP Memory

During factory testing, the calibration values for ADC offset (ADCOFFTRIM register) and ADC linearity correction (ADCINLTRIM1-6 registers) are measured per device. The resulting correction values for ADC offset and linearity and the associated functions to populate these into the corresponding ADC registers are stored in the OTP memory.

20.12 ADC Timings

The process of converting an analog voltage to a digital value is broken down into an S+H phase and a conversion phase. The ADC sample and hold circuits (S+H) are clocked by SYSCLK while the ADC conversion process is clocked by ADCCLK. ADCCLK is generated by dividing down SYSCLK based on the PRESCALE field in the ADCCTL2 register.

The S+H duration is the value of the ACQPS field of the SOC being converted, plus one, times the SYSCLK period. The user must make sure that this duration exceeds both 1 ADCCLK period and the minimum S+H duration specified in the data manual. The conversion time is approximately 10.5 ADCCLK cycles in 12-bit mode and 29.5 ADCCLK cycles in 16-bit mode. The exact conversion time is always a whole number of SYSCLK cycles. See the timing diagrams and tables in Section 20.12.1 for exact timings.

20.12.1 ADC Timing Diagrams

The following diagrams show the ADC conversion timings for two SOCs given the following assumptions:

- SOC0 and SOC1 are configured to use the same trigger.
- No other SOCs are converting or pending when the trigger occurs.
- The round robin pointer is in a state that causes SOC0 to convert first.
- ADCINTSEL is configured to set an ADCINT flag upon end of conversion for SOC0 (whether this flag propagates through to the CPU to cause an interrupt is determined by the configurations in the PIE module).

Table 20-10 describes the parameters in the following timing diagrams. Table 20-11 and Table 20-12 list the ADC timings.

Table 20-10. ADC Timing Parameter Descriptions

Parameter	Description
t_{SH}	The duration of the S+H window. At the end of this window, the value on the S+H capacitor becomes the voltage to be converted into a digital value. The duration is given by (ACQPS + 1) SYSCLK cycles. ACQPS can be configured individually for each SOC, so t_{SH} is not necessarily the same for different SOCs. **Note:** The value on the S+H capacitor is captured approximately 5 ns before the end of the S+H window regardless of device clock settings.
t_{LAT}	The time from the end of the S+H window until the ADC results latch in the ADCRESULTx register. If the ADCRESULTx register is read before this time, the previous conversion results are returned.
t_{EOC}	The time from the end of the S+H window until the S+H window for the next ADC conversion can begin. In 16-bit mode, this coincides with the latching of the conversion results, while in 12-bit mode, the subsequent sample can start before the conversion results are latched.
t_{INT}	The time from the end of the S+H window until an ADCINT flag is set (if configured). If the INTPULSEPOS bit in the ADCCTL1 register is set, t_{INT} coincides with the end of conversion (EOC) signal. If the INTPULSEPOS bit is 0, and the OFFSET field in the ADCINTCYCLE register is not 0, then there is a delay of OFFSET SYSCLK cycles before the ADCINT flag is set. This delay can be used to enter the ISR or trigger the DMA exactly when the sample is ready. If the INTPULSEPOS bit is 0, t_{INT} coincides with the end of the S+H window. If t_{INT} triggers a read of the ADC result register (directly through DMA or indirectly by triggering an ISR that reads the result), care must be taken to make sure the read occurs after the results latch (otherwise, the previous results are read).

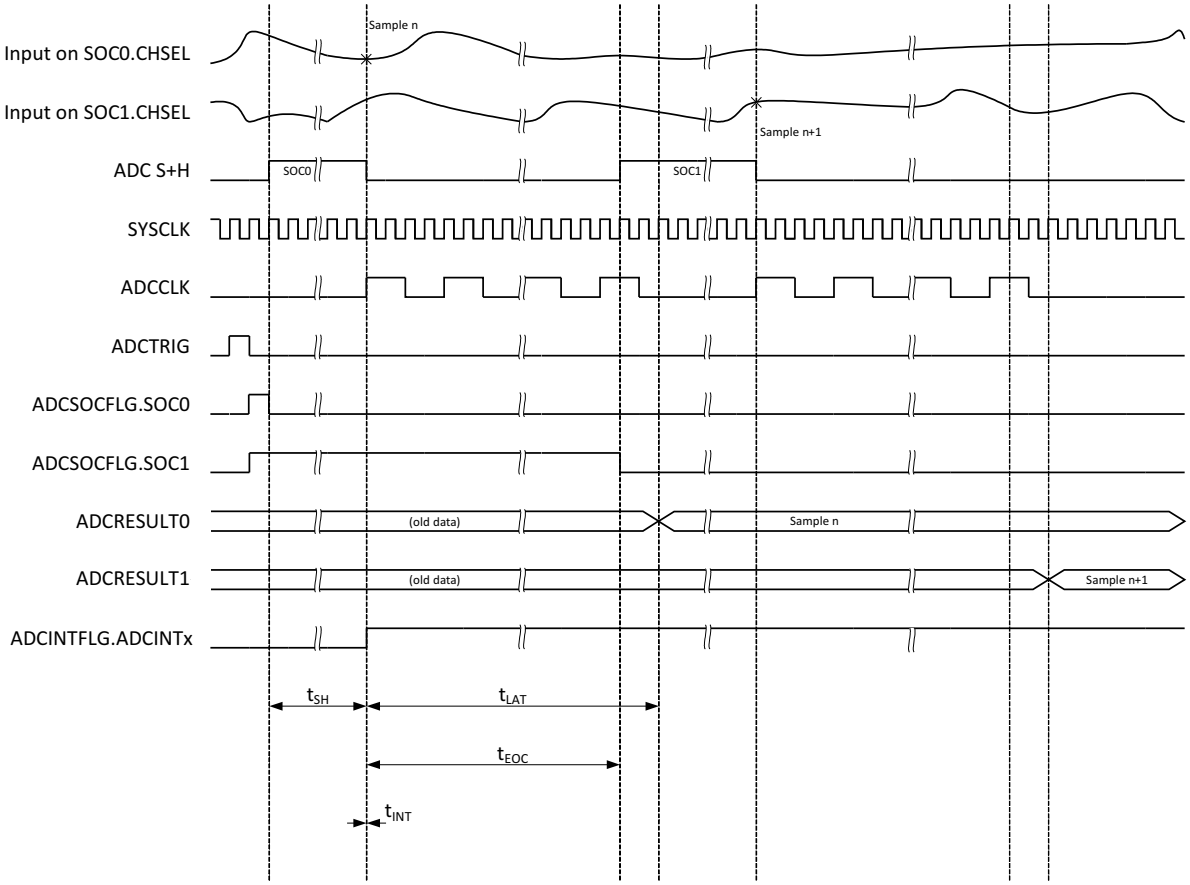


Figure 20-13. ADC Timings for 12-bit Mode in Early Interrupt Mode

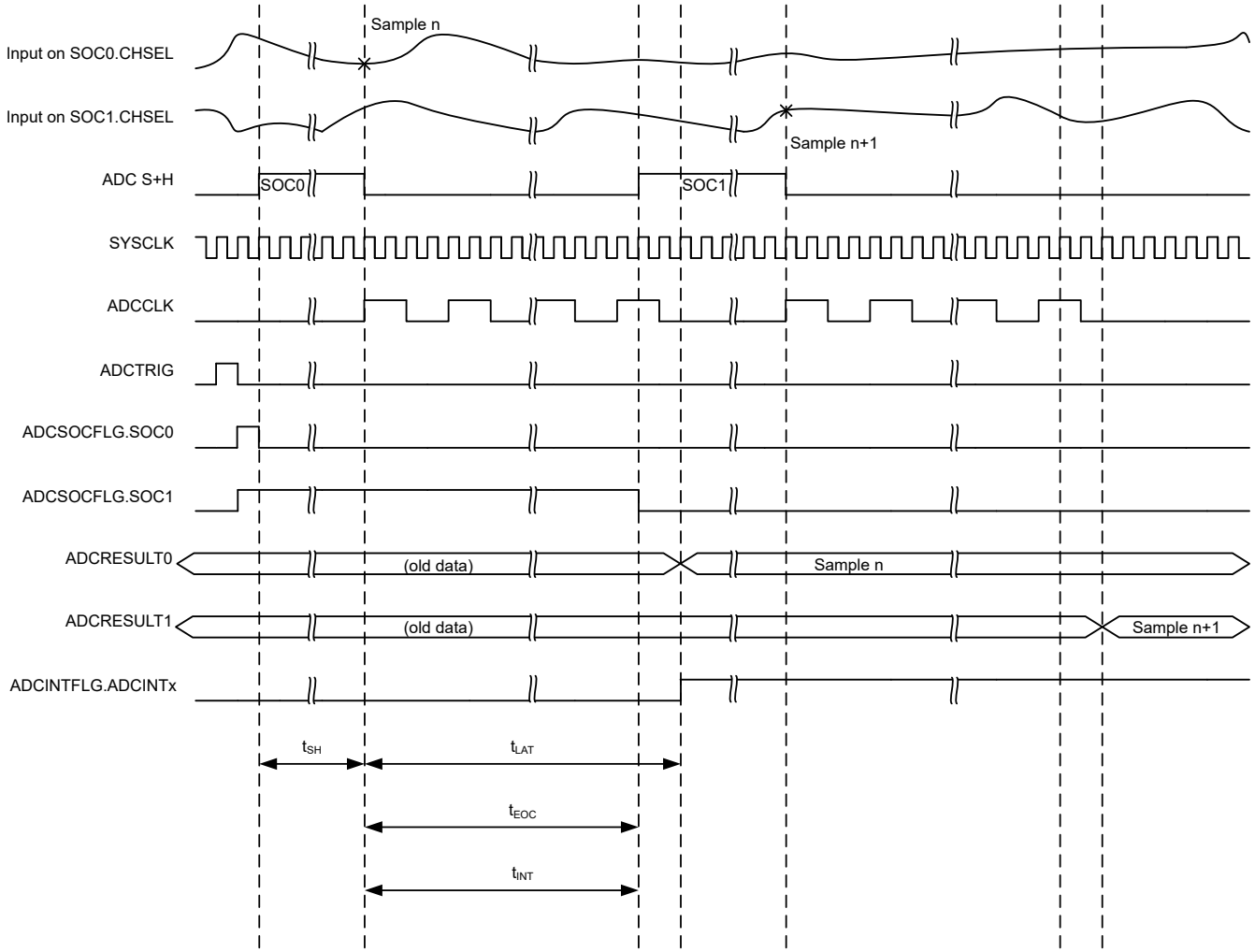


Figure 20-14. ADC Timings for 12-bit Mode in Late Interrupt Mode

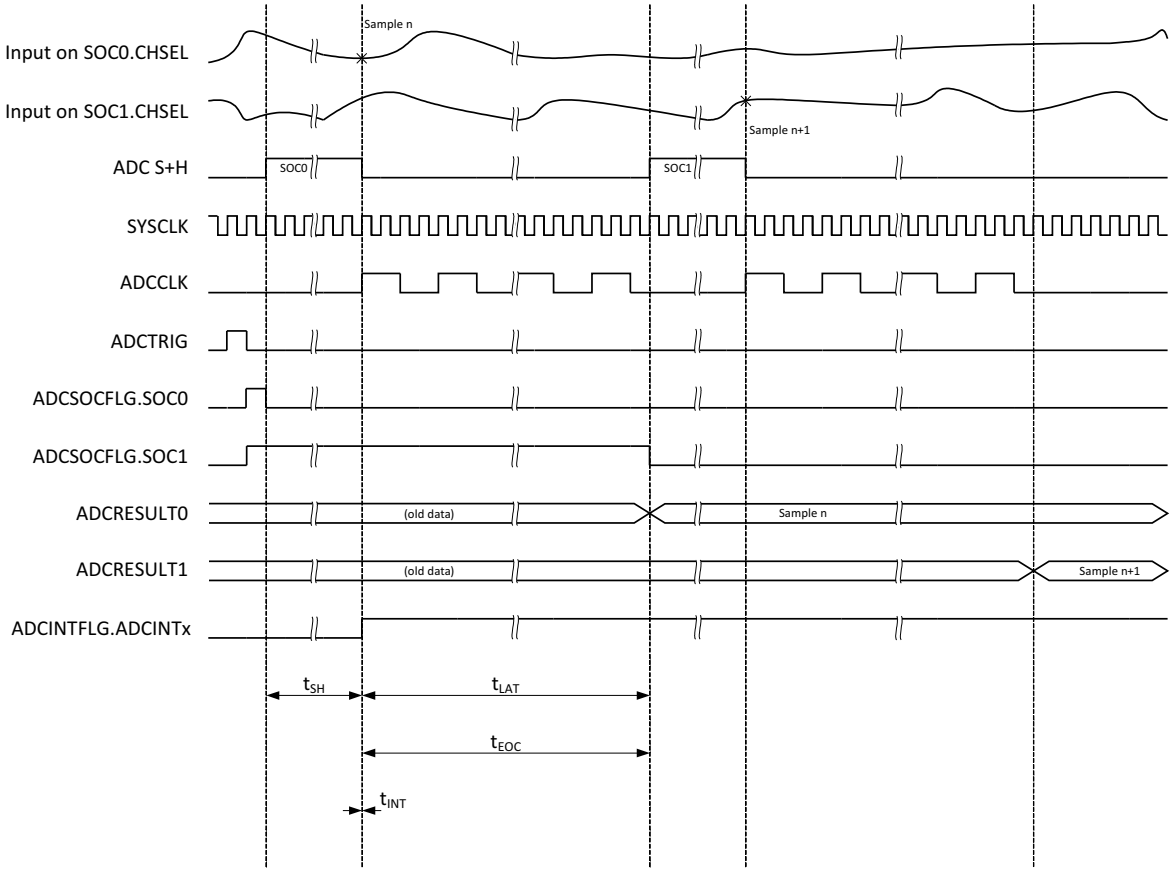


Figure 20-15. ADC Timings for 16-bit Mode in Early Interrupt Mode

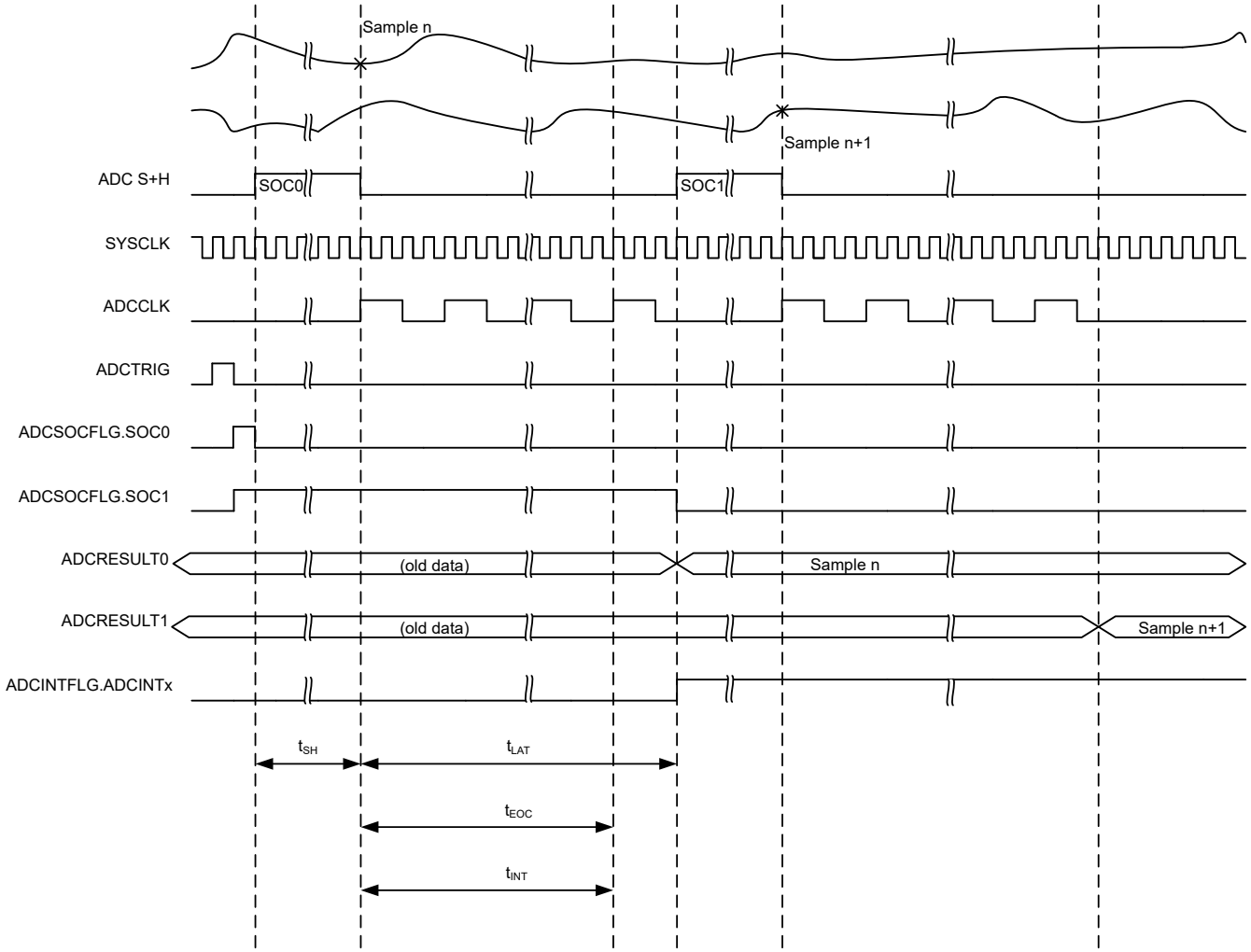


Figure 20-16. ADC Timings for 16-bit Mode in Late Interrupt Mode (SYSCLK Cycles)

Table 20-11. ADC Timings in 12-bit Mode

ADCCLK Prescale		SYSCLK Cycles			
ADCCTL2. PRESCALE	Prescale Ratio	t_{EOC}	t_{LAT} (1)	t_{INT} (Early)(2)	t_{INT} (Late)
0	1	11	13	0	11
2	2	21	23	0	21
3	2.5	26	28	0	26
4	3	31	34	0	31
5	3.5	36	39	0	36
6	4	41	44	0	41
7	4.5	46	49	0	46
8	5	51	55	0	51
9	5.5	56	60	0	56
10	6	61	65	0	61
11	6.5	66	70	0	66
12	7	71	76	0	71
13	7.5	76	81	0	76
14	8	81	86	0	81
15	8.5	86	91	0	86

(1) Refer to the "ADC: DMA Read of Stale Result" advisory in the *TMS320F2838x Real-Time MCUs Silicon Errata*.
(2) By default, t_{INT} occurs one SYSCLK cycle after the S+H window if INTPULSEPOS is 0. This can be changed by writing to the OFFSET field in the ADCINTCYCLE register.

Table 20-12. ADC Timings in 16-bit Mode

ADCCLK Prescale		SYSCLK Cycles			
ADCCTL2. PRESCALE	Prescale Ratio	t_{EOC}	t_{LAT} (1)	t_{INT} (Early)(2)	t_{INT} (Late)
0	1	31	32	0	31
2	2	60	61	0	60
3	2.5	75	75	0	75
4	3	90	91	0	90
5	3.5	104	106	0	104
6	4	119	120	0	119
7	4.5	134	134	0	134
8	5	149	150	0	149
9	5.5	163	165	0	163
10	6	178	179	0	178
11	6.5	193	193	0	193
12	7	208	209	0	208
13	7.5	222	224	0	222
14	8	237	238	0	237
15	8.5	252	252	0	252

(1) Refer to the "ADC: DMA Read of Stale Result" advisory in the *TMS320F2838x Real-Time MCUs Silicon Errata*.
(2) By default, t_{INT} occurs one SYSCLK cycle after the S+H window if INTPULSEPOS is 0. This can be changed by writing to the OFFSET field in the ADCINTCYCLE register.

20.13 Additional Information

The following sections contain additional practical information.

20.13.1 Ensuring Synchronous Operation

For best performance, all ADCs on the device must be operated synchronously. The device data manual specifies the performance in both synchronous and asynchronous mode for those parameters which differ between the modes of operation.

To make sure synchronous operation, all ADCs on the device must operate in lockstep. This is accomplished by writing configurations to all ADCs that cause the sampling and conversion phases of all ADCs to be exactly aligned. The easiest way to accomplish this is to write identical values to the SOC configurations for each ADC for trigger select and ACQPS (S+H duration). In addition, synchronous ADCs must also configure identical values for the SOC priority control, burst mode, burst trigger, and burst size.

20.13.1.1 Basic Synchronous Operation

The following example configures two SOCs each on ADCA and ADCB with identical trigger select and ACQPS values. This results in synchronous operation between ADCA and ADCB. For devices with more than two ADCs, the same principles can be used to synchronize all the ADCs.

```
Example: Basic Synchronous Operation
AdcaRegs.ADCSOC0CTL.bit.CHSEL = 4;      //SOC0 will convert ADCINA4
AdcaRegs.ADCSOC0CTL.bit.ACQPS = 19;     //SOC0 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC0CTL.bit.TRIGSEL = 10;   //SOC0 will begin conversion on ePWM3 SOCB
AdcbRegs.ADCSOC0CTL.bit.CHSEL = 0;      //SOC0 will convert ADCINB0
AdcbRegs.ADCSOC0CTL.bit.ACQPS = 19;     //SOC0 will use sample duration of 20 SYSCLK cycles
AdcbRegs.ADCSOC0CTL.bit.TRIGSEL = 10;   //SOC0 will begin conversion on ePWM3 SOCB

AdcaRegs.ADCSOC1CTL.bit.CHSEL = 4;      //SOC1 will convert ADCINA4
AdcaRegs.ADCSOC1CTL.bit.ACQPS = 30;     //SOC1 will use sample duration of 31 SYSCLK cycles
AdcaRegs.ADCSOC1CTL.bit.TRIGSEL = 10;   //SOC1 will begin conversion on ePWM3 SOCB
AdcbRegs.ADCSOC1CTL.bit.CHSEL = 1;      //SOC1 will convert ADCINB1
AdcbRegs.ADCSOC1CTL.bit.ACQPS = 30;     //SOC1 will use sample duration of 31 SYSCLK cycles
AdcbRegs.ADCSOC1CTL.bit.TRIGSEL = 10;   //SOC1 will begin conversion on ePWM3 SOCB
```

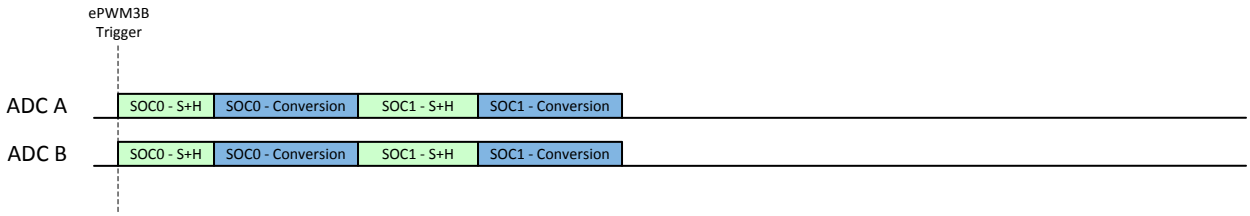


Figure 20-17. Example: Basic Synchronous Operation

Several things can be noted from Figure 20-17. First, while the ACQPS values must be the same for SOCs with the same number, different ACQPS values can be used for SOCs with different numbers. Because of this, synchronous operation does not require a single global S+H time, but instead only channels sampled simultaneously require identical S+H durations. Another important point from this example is that any channel select value can be used for any SOC. Finally, this example assumes round-robin operation. If high-priority SOCs are to be used, the priority must be configured the same on all ADCs.

20.13.1.2 Synchronous Operation with Multiple Trigger Sources

As long as each set of SOCs has identical trigger select and ACQPS settings, multiple trigger sources can be used while still achieving synchronous operation.

The following example demonstrates synchronous operation between ADCA and ADCB while using three SOCs and two trigger sources. Figure 20-18 demonstrates that any combination of relative trigger timings still results in synchronous operation.

```
Example: Synchronous Operation With Multiple Trigger Sources
AdcaRegs.ADCSOC0CTL.bit.CHSEL = 4;      //SOC0 will convert ADCINA4
AdcaRegs.ADCSOC0CTL.bit.ACQPS = 19;     //SOC0 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC0CTL.bit.TRIGSEL = 10;   //SOC0 will begin conversion on ePWM3 SOCB
AdcbRegs.ADCSOC0CTL.bit.CHSEL = 0;      //SOC0 will convert ADCINB0
AdcbRegs.ADCSOC0CTL.bit.ACQPS = 19;     //SOC0 will use sample duration of 20 SYSCLK cycles
AdcbRegs.ADCSOC0CTL.bit.TRIGSEL = 10;   //SOC0 will begin conversion on ePWM3 SOCB

AdcaRegs.ADCSOC1CTL.bit.CHSEL = 4;      //SOC1 will convert ADCINA4
AdcaRegs.ADCSOC1CTL.bit.ACQPS = 30;     //SOC1 will use sample duration of 31 SYSCLK cycles
AdcaRegs.ADCSOC1CTL.bit.TRIGSEL = 10;   //SOC1 will begin conversion on ePWM3 SOCB
AdcbRegs.ADCSOC1CTL.bit.CHSEL = 1;      //SOC1 will convert ADCINB1
AdcbRegs.ADCSOC1CTL.bit.ACQPS = 30;     //SOC1 will use sample duration of 31 SYSCLK cycles
AdcbRegs.ADCSOC1CTL.bit.TRIGSEL = 10;   //SOC1 will begin conversion on ePWM3 SOCB

AdcaRegs.ADCSOC2CTL.bit.CHSEL = 0;      //SOC2 will convert ADCINA0
AdcaRegs.ADCSOC2CTL.bit.ACQPS = 19;     //SOC2 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC2CTL.bit.TRIGSEL = 2;    //SOC2 will begin conversion on CPU Timer1
AdcbRegs.ADCSOC2CTL.bit.CHSEL = 2;      //SOC2 will convert ADCINB2
AdcbRegs.ADCSOC2CTL.bit.ACQPS = 19;     //SOC2 will use sample duration of 20 SYSCLK cycles
AdcbRegs.ADCSOC2CTL.bit.TRIGSEL = 2;    //SOC2 will begin conversion on CPU Timer1
```

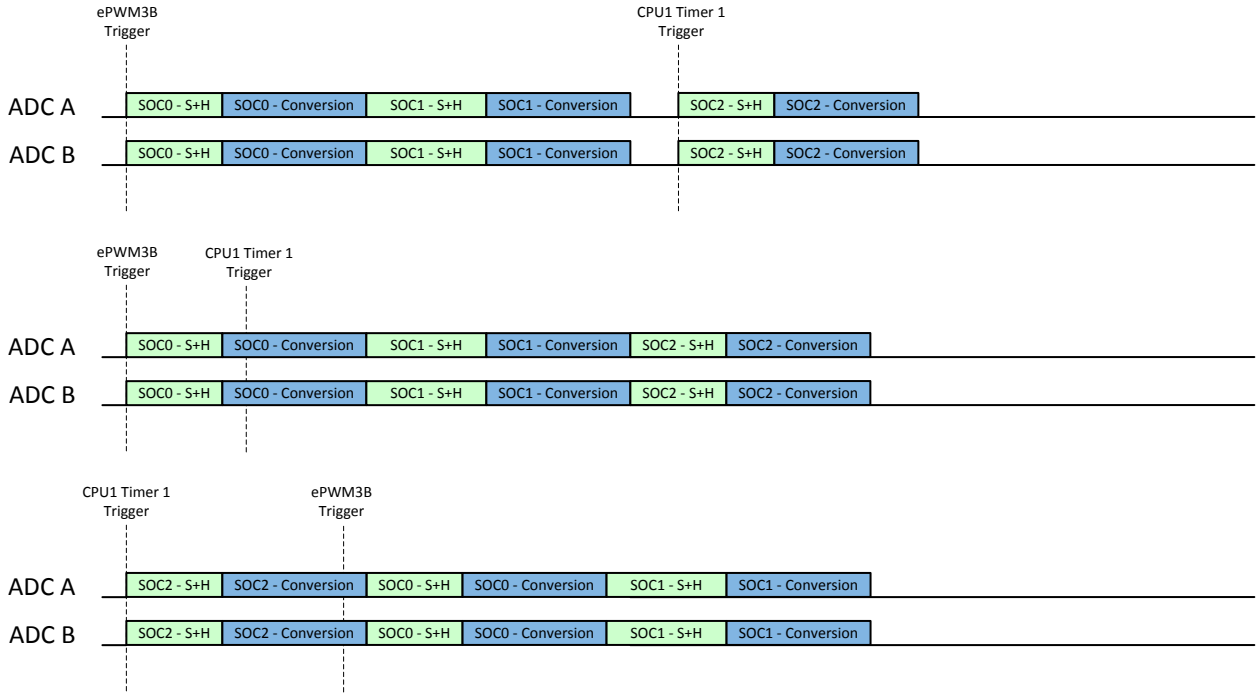


Figure 20-18. Example: Synchronous Operation with Multiple Trigger Sources

Note that any trigger source that can be selected in the TRIGSEL field can be used except for software triggering. There is no way to issue the software triggers for all ADCs simultaneously, so likely results in asynchronous operation. ADCINT1 or ADCINT2 can also be used as a trigger as long as the ADCINTSOCSEL1 and ADCINTSOCSEL2 registers are configured identically for all ADCs and software triggering is not used to start the chain of conversions.

20.13.1.3 Synchronous Operation with Uneven SOC Numbers

If only one trigger source is used, one ADC can use more SOCs than the other ADCs while still operating synchronously.

```
Example: Synchronous Operation With Uneven SOC Numbers
AdcaRegs.ADCSOC0CTL.bit.CHSEL = 4;      //SOC0 will convert ADCINA4
AdcaRegs.ADCSOC0CTL.bit.ACQPS = 19;     //SOC0 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC0CTL.bit.TRIGSEL = 10;   //SOC0 will begin conversion on ePWM3 SOCB
AdcbRegs.ADCSOC0CTL.bit.CHSEL = 0;      //SOC0 will convert ADCINB0
AdcbRegs.ADCSOC0CTL.bit.ACQPS = 19;     //SOC0 will use sample duration of 20 SYSCLK cycles
AdcbRegs.ADCSOC0CTL.bit.TRIGSEL = 10;   //SOC0 will begin conversion on ePWM3 SOCB

AdcaRegs.ADCSOC1CTL.bit.CHSEL = 4;      //SOC1 will convert ADCINA4
AdcaRegs.ADCSOC1CTL.bit.ACQPS = 30;     //SOC1 will use sample duration of 31 SYSCLK cycles
AdcaRegs.ADCSOC1CTL.bit.TRIGSEL = 10;   //SOC1 will begin conversion on ePWM3 SOCB
AdcbRegs.ADCSOC1CTL.bit.CHSEL = 1;      //SOC1 will convert ADCINB1
AdcbRegs.ADCSOC1CTL.bit.ACQPS = 30;     //SOC1 will use sample duration of 31 SYSCLK cycles
AdcbRegs.ADCSOC1CTL.bit.TRIGSEL = 10;   //SOC1 will begin conversion on ePWM3 SOCB

AdcaRegs.ADCSOC2CTL.bit.CHSEL = 0;      //SOC2 will convert ADCINA0
AdcaRegs.ADCSOC2CTL.bit.ACQPS = 19;     //SOC2 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC2CTL.bit.TRIGSEL = 10;   //SOC2 will begin conversion on ePWM3 SOCB
```

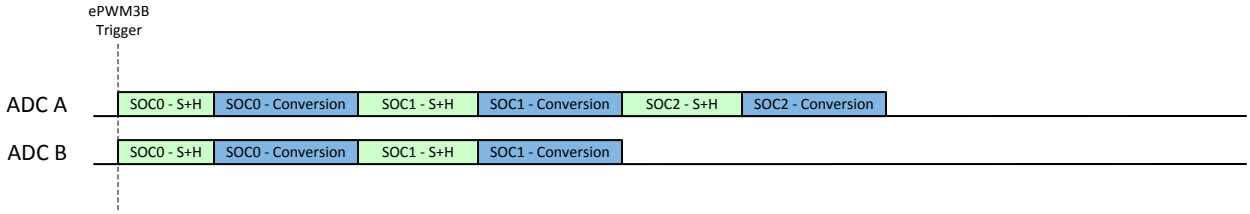


Figure 20-19. Example: Synchronous Operation with Uneven SOC Numbers

Note that if the trigger comes again before all SOCs have completed their conversions, ADCB begins converting immediately on SOC0 while ADCA does not start converting SOC0 again until SOC2 is complete. This results in asynchronous operation, so care must be taken to not overflow the trigger.

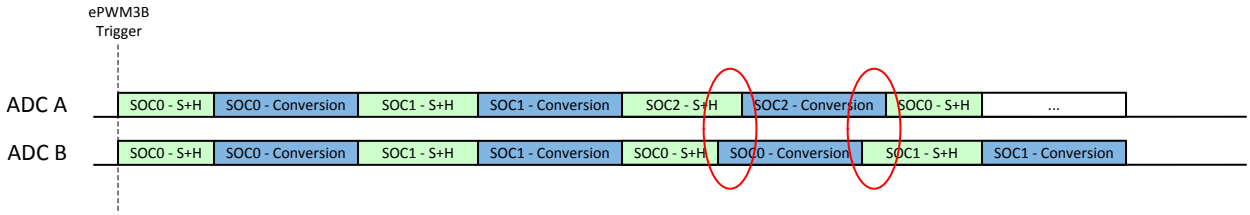


Figure 20-20. Example: Asynchronous Operation with Uneven SOC Numbers – Trigger Overflow

20.13.1.4 Synchronous Operation with Different Resolutions

Configuring different ADCs to use different resolutions results in asynchronous operation. This occurs because the conversion time for 12-bit mode and 16-bit mode are different. Synchronous operation requires both the start and end of the conversion phase to be aligned, so even using the same S+H window duration does not result in synchronous operation.

```
Example: Asynchronous Operation with Different Resolutions
//ADCA = 12-bit mode
AdcaRegs.ADCSOC0CTL.bit.CHSEL = 4;      //SOC0 will convert ADCINA4
AdcaRegs.ADCSOC0CTL.bit.ACQPS = 50;     //SOC0 will use sample duration of 51 SYSCLK cycles
AdcaRegs.ADCSOC0CTL.bit.TRIGSEL = 10;   //SOC0 will begin conversion on ePWM3 SOCB

//ADCB = 16-bit mode
AdcbRegs.ADCSOC0CTL.bit.CHSEL = 0;      //SOC0 will convert ADCINB0/B1
AdcbRegs.ADCSOC0CTL.bit.ACQPS = 50;     //SOC0 will use sample duration of 51 SYSCLK cycles
AdcbRegs.ADCSOC0CTL.bit.TRIGSEL = 10;   //SOC0 will begin conversion on ePWM3 SOCB
```

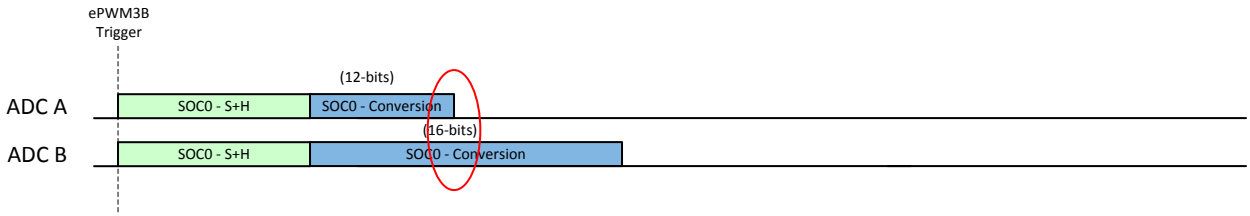


Figure 20-21. Example: Asynchronous Operation with Different Resolutions

To achieve synchronous operation while using both 12-bit and 16-bit resolution, conversions must be done in parallel at one resolution. Once conversions are complete at one resolution, the CPU must switch the resolution on all ADCs and then cause another trigger (this trigger must not be a software SOC force, as all ADCs cannot be started simultaneously using this method).

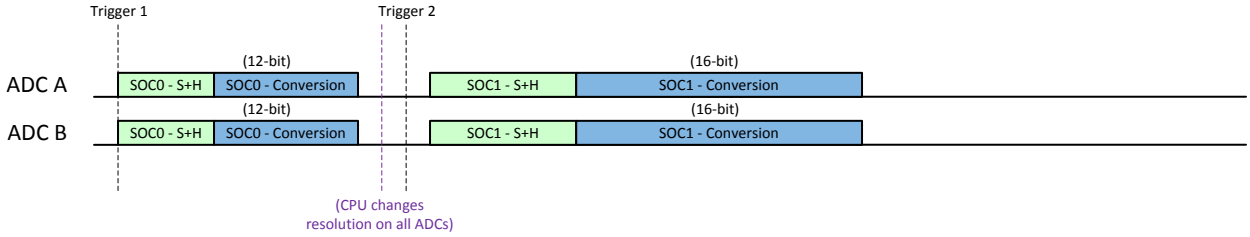


Figure 20-22. Example: Synchronous Operation with Different Resolutions

20.13.1.5 Non-overlapping Conversions

If conversion timings can be assured to not overlap by the user, then it is not necessary to configure all SOCs identically on all ADCs to achieve performance equivalent to synchronous operation. For example, if the two ADC triggers in a system come from two ePWM sources that are always 180-degrees out-of-phase, then SOC0 can be used for both ADCA and ADCB with different trigger sources and different ACQPS values.

```
Example: Operation with Non-overlapping Conversions
//ePWM3 SOCA and SOCB are 180 degrees out of phase
AdcaRegs.ADCSOC0CTL.bit.CHSEL = 4;      //SOC0 will convert ADCINA4
AdcaRegs.ADCSOC0CTL.bit.ACQPS = 19;     //SOC0 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC0CTL.bit.TRIGSEL = 10;   //SOC0 will begin conversion on ePWM3 SOCB
AdcbRegs.ADCSOC0CTL.bit.CHSEL = 0;      //SOC0 will convert ADCINB0
AdcbRegs.ADCSOC0CTL.bit.ACQPS = 19;     //SOC0 will use sample duration of 20 SYSCLK cycles
AdcbRegs.ADCSOC0CTL.bit.TRIGSEL = 9;    //SOC0 will begin conversion on ePWM3 SOCA
```

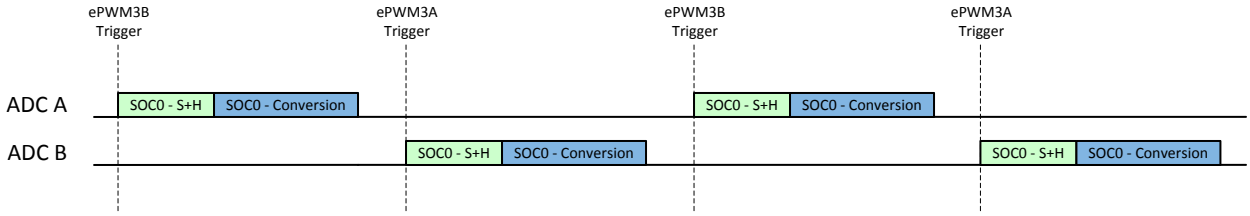


Figure 20-23. Example: Synchronous Equivalent Operation with Non-Overlapping Conversions

20.13.2 Choosing an Acquisition Window Duration

For correct operation, the input signal to the ADC must be allowed adequate time to charge the sample and hold capacitor, Ch. Typically, the S+H duration is chosen such that the sampling capacitor is charged to within ½ LSB or ¼ LSB of the final value, depending on the tolerable settling error.

The best methodology to determine the required settling time is to simulate the ADC and ADC driving circuits to make sure adequate settling performance. See *ADC Input Circuit Evaluation for C2000 MCUs* and *Charge-Sharing Driving Circuits for C2000 ADCs* for additional guidance on ADC signal conditioning circuit design and evaluation.

An approximation of the required settling time can also be determined using an RC settling model. The time constant for the model is given by the equation:

$$\tau = (R_S + R_{on}) \times C_h + R_S \times \left(C_S + C_p\right) \quad (2)$$

And the number of time constants needed is given by the equation:

$$k = ln\left(\frac{2^n}{settling\ error}\right) - ln\left(\frac{C_S + C_P}{CH}\right) \quad (3)$$

So the total S+H time must be set to at least:

$$t = k \cdot \tau \quad (4)$$

Where the following parameters are provided by the ADC input model in the device data manual:

- n = ADC resolution (in bits)
- R_{ON} = ADC sampling switch resistance (provided in Ω)
- C_H = ADC sampling capacitor (provided in pF)
- C_p = ADC channel parasitic input capacitance (provided in pF)

And the following parameters are dependent on the application design:

- settling error = tolerable settling error (in LSBs)
- R_s = ADC driving circuit source impedance (typically in Ω or kΩ)
- C_S = capacitance on ADC input pin (typically in pF or nF)

For example, assuming the following parameters:

- n = 12-bits
- R_{ON} = 500 Ω
- C_H = 12.5 pF
- C_p = 12.7 pF
- settling error = ¼ LSB
- R_s = 180 Ω
- C_s = 150 pF

The time constant is calculated as:

$$\tau = (180\Omega + 500\Omega) \times 12.5pF + 180\Omega \times (150pF + 12.7pF) = 37.8ns \quad (5)$$

And the number of required time constants is:

$$k = ln\left(\frac{2^{12}}{0.25}\right) - ln\left(\frac{150pF + 12.7pF}{12.5pF}\right) = 9.70 - 2.57 = 7.13 \quad (6)$$

So the S+H time must be set to at least: 37.8 ns × 7.13 = 270 ns

If SYSCLK = 200 MHz , then each SYSCLK cycle is 5 ns. S+H duration is 270 ns/5 ns = 54 SYSCLK cycles, so ACQPS for this input is set to at least CEILING(54.0) – 1 = 53.

While this gives a rough estimate of the required acquisition window, a better method is to setup a circuit with the ADC input model, a model of the source impedance/capacitance, and any board parasitics in SPICE (or similar software) and simulate to verify that the sampling capacitor settles to the desired accuracy.

Note

The device data manual specifies a minimum ADC S+H window duration. Do not use an ACQPS value that gives a duration less than this specification.

20.13.3 Achieving Simultaneous Sampling

While each ADC does not have dual S+H circuits, it is easy to achieve simultaneous sampling. This is accomplished by setting the SOC triggers on two or more ADC modules to use the same trigger source. The following example demonstrates simultaneous sampling on 4 ADCs based on an ePWM3 event. ADCINA3, ADCINB5, ADCINC5, and ADCIND2 are sampled. An acquisition window of 20 SYSCLK cycles is used, but different durations are possible.

```
AdcaRegs.ADCSOC0CTL.bit.CHSEL = 3;        //SOC0 will convert ADCINA3
AdcaRegs.ADCSOC0CTL.bit.ACQPS = 19;       //SOC0 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC0CTL.bit.TRIGSEL = 10;     //SOC0 will begin conversion on ePWM3 SOCB
AdcbRegs.ADCSOC0CTL.bit.CHSEL = 5;        //SOC0 will convert ADCINB5
AdcbRegs.ADCSOC0CTL.bit.ACQPS = 19;       //SOC0 will use sample duration of 20 SYSCLK cycles
AdcbRegs.ADCSOC0CTL.bit.TRIGSEL = 10;     //SOC0 will begin conversion on ePWM3 SOCB
AdccRegs.ADCSOC0CTL.bit.CHSEL = 5;        //SOC0 will convert ADCINC5
AdccRegs.ADCSOC0CTL.bit.ACQPS = 19;       //SOC0 will use sample duration of 20 SYSCLK cycles
AdccRegs.ADCSOC0CTL.bit.TRIGSEL = 10;     //SOC0 will begin conversion on ePWM3 SOCB
AdcdRegs.ADCSOC0CTL.bit.CHSEL = 2;        //SOC0 will convert ADCIND2
AdcdRegs.ADCSOC0CTL.bit.ACQPS = 19;       //SOC0 will use sample duration of 20 SYSCLK cycles
AdcdRegs.ADCSOC0CTL.bit.TRIGSEL = 10;     //SOC0 will begin conversion on ePWM3 SOCB
```

When the ePWM3 trigger is received, all four ADCs begin converting in parallel immediately. All results are stored in the ADCRESULT0 register for each ADC. Note that this assumes that all ADCs are idle when the trigger is received. If one or more ADCs is busy, the samples do not happen at exactly the same time.

20.13.4 Result Register Mapping

The ADC results and the ADC PPB results are duplicated for each memory bus controller in the system. Bus controllers include all CPUs, DMAs, and CLAs present on the specific part family and part number. For each bus controller, no access configuration is needed to allow read access to the result registers and no contention occurs in cases where multiple bus controllers try to read the ADC results simultaneously.

20.13.5 Internal Temperature Sensor

The internal temperature sensor measures the junction temperature of the device. The output of the sensor can be sampled with the ADC through an internal connection. This can be enabled on channel ADCIN13 on ADCA by setting the ENABLE bit in the TSNSCTL register.

To convert the temperature sensor reading into a temperature, pass the temperature sensor reading to the GetTemperatureC() function in F2838x_TempSensorConv.c.

Note

To sample the temperature sensor, the ADC must be in single-ended 12-bit mode.

If the temperature sensor is sampled in 16-bit mode, the internal temperature sensor is sampled with a 12-bit resolution, regardless of the ADC mode currently configured. This does not change the resolution of results obtained on other SOCs for the given ADC.

20.13.6 Designing an External Reference Circuit

Figure 20-24 shows the basic organization of the external voltage reference generation circuitry. A single reference voltage generation source must be shared by all ADC modules. This minimizes reference voltage mismatch between ADC modules. The reference voltage must then be buffered by a precision op-amp with good bandwidth and low output impedance before being driven into the reference pin. A capacitor between the high and low reference pins must be placed on the PCB as close to the pins as practical to help absorb high-frequency currents. A series resistor (typically <1Ω) in series with this capacitor can be necessary to make sure op-amp stability.

It is also possible to share two reference pins between one op-amp driver. This organization is shown in Figure 20-25. This gives slightly reduced performance compared to the case where each reference pin has a dedicated op-amp buffer, but it can still be possible to achieve all ADC specifications in the data sheet.

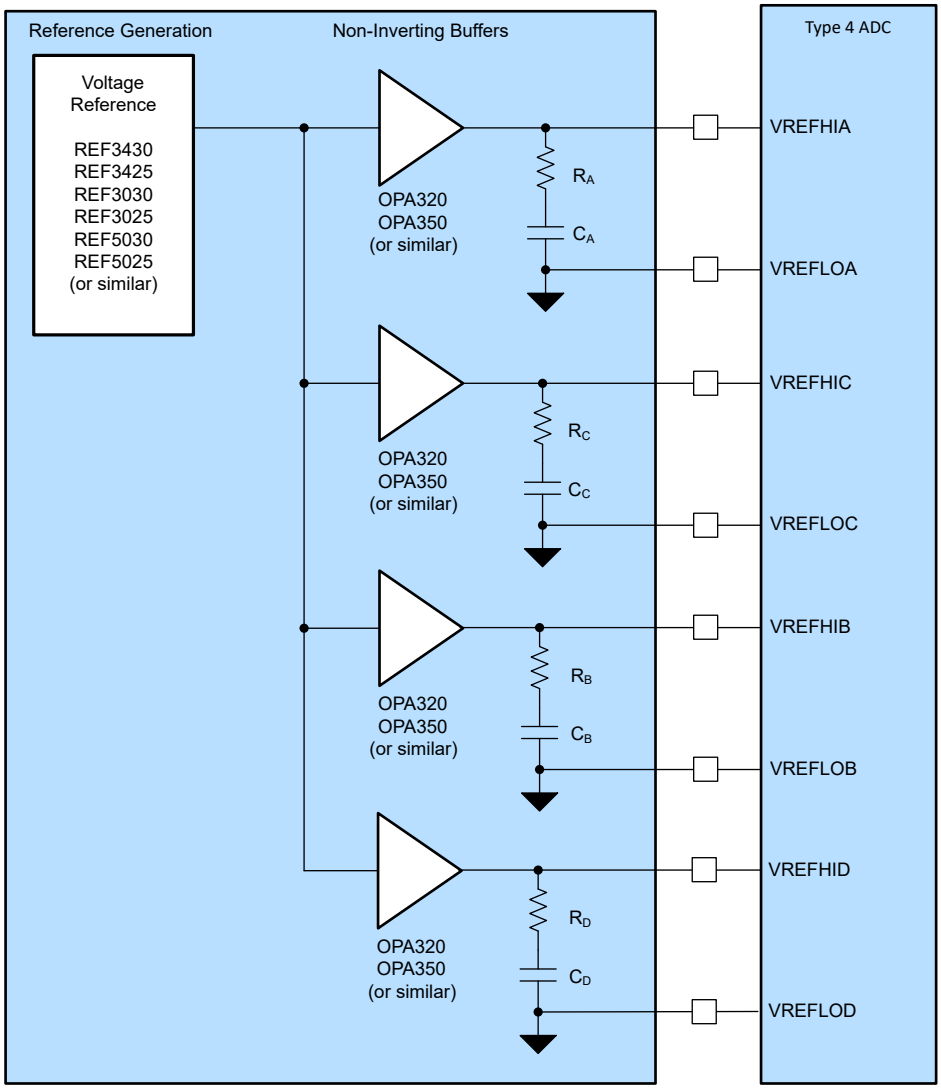


Figure 20-24. ADC Reference System

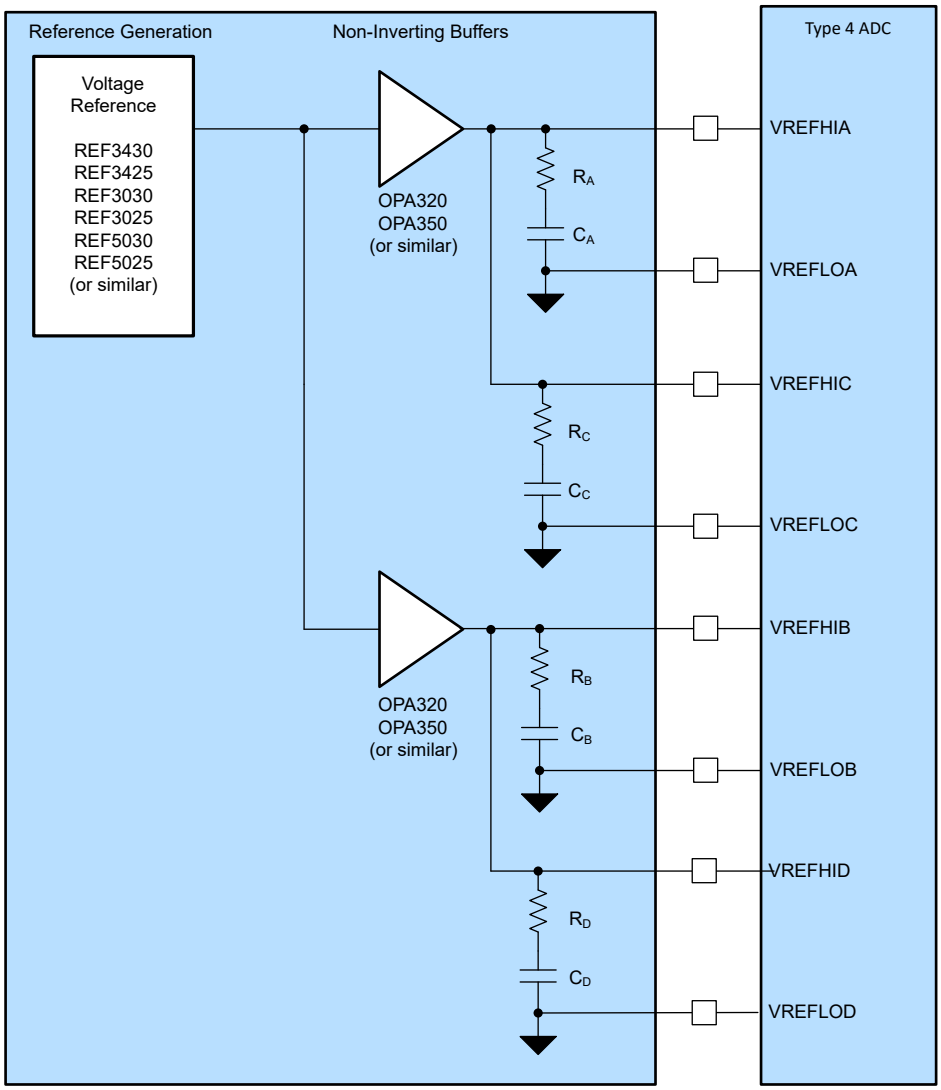


Figure 20-25. ADC Shared Reference System

20.14 Software

20.14.1 ADC Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/adc

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

20.14.1.1 ADC Software Triggering

FILE: adc_ex1_soc_software.c

This example converts some voltages on ADCA and ADCC based on a software trigger.

The ADCC will not convert until ADCA is complete, so the ADCs will not run asynchronously. However, this is much less efficient than allowing the ADCs to convert synchronously in parallel (for example, by using an ePWM trigger).

External Connections

- A0, A1, C2, and C3 should be connected to signals to convert

Watch Variables

- *myADC0Result0* - Digital representation of the voltage on pin A0
- *myADC0Result1* - Digital representation of the voltage on pin A1
- *myADC1Result0* - Digital representation of the voltage on pin C2
- *myADC1Result1* - Digital representation of the voltage on pin C3

20.14.1.2 ADC ePWM Triggering

FILE: adc_ex2_soc_epwm.c

This example sets up ePWM1 to periodically trigger a conversion on ADCA.

External Connections

- A0 should be connected to a signal to convert

Watch Variables

- *myADC0Results* - A sequence of analog-to-digital conversion samples from pin A0. The time between samples is determined based on the period of the ePWM timer.

20.14.1.3 ADC Temperature Sensor Conversion

FILE: adc_ex3_temp_sensor.c

This example sets up the ePWM to periodically trigger the ADC. The ADC converts the internal connection to the temperature sensor, which is then interpreted as a temperature by calling the ADC_getTemperatureC() function.

Watch Variables

- *sensorSample* - The raw reading from the temperature sensor
- *sensorTemp* - The interpretation of the sensor sample as a temperature in degrees Celsius.

20.14.1.4 ADC Synchronous SOC Software Force (adc_soc_software_sync)

FILE: adc_ex4_soc_software_sync.c

This example converts some voltages on ADCA and ADCC using input 5 of the input X-BAR as a software force. Input 5 is triggered by toggling GPIO0, but any spare GPIO could be used. This method will ensure that both ADCs start converting at exactly the same time.

External Connections

- A2, A3, C2, C3 pins should be connected to signals to convert

Watch Variables

- *myADC0Result0* : a digital representation of the voltage on pin A2

- *myADC0Result1* : a digital representation of the voltage on pin A3
- *myADC1Result0* : a digital representation of the voltage on pin C2
- *myADC1Result1* : a digital representation of the voltage on pin C3

20.14.1.5 ADC Continuous Triggering (adc_soc_continuous)

FILE: adc_ex5_soc_continuous.c

This example sets up the ADC to convert continuously, achieving maximum sampling rate.
External Connections

- A0 pin should be connected to signal to convert

Watch Variables

- *adcAResults* - A sequence of analog-to-digital conversion samples from pin A0. The time between samples is the minimum possible based on the ADC speed.

20.14.1.6 ADC Continuous Conversions Read by DMA (adc_soc_continuous_dma)

FILE: adc_ex6_soc_continuous_dma.c

This example sets up two ADC channels to convert simultaneously. The results will be transferred by the DMA into a buffer in RAM.

External Connections

- A3 & C3 pins should be connected to signals to convert

Watch Variables

- *myADC0DataBuffer* : a digital representation of the voltage on pin A3
- *myADC1DataBuffer*: a digital representation of the voltage on pin C3

20.14.1.7 ADC PPB Offset (adc_ppb_offset)

FILE: adc_ex7_ppb_offset.c

This example software triggers the ADC. Some SOCs have automatic offset adjustment applied by the post-processing block. After the program runs, the memory will contain ADC & post-processing block(PPB) results.

External Connections

- A2, C2 pins should be connected to signals to convert

Watch Variables

- *myADC0Result* : a digital representation of the voltage on pin A2
- *myADC0PPBResult* : a digital representation of the voltage on pin A2, minus 100 LSBs of automatically added offset
- *myADC1Result* : a digital representation of the voltage on pin C2
- *myADC1PPBResult* : a digital representation of the voltage on pin C2 plus 100 LSBs of automatically added offset

20.14.1.8 ADC PPB Limits (adc_ppb_limits)

FILE: adc_ex8_ppb_limits.c

This example sets up the ePWM to periodically trigger the ADC. If the results are outside of the defined range, the post-processing block will generate an interrupt.

The default limits are 1000LSBs and 3000LSBs. With VREFHI set to 3.3V, the PPB will generate an interrupt if the input voltage goes above about 2.4V or below about 0.8V.

External Connections

- A0 should be connected to a signal to convert

Watch Variables

- None

20.14.1.9 ADC PPB Delay Capture (adc_ppb_delay)

FILE: adc_ex9_ppb_delay.c

This example demonstrates delay capture using the post-processing block.

Two asynchronous ADC triggers are setup:
- ePWM1, with period 2048, triggering SOC0 to convert on pin A0
- ePWM2, with period 9999, triggering SOC1 to convert on pin A2
  Each conversion generates an ISR at the end of the conversion. In the ISR for SOC0, a conversion counter is incremented and the PPB is checked to determine if the sample was delayed.
  After the program runs, the memory will contain:

conversion : the sequence of conversions using SOC0 that were delayed
- *delay* : the corresponding delay of each of the delayed conversions

20.14.1.10 ADC ePWM Triggering Multiple SOC

FILE: adc_ex10_multiple_soc_epwm.c

This example sets up ePWM1 to periodically trigger a set of conversions on ADCA and ADCC. This example demonstrates multiple ADCs working together to process of a batch of conversions using the available parallelism across multiple ADCs.

ADCA Interrupt ISRs are used to read results of both ADCA and ADCC.

External Connections
- A0, A1, A2 and C2, C3, C4 pins should be connected to signals to be converted.

Watch Variables
- *adcAResult0* - Digital representation of the voltage on pin A0
- *adcAResult1* - Digital representation of the voltage on pin A1
- *adcAResult2* - Digital representation of the voltage on pin A2
- *adcCResult0* - Digital representation of the voltage on pin C2
- *adcCResult1* - Digital representation of the voltage on pin C3
- *adcCResult2* - Digital representation of the voltage on pin C4

20.14.1.11 ADC Burst Mode

FILE: adc_ex11_burst_mode_epwm.c

This example sets up ePWM1 to periodically trigger ADCA using burst mode. This allows for different channels to be sampled with each burst.

Each burst triggers 3 conversions. A0 and A1 are part of every burst while the third conversion rotates between A2, A3, and A4. This allows high importance signals to be sampled at high speed while lower priority signals can be sampled at a lower rate.

ADCA Interrupt ISRs are used to read results for ADCA.

External Connections
- A0, A1, A2, A3, A4

Watch Variables
- *adcAResult0* - Digital representation of the voltage on pin A0
- *adcAResult1* - Digital representation of the voltage on pin A1
- *adcAResult2* - Digital representation of the voltage on pin A2
- *adcAResult3* - Digital representation of the voltage on pin A3
- *adcAResult4* - Digital representation of the voltage on pin A4

20.14.1.12 ADC Burst Mode Oversampling

FILE: adc_ex12_burst_mode_oversampling.c

This example is an ADC oversampling example implemented with software. The ADC SOCs are configured in burst mode, triggered by the ePWM SOC A event trigger.

External Connection

- A2

Watch Variables

- *Iv_results* - Array of digital values measured on pin A2 (oversampling is configured by Oversampling_Amount)

20.14.1.13 ADC SOC Oversampling

FILE: adc_ex13_soc_oversampling.c

This example sets up ePWM1 to periodically trigger a set of conversions on ADCA including multiple SOCs that all convert A2 to achieve oversampling on A2.

ADCA Interrupt ISRs are used to read results of ADCA.

External Connections

- A0, A1, A2 should be connected to signals to be converted.

Watch Variables

- *adcAResult0* - Digital representation of the voltage on pin A0
- *adcAResult1* - Digital representation of the voltage on pin A1
- *adcAResult2* - Digital representation of the voltage on pin A2

20.14.1.14 ADC PPB PWM trip (adc_ppb_pwm_trip)

FILE: adc_ex14_ppb_pwm_trip.c

This example demonstrates EPWM tripping through ADC limit detection PPB block. ADCAINT1 is configured to periodically trigger the ADCA channel 2 post initial software forced trigger. The limit detection post-processing block(PPB) is configured and if the ADC results are outside of the defined range, the post-processing block will generate an ADCxEVTy event. This event is configured as EPWM trip source through configuring EPWM XBAR and corresponding EPWM's trip zone and digital compare sub-modules. The example showcases

- one-shot
- cycle-by-cycle
- and direct tripping of PWMs through ADCAEVT1 source via Digital compare submodule.

The default limits are 0LSBs and 3600LSBs. With VREFHI set to 3.3V, the PPB will generate a trip event if the input voltage goes above about 2.9V.

External Connections

- A2 should be connected to a signal to convert
- Observe the following signals on an oscilloscope
  - ePWM1(GPIO0 - GPIO1)
  - ePWM2(GPIO2 - GPIO3)
  - ePWM3(GPIO4 - GPIO5)
-

Watch Variables

- adcA2Results - digital representation of the voltage on pin A2

20.14.1.15 ADC High Priority SOC (adc_high_priority_soc)

FILE: adc_ex15_high_priority_soc.c

This example demonstrates configuration of ADC high priority SOCs in order to sample fastest control loop signals with high priority while low priority signals are sampled with default round robin priority.

ADC PPB block is configured to capture delay between SOC trigger and actual start of the SOC sampling in order to quantify the jitters in sampling high priority signals due to low priority signals. The delay in processing an

SOC is captured in ADCPPBxSTAMP.DLYSTAMP register field and the total delay in sampling an SOC is equal to (DLYSTAMP - 2) cycles. In an optimal design the high priority SOC is expected to have less delay between SOC trigger and actual start of the sample.

In this example ADCA, ADCB and ADCD are configured to sample both high and low priority signals. SOC0-3 are configured as high priority SOCs while rest are configured with default round-robin priority. ADCA SOC0-3 are configured as high priority SOCs sampling channels A0-A3, ADCB SOC0-1 are configured as high priority SOCs sampling channels B0-B1 and ADCD SOC0-1 are configured as high priority SOCs sampling channels D0-D1. For sampling low priority signals SOC4-SOC5 are configured sampling channel 4 and channel 13 respectively for ADCA and channel 4 and 5 for ADCB and ADCD. For ADCA, channel 13 is connected to internal temperature sensor output and hence no signal needs to be connected to channel A13. High priority SOC results are read in ADCINT1 ISR while low priority SOC results are read in idle loop.

This example has two modes of operation as follows. Desired mode can be selected by configuring the EX_ADC_LP_SOC_TRIGGER macro accordingly. Mode 0: ADCINT as round robin SOC trigger Mode 1: EPWM2 as round robin SOC trigger

In mode 0, EPWM1 is configured as trigger for high priority SOCs while ADCINT1 is configured as low priority SOC trigger. ADCINT1 is configured to be triggered on completion of SOC1 conversion which in turn trigger low priority SOCs 4 and 5 and ADCINT2 is configured to be triggered on completion of SOC5. ADC result for high priority SOCs are read in ADCINT ISR while low priority SOC results are read in idle loop. ADCINT3 is configured to be triggered on completion of SOC3 and hence SOC2-SOC3 results for ADCA are read post checking if the conversion is complete in ADCINT1 ISR.

In mode 0, SOC0-SOC1 for all ADCs will experience minimal delay(0 and 1 conversions) in processing due to the high priority configuration. For ADCA, SOC4-SOC5 are triggered when SOC2-SOC3 conversion is ongoing, hence SOC4-SOC5 will see some delay(2 and 3 conversions) in processing as expected. For ADCB and ADCD, SOC4-SOC5 will see minimal delay (0 and 1 conversions) in processing as expected.

In mode 1, EPWM1 is configured as trigger for high priority SOCs while EPWM2 is configured as low priority SOC trigger. ADCINT1 is configured to be triggered on completion of SOC3 conversion and ADC result for high priority SOCs are read in the ADCINT ISR. Low priority SOCs 4 and 5 are triggered through EPWM2 and ADCINT2 is configured to be triggered on completion of SOC5. The result for low priority SOCs are read in background loop. Since, SOC4-5 are triggered post SOC2-3 conversion(due to configured EPWM1 and EPWM2 trigger frequency and duty), SOC4-SOC5 will have minimal delay(0 and 1 conversions respectively) in SOC processing as expected.

Optimization Level: Example is expected to run with opt level = O2.

To view ADC results, put breakpoint at the statement where indexB is reset to zero in idle loop.

External Connections

- A0-A4, B0-B1, B4-B5, D0-D1 and D4-D5 pins should be connected to signals to be converted.
- Observe ePWM1, ePWM2 signals on oscilloscope

Watch Variables

ADC Results:

- adcA0Results - adcA4Results - ADC result of channels A0-A4
- adcA13Results - ADC result of channels A13(Temp Sensor output)
- adcB0Results - adcB1Results - ADC result of channels B0-B1
- adcB4Results - ADC result of channel B4
- adcB5Results - ADC result of channel B5
- adcD0Results - adcD1Results - ADC result of channels D0-D1
- adcD4Results - ADC result of channel D4
- adcD5Results - ADC result of channel D5

SOC conversion delays:

- delaySocA0 - Delay in sampling ADCA SOC0
- delaySocA3 - Delay in sampling ADCA SOC3

- delaySocA4-delaySocA5 - Delay in sampling ADCA SOC4-SOC5
- delaySocB0-delaySocB1 - Delay in sampling ADCB SOC0-SOC1
- delaySocB4-delaySocB5 - Delay in sampling ADCB SOC4-SOC5
- delaySocD0-delaySocD1 - Delay in sampling ADCD SOC0-SOC1
- delaySocD4-delaySocD5 - Delay in sampling ADCD SOC4-SOC5

20.14.1.16 ADC Interleaved Averaging in Software

FILE: adc_ex16_sw_interleaved_averaging.c

This example demonstrates software interleaved averaging of ADC input channels. ADCA/B channel 0 and 1 are sampled one after another in order to achieve interleaved averaging. SOC0-15 of ADCA and ADCB are configured to sample channel 0 and 1 alternatively with channel 0 being sampled by even SOCs namely SOC0, 2, 4, 6, 8, 10, 12 & 14 and channel 1 being sampled by odd SOCs namely SOC1, 3, 5, 7, 9, 11, 13 & 15.

Sampling is initially triggered through external GPIO signal through enabling ADCEXTSOC signal via input XBAR and thereafter through ADCINT1. GPIO33 is configured to trigger ADCA and B SOCs for the first time in order to ensure synchronous operation. GPIO33 needs to be connected to GPIO32 which in turn is driven by software to trigger the respective ADC SOCs.

ADCA Interrupt ISRs are used to read results of both ADCA and ADCB to demonstrate parallel operation of multiple ADCs. ADCINT2 is configured to be triggered after completion of SOC7 while ADCINT1 is configured to be triggered after completion of SOC15 conversion and respective SOC results of ADCA and ADCB are read in ADCAINT2 and ADCAINT1 ISRs. Read SOC0-SOC15 ADC results are then averaged in ADCINT1 ISR to get the filtered ADC output.

Early interrupt mode is configured to trigger the ADC interrupt just at the end of acquisition window in order to save cycles(~42 cycles) since ADCA result of SOC7 in ADCINT2 and SOC15 in ADCINT1 are read post around 60 and 65 cycles respectively. Also, Fast ISRs are configured in the example to save some cycles during compiler specific context save and restore.

Optimization level: This example is expected to be run with opt level = 2

Sampling rate related calculations:
- ADC acquisition cycles programmed(S + H) = 15 SYSCLKS
- Conversion time for 12-bit data = 10.5 ADCCLKS = N = 42 SYSCLKs
- Time from 1st SOC trigger to interrupt trigger: 15 * 57 + 15 = 870 SYSCLKs
- Next ADC trigger will come after 870 SYSCLKs
- Approximate ISR trigger frequency = 1/(870 * 5ns) = 1/4350ns = 229 KSPS
- Time taken in ADCINT1 ISR : 132 SYSCLKs (Fast ISR)(O2)
- Time taken in ADCINT2 ISR : 76 SYSCLKs (Fast ISR)(O2)

To view results in graph window, add breakpoint inside while(1) loop in main() where bufferFull flag is cleared and plot the watch variables.

External Connections
- A0, A1, B0 and B1 pins should be connected to signals to be converted
- Connect GPIO32 to GPIO33 to trigger ADC channels for the first time

Watch Variables
- *adcA0Results* - A sequence of analog-to-digital conversion samples from pin A0.
- *adcA1Results* - A sequence of analog-to-digital conversion samples from pin A1.
- *adcB0Results* - A sequence of analog-to-digital conversion samples from pin B0.
- *adcB1Results* - A sequence of analog-to-digital conversion samples from pin B1.

20.14.1.17 ADC Open Shorts Detection (adc_open_shorts_detection)

FILE: adc_ex17_open_shorts_detection.c

This example demonstrates the ADC open/shorts detection(ADCOSDETECT) circuit configuration for detecting pin faults in the system. The example enables the open/shorts detection circuit along with mandatory ADC configurations and diagnoses ADCA A0 input pin state before starting normal ADC conversions.

To enable the ADC OSDetect circuit:

1. Configure the ADC for conversion (E.g. channel, SOC, ACQPS, prescalar, trigger etc). The OSDetect functionality is available in 12-bit only, hence ADCA is configured in 12-bit mode in the example.
2. Set up the ADCOSDETECT register for the desired voltage divider connection. Refer device TRM for details on available OSDetect configurations.
3. Initiate a conversion and inspect the conversion result. Note: The results must be interpreted based on what is driving on the input side and what are the values of Rs and Cp. If the Vs signal can be disconnected from the input pin, the circuit can be used to detect open and shorted input pins. In the example, after configuring ADCA A0 channel in 12-bit mode along with other required ADC configurations, following algorithm is used to check the A0 pin status: Step 1: Configure full scale OSDETECT mode & capture ADC results(resultHi) Step 2: Configure zero scale OSDETECT mode & capture ADC results(resultLo) Step 3: Disable OSDETECT mode and capture ADC results(resultNormal) Step 4: Determine the state of the ADC pin a. If the pin is open, resultLo would be equal to Vreflo and resultHi would be equal to Vrefhi b. If the pin is shorted to Vrefhi, resultLo should be approximately equal to Vrefhi and resultHi should be equal to Vrefhi c. If the pin is shorted to Vreflo, resultLo should be equal to Vreflo and resultHi should be approximately equal to Vreflo d. If the pin is connected to a valid signal, resultLo should be greater than osdLoLimit but less than resultNormal while resultHi

should be less than osdHiLimit but greater than resultNormal Input | Full-Scale output | Zero-scale Output | Pin Status

Unknown| VREFHI | VREFLO | Open VREFHI | VREFHI | approx. VREFHI | Shorted to VREFHI VREFLO | approx. VREFLO | VREFLO | Shorted to VREFLO

Vn | Vn < resultHi < VREFHI | VREFLO < resultLo < Vn | Good

Step 5: osDetectStatusVal of value greater than 4 would mean that there is no pin fault. a. If osDetectStatusVal == 1, means pin A0 is OPEN b. If osDetectStatusVal == 2, means pin A0 is shorted to VREFLO c. If osDetectStatusVal == 4, means pin A0 is shorted to VREFHI d. If osDetectStatusVal == 8, means pin A0 is in GOOD/VALID state e. Any value of osDetectStatusVal > 4, means pin A0 is in VALID state

Following points should be noted while configuring the ADC in OSDETECT mode.

1. The divider resistance tolerances can vary widely, hence this feature should not be used to check for conversion accuracy.
2. Consult the device data manual for implementation and availability of analog input channels.
3. Due to high drive impedance, a S+H duration much longer than the ADC minimum will be needed.

External Connections

- A0 pin should be connected to signals to convert

Watch Variables

- *osDetectStatusVal* : OS detection status of voltage on pin A0
- *adcAResult0* : a digital representation of the voltage on pin A0

20.15 ADC Registers

This section describes the Analog-to-Digital Converter Registers.

20.15.1 ADC Base Address Table (C28)

Table 20-13. ADC Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
AdcaRegs	ADC_REGS	ADCA_BASE	0x0000_7400	YES	YES	-	YES	YES
AdcbRegs	ADC_REGS	ADCB_BASE	0x0000_7480	YES	YES	-	YES	YES
AdccRegs	ADC_REGS	ADCC_BASE	0x0000_7500	YES	YES	-	YES	YES
AdcdRegs	ADC_REGS	ADCD_BASE	0x0000_7580	YES	YES	-	YES	YES
AdcaResultRegs	ADC_RESULT_REGS	ADCARESULT_BASE	0x0000_0B00	YES	YES	YES	YES	-
AdcbResultRegs	ADC_RESULT_REGS	ADCBRESULT_BASE	0x0000_0B20	YES	YES	YES	YES	-
AdccResultRegs	ADC_RESULT_REGS	ADCCRESULT_BASE	0x0000_0B40	YES	YES	YES	YES	-
AdcdResultRegs	ADC_RESULT_REGS	ADCDRESULT_BASE	0x0000_0B60	YES	YES	YES	YES	-

20.15.2 ADC_REGS Registers

Table 20-14 lists the memory-mapped registers for the ADC_REGS registers. All register offset addresses not listed in Table 20-14 should be considered as reserved locations and the register contents should not be modified.

Table 20-14. ADC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	ADCCTL1	ADC Control 1 Register	EALLOW	Go
1h	ADCCTL2	ADC Control 2 Register	EALLOW	Go
2h	ADCBURSTCTL	ADC Burst Control Register	EALLOW	Go
3h	ADCINTFLG	ADC Interrupt Flag Register		Go
4h	ADCINTFLGCLR	ADC Interrupt Flag Clear Register		Go
5h	ADCINTOVF	ADC Interrupt Overflow Register		Go
6h	ADCINTOVFCLR	ADC Interrupt Overflow Clear Register		Go
7h	ADCINTSEL1N2	ADC Interrupt 1 and 2 Selection Register	EALLOW	Go
8h	ADCINTSEL3N4	ADC Interrupt 3 and 4 Selection Register	EALLOW	Go
9h	ADCSOCPRICTL	ADC SOC Priority Control Register	EALLOW	Go
Ah	ADCINTSOCSEL1	ADC Interrupt SOC Selection 1 Register	EALLOW	Go
Bh	ADCINTSOCSEL2	ADC Interrupt SOC Selection 2 Register	EALLOW	Go
Ch	ADCSOCFLG1	ADC SOC Flag 1 Register		Go
Dh	ADCSOCFRC1	ADC SOC Force 1 Register		Go
Eh	ADCSOCOVF1	ADC SOC Overflow 1 Register		Go
Fh	ADCSOCOVFCLR1	ADC SOC Overflow Clear 1 Register		Go
10h	ADCSOC0CTL	ADC SOC0 Control Register	EALLOW	Go
12h	ADCSOC1CTL	ADC SOC1 Control Register	EALLOW	Go
14h	ADCSOC2CTL	ADC SOC2 Control Register	EALLOW	Go
16h	ADCSOC3CTL	ADC SOC3 Control Register	EALLOW	Go
18h	ADCSOC4CTL	ADC SOC4 Control Register	EALLOW	Go
1Ah	ADCSOC5CTL	ADC SOC5 Control Register	EALLOW	Go
1Ch	ADCSOC6CTL	ADC SOC6 Control Register	EALLOW	Go
1Eh	ADCSOC7CTL	ADC SOC7 Control Register	EALLOW	Go
20h	ADCSOC8CTL	ADC SOC8 Control Register	EALLOW	Go
22h	ADCSOC9CTL	ADC SOC9 Control Register	EALLOW	Go
24h	ADCSOC10CTL	ADC SOC10 Control Register	EALLOW	Go
26h	ADCSOC11CTL	ADC SOC11 Control Register	EALLOW	Go
28h	ADCSOC12CTL	ADC SOC12 Control Register	EALLOW	Go
2Ah	ADCSOC13CTL	ADC SOC13 Control Register	EALLOW	Go
2Ch	ADCSOC14CTL	ADC SOC14 Control Register	EALLOW	Go
2Eh	ADCSOC15CTL	ADC SOC15 Control Register	EALLOW	Go
30h	ADCEVTSTAT	ADC Event Status Register		Go
32h	ADCEVTCLR	ADC Event Clear Register		Go
34h	ADCEVTSEL	ADC Event Selection Register	EALLOW	Go
36h	ADCEVTINTSEL	ADC Event Interrupt Selection Register	EALLOW	Go
38h	ADCOSDETECT	ADC Open and Shorts Detect Register	EALLOW	Go
39h	ADCCOUNTER	ADC Counter Register		Go
3Ah	ADCREV	ADC Revision Register		Go
3Bh	ADCOFFTRIM	ADC Offset Trim Register	EALLOW	Go
40h	ADCPPB1CONFIG	ADC PPB1 Config Register	EALLOW	Go

Table 20-14. ADC_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
41h	ADCPPB1STAMP	ADC PPB1 Sample Delay Time Stamp Register		Go
42h	ADCPPB1OFFCAL	ADC PPB1 Offset Calibration Register	EALLOW	Go
43h	ADCPPB1OFFREF	ADC PPB1 Offset Reference Register		Go
44h	ADCPPB1TRIPHI	ADC PPB1 Trip High Register	EALLOW	Go
46h	ADCPPB1TRIPLO	ADC PPB1 Trip Low/Trigger Time Stamp Register	EALLOW	Go
48h	ADCPPB2CONFIG	ADC PPB2 Config Register	EALLOW	Go
49h	ADCPPB2STAMP	ADC PPB2 Sample Delay Time Stamp Register		Go
4Ah	ADCPPB2OFFCAL	ADC PPB2 Offset Calibration Register	EALLOW	Go
4Bh	ADCPPB2OFFREF	ADC PPB2 Offset Reference Register		Go
4Ch	ADCPPB2TRIPHI	ADC PPB2 Trip High Register	EALLOW	Go
4Eh	ADCPPB2TRIPLO	ADC PPB2 Trip Low/Trigger Time Stamp Register	EALLOW	Go
50h	ADCPPB3CONFIG	ADC PPB3 Config Register	EALLOW	Go
51h	ADCPPB3STAMP	ADC PPB3 Sample Delay Time Stamp Register		Go
52h	ADCPPB3OFFCAL	ADC PPB3 Offset Calibration Register	EALLOW	Go
53h	ADCPPB3OFFREF	ADC PPB3 Offset Reference Register		Go
54h	ADCPPB3TRIPHI	ADC PPB3 Trip High Register	EALLOW	Go
56h	ADCPPB3TRIPLO	ADC PPB3 Trip Low/Trigger Time Stamp Register	EALLOW	Go
58h	ADCPPB4CONFIG	ADC PPB4 Config Register	EALLOW	Go
59h	ADCPPB4STAMP	ADC PPB4 Sample Delay Time Stamp Register		Go
5Ah	ADCPPB4OFFCAL	ADC PPB4 Offset Calibration Register	EALLOW	Go
5Bh	ADCPPB4OFFREF	ADC PPB4 Offset Reference Register		Go
5Ch	ADCPPB4TRIPHI	ADC PPB4 Trip High Register	EALLOW	Go
5Eh	ADCPPB4TRIPLO	ADC PPB4 Trip Low/Trigger Time Stamp Register	EALLOW	Go
6Fh	ADCINTCYCLE	ADC Early Interrupt Generation Cycle	EALLOW	Go
70h	ADCINLTRIM1	ADC Linearity Trim 1 Register	EALLOW	Go
72h	ADCINLTRIM2	ADC Linearity Trim 2 Register	EALLOW	Go
74h	ADCINLTRIM3	ADC Linearity Trim 3 Register	EALLOW	Go
76h	ADCINLTRIM4	ADC Linearity Trim 4 Register	EALLOW	Go
78h	ADCINLTRIM5	ADC Linearity Trim 5 Register	EALLOW	Go
7Ah	ADCINLTRIM6	ADC Linearity Trim 6 Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 20-15 shows the codes that are used for access types in this section.

Table 20-15. ADC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
Reset or Default Value		

Table 20-15. ADC_REGS Access Type Codes (continued)

Access Type	Code	Description
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

20.15.2.1 ADCCTL1 Register (Offset = 0h) [Reset = 0h]

ADCCTL1 is shown in Figure 20-26 and described in Table 20-16.

Return to the Summary Table.

ADC Control 1 Register

Figure 20-26. ADCCTL1 Register

15	14	13	12	11	10	9	8
RESERVED		ADCBSY	RESERVED	ADCBSYCHN			
R-0h		R-0h	R-0h	R-0h			
7	6	5	4	3	2	1	0
ADCPWDNZ	RESERVED				INTPULSEPOS	RESERVED	
R/W-0h	R-0h				R/W-0h	R-0h	

Table 20-16. ADCCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	RESERVED	R	0h	Reserved
13	ADCBSY	R	0h	ADC Busy. Set when ADC SOC is generated, cleared by hardware four ADC clocks after negative edge of S/H pulse. Used by the ADC state machine to determine if ADC is available to sample. 0 ADC is available to sample next channel 1 ADC is busy and cannot sample another channel Reset type: SYSRSn
12	RESERVED	R	0h	Reserved
11-8	ADCBSYCHN	R	0h	ADC Busy Channel. Set when an ADC Start of Conversion (SOC) is generated. When ADCBSY=0: holds the value of the last converted SOC When ADCBSY=1: reflects the SOC currently being processed 0h SOC0 is currently processing or was last SOC converted 1h SOC1 is currently processing or was last SOC converted 2h SOC2 is currently processing or was last SOC converted 3h SOC3 is currently processing or was last SOC converted 4h SOC4 is currently processing or was last SOC converted 5h SOC5 is currently processing or was last SOC converted 6h SOC6 is currently processing or was last SOC converted 7h SOC7 is currently processing or was last SOC converted 8h SOC8 is currently processing or was last SOC converted 9h SOC9 is currently processing or was last SOC converted Ah SOC10 is currently processing or was last SOC converted Bh SOC11 is currently processing or was last SOC converted Ch SOC12 is currently processing or was last SOC converted Dh SOC13 is currently processing or was last SOC converted Eh SOC14 is currently processing or was last SOC converted Fh SOC15 is currently processing or was last SOC converted Reset type: SYSRSn
7	ADCPWDNZ	R/W	0h	ADC Power Down (active low). This bit controls the power up and power down of all the analog circuitry inside the analog core. 0 All analog circuitry inside the core is powered down 1 All analog circuitry inside the core is powered up Reset type: SYSRSn
6-3	RESERVED	R	0h	Reserved
2	INTPULSEPOS	R/W	0h	ADC Interrupt Pulse Position. 0 Interrupt pulse generation occurs when ADC begins conversion (at the end of the acquisition window) plus a number of SYSCLK cycles as specified in the ADCINTCYCLE.OFFSET register. 1 Interrupt pulse generation occurs at the end of the conversion, 1 cycle prior to the ADC result latching into its result register Reset type: SYSRSn

Table 20-16. ADCCTL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	RESERVED	R	0h	Reserved

20.15.2.2 ADCCTL2 Register (Offset = 1h) [Reset = 0h]

ADCCTL2 is shown in Figure 20-27 and described in Table 20-17.

Return to the Summary Table.

ADC Control 2 Register

Figure 20-27. ADCCTL2 Register

15	14	13	12	11	10	9	8
RESERVED				RESERVED			
R-0h				R-0h			
7	6	5	4	3	2	1	0
SIGNALMODE	RESOLUTION	RESERVED		PRESCALE			
R/W-0h	R/W-0h	R-0h		R/W-0h			

Table 20-17. ADCCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-8	RESERVED	R	0h	Reserved
7	SIGNALMODE	R/W	0h	SOC Signaling Mode. Selects the input mode of the converter. Use the AdcSetMode function to change the signal mode. 0 Single-ended 1 Differential Reset type: SYSRSn
6	RESOLUTION	R/W	0h	SOC Conversion Resolution. Selects the resolution of the converter. Use the AdcSetMode function to change the resolution. 0 12-bit resolution 1 16-bit resolution Reset type: SYSRSn
5-4	RESERVED	R	0h	Reserved
3-0	PRESCALE	R/W	0h	ADC Clock Prescaler. 0000 ADCCLK = Input Clock / 1.0 0001 Invalid 0010 ADCCLK = Input Clock / 2.0 0011 ADCCLK = Input Clock / 2.5 0100 ADCCLK = Input Clock / 3.0 0101 ADCCLK = Input Clock / 3.5 0110 ADCCLK = Input Clock / 4.0 0111 ADCCLK = Input Clock / 4.5 1000 ADCCLK = Input Clock / 5.0 1001 ADCCLK = Input Clock / 5.5 1010 ADCCLK = Input Clock / 6.0 1011 ADCCLK = Input Clock / 6.5 1100 ADCCLK = Input Clock / 7.0 1101 ADCCLK = Input Clock / 7.5 1110 ADCCLK = Input Clock / 8.0 1111 ADCCLK = Input Clock / 8.5 Reset type: SYSRSn

20.15.2.3 ADCBURSTCTL Register (Offset = 2h) [Reset = 0h]

ADCBURSTCTL is shown in Figure 20-28 and described in Table 20-18.

Return to the Summary Table.

ADC Burst Control Register

Figure 20-28. ADCBURSTCTL Register

15	14	13	12	11	10	9	8
BURSTEN	RESERVED			BURSTSIZE			
R/W-0h	R-0h			R/W-0h			
7	6	5	4	3	2	1	0
RESERVED		BURSTTRIGSEL					
R-0h		R/W-0h					

Table 20-18. ADCBURSTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	BURSTEN	R/W	0h	SOC Burst Mode Enable. This bit enables the SOC Burst Mode of operation. 0 Burst mode is disabled. 1 Burst mode is enabled. Reset type: SYSRSn
14-12	RESERVED	R	0h	Reserved
11-8	BURSTSIZE	R/W	0h	SOC Burst Size Select. This bit field determines how many SOCs are converted when a burst conversion sequence is started. The first SOC converted is defined by the round robin pointer, which is advanced as each SOC is converted. 0h 1 SOC converted 1h 2 SOCs converted 2h 3 SOCs converted 3h 4 SOCs converted 4h 5 SOCs converted 5h 6 SOCs converted 6h 7 SOCs converted 7h 8 SOCs converted 8h 9 SOCs converted 9h 10 SOCs converted Ah 11 SOCs converted Bh 12 SOCs converted Ch 13 SOCs converted Dh 14 SOCs converted Eh 15 SOCs converted Fh 16 SOCs converted Reset type: SYSRSn
7-6	RESERVED	R	0h	Reserved

Table 20-18. ADCBURSTCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	BURSTTRIGSEL	R/W	0h	SOC Burst Trigger Source Select. Configures which trigger will start a burst conversion sequence. 00h BURSTTRIG0 - Software only 01h BURSTTRIG1 - CPU1 Timer 0, TINT0n 02h BURSTTRIG2 - CPU1 Timer 1, TINT1n 03h BURSTTRIG3 - CPU1 Timer 2, TINT2n 04h BURSTTRIG4 - GPIO, Input X-Bar INPUT5 05h BURSTTRIG5 - ePWM1, ADCSOCA 06h BURSTTRIG6 - ePWM1, ADCSOCB 07h BURSTTRIG7 - ePWM2, ADCSOCA 08h BURSTTRIG8 - ePWM2, ADCSOCB 09h BURSTTRIG9 - ePWM3, ADCSOCA 0Ah BURSTTRIG10 - ePWM3, ADCSOCB 0Bh BURSTTRIG11 - ePWM4, ADCSOCA 0Ch BURSTTRIG12 - ePWM4, ADCSOCB 0Dh BURSTTRIG13 - ePWM5, ADCSOCA 0Eh BURSTTRIG14 - ePWM5, ADCSOCB 0Fh BURSTTRIG15 - ePWM6, ADCSOCA 10h BURSTTRIG16 - ePWM6, ADCSOCB 11h BURSTTRIG17 - ePWM7, ADCSOCA 12h BURSTTRIG18 - ePWM7, ADCSOCB 13h BURSTTRIG19 - ePWM8, ADCSOCA 14h BURSTTRIG20 - ePWM8, ADCSOCB 15h BURSTTRIG21 - ePWM9, ADCSOCA 16h BURSTTRIG22 - ePWM9, ADCSOCB 17h BURSTTRIG23 - ePWM10, ADCSOCA 18h BURSTTRIG24 - ePWM10, ADCSOCB 19h BURSTTRIG25 - ePWM11, ADCSOCA 1Ah BURSTTRIG26 - ePWM11, ADCSOCB 1Bh BURSTTRIG27 - ePWM12, ADCSOCA 1Ch BURSTTRIG28 - ePWM12, ADCSOCB 1Dh BURSTTRIG29 - CPU2 Timer 0, TINT0n 1Eh BURSTTRIG30 - CPU2 Timer 1, TINT1n 1Fh BURSTTRIG31 - CPU2 Timer 2, TINT2n 20h BURSTTRIG32 - ePWM13, ADCSOCA 21h BURSTTRIG33 - ePWM13, ADCSOCB 22h BURSTTRIG34 - ePWM14, ADCSOCA 23h BURSTTRIG35 - ePWM14, ADCSOCB 24h BURSTTRIG36 - ePWM15, ADCSOCA 25h BURSTTRIG37 - ePWM15, ADCSOCB 26h BURSTTRIG38 - ePWM16, ADCSOCA 27h BURSTTRIG39 - ePWM16, ADCSOCB 28h - 3Fh - Reserved Reset type: SYSRSn

20.15.2.4 ADCINTFLG Register (Offset = 3h) [Reset = 0h]

ADCINTFLG is shown in Figure 20-29 and described in Table 20-19.

Return to the Summary Table.

ADC Interrupt Flag Register

Figure 20-29. ADCINTFLG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				ADCINT4	ADCINT3	ADCINT2	ADCINT1
R-0h				R-0h	R-0h	R-0h	R-0h

Table 20-19. ADCINTFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	ADCINT4	R	0h	ADC Interrupt 4 Flag. Reading these flags indicates if the associated ADCINT pulse was generated since the last clear. 0 No ADC interrupt pulse generated 1 ADC interrupt pulse generated If the ADC interrupt is placed in continue to interrupt mode (INTSELxNy register) then further interrupt pulses are generated whenever a selected EOC event occurs even if the flag bit is set. If the continuous mode is not enabled, then no further interrupt pulses are generated until the user clears this flag bit using the ADCINTFLGCLR register. Rather, an ADC interrupt overflow event occurs in the ADCINTOVF register. Reset type: SYSRSn
2	ADCINT3	R	0h	ADC Interrupt 3 Flag. Reading these flags indicates if the associated ADCINT pulse was generated since the last clear. 0 No ADC interrupt pulse generated 1 ADC interrupt pulse generated If the ADC interrupt is placed in continue to interrupt mode (INTSELxNy register) then further interrupt pulses are generated whenever a selected EOC event occurs even if the flag bit is set. If the continuous mode is not enabled, then no further interrupt pulses are generated until the user clears this flag bit using the ADCINTFLGCLR register. Rather, an ADC interrupt overflow event occurs in the ADCINTOVF register. Reset type: SYSRSn
1	ADCINT2	R	0h	ADC Interrupt 2 Flag. Reading these flags indicates if the associated ADCINT pulse was generated since the last clear. 0 No ADC interrupt pulse generated 1 ADC interrupt pulse generated If the ADC interrupt is placed in continue to interrupt mode (INTSELxNy register) then further interrupt pulses are generated whenever a selected EOC event occurs even if the flag bit is set. If the continuous mode is not enabled, then no further interrupt pulses are generated until the user clears this flag bit using the ADCINTFLGCLR register. Rather, an ADC interrupt overflow event occurs in the ADCINTOVF register. Reset type: SYSRSn

Table 20-19. ADCINTFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	ADCINT1	R	0h	ADC Interrupt 1 Flag. Reading these flags indicates if the associated ADCINT pulse was generated since the last clear. 0 No ADC interrupt pulse generated 1 ADC interrupt pulse generated If the ADC interrupt is placed in continue to interrupt mode (INTSELxNy register) then further interrupt pulses are generated whenever a selected EOC event occurs even if the flag bit is set. If the continuous mode is not enabled, then no further interrupt pulses are generated until the user clears this flag bit using the ADCINTFLGCLR register. Rather, an ADC interrupt overflow event occurs in the ADCINTOVF register. Reset type: SYSRSn

20.15.2.5 ADCINTFLGCLR Register (Offset = 4h) [Reset = 0h]

ADCINTFLGCLR is shown in Figure 20-30 and described in Table 20-20.

Return to the Summary Table.

ADC Interrupt Flag Clear Register

Figure 20-30. ADCINTFLGCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				ADCINT4	ADCINT3	ADCINT2	ADCINT1
R-0h				R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h

Table 20-20. ADCINTFLGCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	ADCINT4	R-0/W1C	0h	ADC Interrupt 4 Flag Clear. Reads return 0. 0 No action 1 Clears respective flag bit in the ADCINTFLG register. If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority and the overflow bit will not be set Reset type: SYSRSn
2	ADCINT3	R-0/W1C	0h	ADC Interrupt 3 Flag Clear. Reads return 0. 0 No action 1 Clears respective flag bit in the ADCINTFLG register. If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority and the overflow bit will not be set Reset type: SYSRSn
1	ADCINT2	R-0/W1C	0h	ADC Interrupt 2 Flag Clear. Reads return 0. 0 No action 1 Clears respective flag bit in the ADCINTFLG register. If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority and the overflow bit will not be set Reset type: SYSRSn
0	ADCINT1	R-0/W1C	0h	ADC Interrupt 1 Flag Clear. Reads return 0. 0 No action 1 Clears respective flag bit in the ADCINTFLG register. If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority and the overflow bit will not be set Reset type: SYSRSn

20.15.2.6 ADCINTOVF Register (Offset = 5h) [Reset = 0h]

ADCINTOVF is shown in Figure 20-31 and described in Table 20-21.

Return to the Summary Table.

ADC Interrupt Overflow Register

Figure 20-31. ADCINTOVF Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				ADCINT4	ADCINT3	ADCINT2	ADCINT1
R-0h				R-0h	R-0h	R-0h	R-0h

Table 20-21. ADCINTOVF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	ADCINT4	R	0h	ADC Interrupt 4 Overflow Flags Indicates if an overflow occurred when generating ADCINT pulses. If the respective ADCINTFLG bit is set and a selected additional EOC trigger is generated, then an overflow condition occurs. 0 No ADC interrupt overflow event detected. 1 ADC Interrupt overflow event detected. The overflow bit does not care about the continuous mode bit state. An overflow condition is generated irrespective of this mode selection. Reset type: SYSRSn
2	ADCINT3	R	0h	ADC Interrupt 3 Overflow Flags Indicates if an overflow occurred when generating ADCINT pulses. If the respective ADCINTFLG bit is set and a selected additional EOC trigger is generated, then an overflow condition occurs. 0 No ADC interrupt overflow event detected. 1 ADC Interrupt overflow event detected. The overflow bit does not care about the continuous mode bit state. An overflow condition is generated irrespective of this mode selection. Reset type: SYSRSn
1	ADCINT2	R	0h	ADC Interrupt 2 Overflow Flags Indicates if an overflow occurred when generating ADCINT pulses. If the respective ADCINTFLG bit is set and a selected additional EOC trigger is generated, then an overflow condition occurs. 0 No ADC interrupt overflow event detected. 1 ADC Interrupt overflow event detected. The overflow bit does not care about the continuous mode bit state. An overflow condition is generated irrespective of this mode selection. Reset type: SYSRSn
0	ADCINT1	R	0h	ADC Interrupt 1 Overflow Flags Indicates if an overflow occurred when generating ADCINT pulses. If the respective ADCINTFLG bit is set and a selected additional EOC trigger is generated, then an overflow condition occurs. 0 No ADC interrupt overflow event detected. 1 ADC Interrupt overflow event detected. The overflow bit does not care about the continuous mode bit state. An overflow condition is generated irrespective of this mode selection. Reset type: SYSRSn

20.15.2.7 ADCINTOVFCLR Register (Offset = 6h) [Reset = 0h]

ADCINTOVFCLR is shown in Figure 20-32 and described in Table 20-22.

Return to the Summary Table.

ADC Interrupt Overflow Clear Register

Figure 20-32. ADCINTOVFCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				ADCINT4	ADCINT3	ADCINT2	ADCINT1
R-0h				R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h

Table 20-22. ADCINTOVFCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	ADCINT4	R-0/W1C	0h	ADC Interrupt 4 Overflow Clear Bits 0 No action. 1 Clears the respective overflow bit in the ADCINTOVF register. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCINTOVF register, then hardware has priority and the ADCINTOVF bit will be set. Reset type: SYSRSn
2	ADCINT3	R-0/W1C	0h	ADC Interrupt 3 Overflow Clear Bits 0 No action. 1 Clears the respective overflow bit in the ADCINTOVF register. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCINTOVF register, then hardware has priority and the ADCINTOVF bit will be set. Reset type: SYSRSn
1	ADCINT2	R-0/W1C	0h	ADC Interrupt 2 Overflow Clear Bits 0 No action. 1 Clears the respective overflow bit in the ADCINTOVF register. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCINTOVF register, then hardware has priority and the ADCINTOVF bit will be set. Reset type: SYSRSn
0	ADCINT1	R-0/W1C	0h	ADC Interrupt 1 Overflow Clear Bits 0 No action. 1 Clears the respective overflow bit in the ADCINTOVF register. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCINTOVF register, then hardware has priority and the ADCINTOVF bit will be set. Reset type: SYSRSn

20.15.2.8 ADCINTSEL1N2 Register (Offset = 7h) [Reset = 0h]

ADCINTSEL1N2 is shown in Figure 20-33 and described in Table 20-23.

Return to the Summary Table.

ADC Interrupt 1 and 2 Selection Register

Figure 20-33. ADCINTSEL1N2 Register

15	14	13	12	11	10	9	8
RESERVED	INT2CONT	INT2E	RESERVED	INT2SEL			
R-0h	R/W-0h	R/W-0h	R-0h	R/W-0h			
7	6	5	4	3	2	1	0
RESERVED	INT1CONT	INT1E	RESERVED	INT1SEL			
R-0h	R/W-0h	R/W-0h	R-0h	R/W-0h			

Table 20-23. ADCINTSEL1N2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	INT2CONT	R/W	0h	ADCINT2 Continue to Interrupt Mode 0 No further ADCINT2 pulses are generated until ADCINT2 flag (in ADCINTFLG register) is cleared by user. 1 ADCINT2 pulses are generated whenever an EOC pulse is generated irrespective of whether the flag bit is cleared or not. Reset type: SYSRSn
13	INT2E	R/W	0h	ADCINT2 Interrupt Enable 0 ADCINT2 is disabled 1 ADCINT2 is enabled Reset type: SYSRSn
12	RESERVED	R	0h	Reserved
11-8	INT2SEL	R/W	0h	ADCINT2 EOC Source Select 0h EOC0 is trigger for ADCINT2 1h EOC1 is trigger for ADCINT2 2h EOC2 is trigger for ADCINT2 3h EOC3 is trigger for ADCINT2 4h EOC4 is trigger for ADCINT2 5h EOC5 is trigger for ADCINT2 6h EOC6 is trigger for ADCINT2 7h EOC7 is trigger for ADCINT2 8h EOC8 is trigger for ADCINT2 9h EOC9 is trigger for ADCINT2 Ah EOC10 is trigger for ADCINT2 Bh EOC11 is trigger for ADCINT2 Ch EOC12 is trigger for ADCINT2 Dh EOC13 is trigger for ADCINT2 Eh EOC14 is trigger for ADCINT2 Fh EOC15 is trigger for ADCINT2 Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6	INT1CONT	R/W	0h	ADCINT1 Continue to Interrupt Mode 0 No further ADCINT1 pulses are generated until ADCINT1 flag (in ADCINTFLG register) is cleared by user. 1 ADCINT1 pulses are generated whenever an EOC pulse is generated irrespective of whether the flag bit is cleared or not. Reset type: SYSRSn
5	INT1E	R/W	0h	ADCINT1 Interrupt Enable 0 ADCINT1 is disabled 1 ADCINT1 is enabled Reset type: SYSRSn
4	RESERVED	R	0h	Reserved

Table 20-23. ADCINTSEL1N2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	INT1SEL	R/W	0h	ADCINT1 EOC Source Select 0h EOC0 is trigger for ADCINT1 1h EOC1 is trigger for ADCINT1 2h EOC2 is trigger for ADCINT1 3h EOC3 is trigger for ADCINT1 4h EOC4 is trigger for ADCINT1 5h EOC5 is trigger for ADCINT1 6h EOC6 is trigger for ADCINT1 7h EOC7 is trigger for ADCINT1 8h EOC8 is trigger for ADCINT1 9h EOC9 is trigger for ADCINT1 Ah EOC10 is trigger for ADCINT1 Bh EOC11 is trigger for ADCINT1 Ch EOC12 is trigger for ADCINT1 Dh EOC13 is trigger for ADCINT1 Eh EOC14 is trigger for ADCINT1 Fh EOC15 is trigger for ADCINT1 Reset type: SYSRSn

20.15.2.9 ADCINTSEL3N4 Register (Offset = 8h) [Reset = 0h]

ADCINTSEL3N4 is shown in Figure 20-34 and described in Table 20-24.

Return to the Summary Table.

ADC Interrupt 3 and 4 Selection Register

Figure 20-34. ADCINTSEL3N4 Register

15	14	13	12	11	10	9	8
RESERVED	INT4CONT	INT4E	RESERVED	INT4SEL			
R-0h	R/W-0h	R/W-0h	R-0h	R/W-0h			
7	6	5	4	3	2	1	0
RESERVED	INT3CONT	INT3E	RESERVED	INT3SEL			
R-0h	R/W-0h	R/W-0h	R-0h	R/W-0h			

Table 20-24. ADCINTSEL3N4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	INT4CONT	R/W	0h	ADCINT4 Continue to Interrupt Mode 0 No further ADCINT4 pulses are generated until ADCINT4 flag (in ADCINTFLG register) is cleared by user. 1 ADCINT4 pulses are generated whenever an EOC pulse is generated irrespective of whether the flag bit is cleared or not. Reset type: SYSRSn
13	INT4E	R/W	0h	ADCINT4 Interrupt Enable 0 ADCINT4 is disabled 1 ADCINT4 is enabled Reset type: SYSRSn
12	RESERVED	R	0h	Reserved
11-8	INT4SEL	R/W	0h	ADCINT4 EOC Source Select 0h EOC0 is trigger for ADCINT4 1h EOC1 is trigger for ADCINT4 2h EOC2 is trigger for ADCINT4 3h EOC3 is trigger for ADCINT4 4h EOC4 is trigger for ADCINT4 5h EOC5 is trigger for ADCINT4 6h EOC6 is trigger for ADCINT4 7h EOC7 is trigger for ADCINT4 8h EOC8 is trigger for ADCINT4 9h EOC9 is trigger for ADCINT4 Ah EOC10 is trigger for ADCINT4 Bh EOC11 is trigger for ADCINT4 Ch EOC12 is trigger for ADCINT4 Dh EOC13 is trigger for ADCINT4 Eh EOC14 is trigger for ADCINT4 Fh EOC15 is trigger for ADCINT4 Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6	INT3CONT	R/W	0h	ADCINT3 Continue to Interrupt Mode 0 No further ADCINT3 pulses are generated until ADCINT3 flag (in ADCINTFLG register) is cleared by user. 1 ADCINT3 pulses are generated whenever an EOC pulse is generated irrespective of whether the flag bit is cleared or not. Reset type: SYSRSn
5	INT3E	R/W	0h	ADCINT3 Interrupt Enable 0 ADCINT3 is disabled 1 ADCINT3 is enabled Reset type: SYSRSn
4	RESERVED	R	0h	Reserved

Table 20-24. ADCINTSEL3N4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	INT3SEL	R/W	0h	ADCINT3 EOC Source Select 0h EOC0 is trigger for ADCINT3 1h EOC1 is trigger for ADCINT3 2h EOC2 is trigger for ADCINT3 3h EOC3 is trigger for ADCINT3 4h EOC4 is trigger for ADCINT3 5h EOC5 is trigger for ADCINT3 6h EOC6 is trigger for ADCINT3 7h EOC7 is trigger for ADCINT3 8h EOC8 is trigger for ADCINT3 9h EOC9 is trigger for ADCINT3 Ah EOC10 is trigger for ADCINT3 Bh EOC11 is trigger for ADCINT3 Ch EOC12 is trigger for ADCINT3 Dh EOC13 is trigger for ADCINT3 Eh EOC14 is trigger for ADCINT3 Fh EOC15 is trigger for ADCINT3 Reset type: SYSRSn

20.15.2.10 ADCSOCPRICTL Register (Offset = 9h) [Reset = 200h]

ADCSOCPRICTL is shown in Figure 20-35 and described in Table 20-25.

Return to the Summary Table.

ADC SOC Priority Control Register

Figure 20-35. ADCSOCPRICTL Register

15	14	13	12	11	10	9	8
RESERVED						RRPOINTER	
R-0h						R-10h	
7	6	5	4	3	2	1	0
RRPOINTER			SOCPRIORITY				
R-10h			R/W-0h				

Table 20-25. ADCSOCPRICTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-5	RRPOINTER	R	10h	Round Robin Pointer. Holds the value of the last converted round robin SOCx to be used by the round robin scheme to determine order of conversions. 00h SOC0 was last round robin SOC to convert, SOC1 is highest round robin priority. 01h SOC1 was last round robin SOC to convert, SOC2 is highest round robin priority. 02h SOC2 was last round robin SOC to convert, SOC3 is highest round robin priority. 03h SOC3 was last round robin SOC to convert, SOC4 is highest round robin priority. 04h SOC4 was last round robin SOC to convert, SOC5 is highest round robin priority. 05h SOC5 was last round robin SOC to convert, SOC6 is highest round robin priority. 06h SOC6 was last round robin SOC to convert, SOC7 is highest round robin priority. 07h SOC7 was last round robin SOC to convert, SOC8 is highest round robin priority. 08h SOC8 was last round robin SOC to convert, SOC9 is highest round robin priority. 09h SOC9 was last round robin SOC to convert, SOC10 is highest round robin priority. 0Ah SOC10 was last round robin SOC to convert, SOC11 is highest round robin priority. 0Bh SOC11 was last round robin SOC to convert, SOC12 is highest round robin priority. 0Ch SOC12 was last round robin SOC to convert, SOC13 is highest round robin priority. 0Dh SOC13 was last round robin SOC to convert, SOC14 is highest round robin priority. 0Eh SOC14 was last round robin SOC to convert, SOC15 is highest round robin priority. 0Fh SOC15 was last round robin SOC to convert, SOC0 is highest round robin priority. 10h Reset value to indicate no SOC has been converted. SOC0 is highest round robin priority. Set to this value when the ADC module is reset by SOFTPRES or when the ADCSOCPRICTL register is written. In the latter case, if a conversion is currently in progress, it will complete and then the new priority will take effect. Others Invalid value. Reset type: SYSRSn

Table 20-25. ADCSOCPRICTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	SOCPRIORITY	R/W	0h	SOC Priority Determines the cutoff point for priority mode and round robin arbitration for SOCx 00h SOC priority is handled in round robin mode for all channels. 01h SOC0 is high priority, rest of channels are in round robin mode. 02h SOC0-SOC1 are high priority, SOC2-SOC15 are in round robin mode. 03h SOC0-SOC2 are high priority, SOC3-SOC15 are in round robin mode. 04h SOC0-SOC3 are high priority, SOC4-SOC15 are in round robin mode. 05h SOC0-SOC4 are high priority, SOC5-SOC15 are in round robin mode. 06h SOC0-SOC5 are high priority, SOC6-SOC15 are in round robin mode. 07h SOC0-SOC6 are high priority, SOC7-SOC15 are in round robin mode. 08h SOC0-SOC7 are high priority, SOC8-SOC15 are in round robin mode. 09h SOC0-SOC8 are high priority, SOC9-SOC15 are in round robin mode. 0Ah SOC0-SOC9 are high priority, SOC10-SOC15 are in round robin mode. 0Bh SOC0-SOC10 are high priority, SOC11-SOC15 are in round robin mode. 0Ch SOC0-SOC11 are high priority, SOC12-SOC15 are in round robin mode. 0Dh SOC0-SOC12 are high priority, SOC13-SOC15 are in round robin mode. 0Eh SOC0-SOC13 are high priority, SOC14-SOC15 are in round robin mode. 0Fh SOC0-SOC14 are high priority, SOC15 is in round robin mode. 10h All SOCs are in high priority mode, arbitrated by SOC number. Others Invalid selection. Reset type: SYSRSn

20.15.2.11 ADCINTSOCSEL1 Register (Offset = Ah) [Reset = 0h]

ADCINTSOCSEL1 is shown in Figure 20-36 and described in Table 20-26.

Return to the Summary Table.

ADC Interrupt SOC Selection 1 Register

Figure 20-36. ADCINTSOCSEL1 Register

15	14	13	12	11	10	9	8
SOC7		SOC6		SOC5		SOC4	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
SOC3		SOC2		SOC1		SOC0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 20-26. ADCINTSOCSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	SOC7	R/W	0h	SOC7 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC7. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC7. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC7. 10 ADCINT2 will trigger SOC7. 11 Invalid selection. Reset type: SYSRSn
13-12	SOC6	R/W	0h	SOC6 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC6. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC6. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC6. 10 ADCINT2 will trigger SOC6. 11 Invalid selection. Reset type: SYSRSn
11-10	SOC5	R/W	0h	SOC5 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC5. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC5. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC5. 10 ADCINT2 will trigger SOC5. 11 Invalid selection. Reset type: SYSRSn
9-8	SOC4	R/W	0h	SOC4 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC4. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC4. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC4. 10 ADCINT2 will trigger SOC4. 11 Invalid selection. Reset type: SYSRSn

Table 20-26. ADCINTSOCSEL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	SOC3	R/W	0h	SOC3 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC3. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC3. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC3. 10 ADCINT2 will trigger SOC3. 11 Invalid selection. Reset type: SYSRSn
5-4	SOC2	R/W	0h	SOC2 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC2. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC2. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC2. 10 ADCINT2 will trigger SOC2. 11 Invalid selection. Reset type: SYSRSn
3-2	SOC1	R/W	0h	SOC1 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC1. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC1. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC1. 10 ADCINT2 will trigger SOC1. 11 Invalid selection. Reset type: SYSRSn
1-0	SOC0	R/W	0h	SOC0 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC0. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC0. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC0. 10 ADCINT2 will trigger SOC0. 11 Invalid selection. Reset type: SYSRSn

20.15.2.12 ADCINTSOCSEL2 Register (Offset = Bh) [Reset = 0h]

ADCINTSOCSEL2 is shown in Figure 20-37 and described in Table 20-27.

Return to the Summary Table.

ADC Interrupt SOC Selection 2 Register

Figure 20-37. ADCINTSOCSEL2 Register

15	14	13	12	11	10	9	8
SOC15		SOC14		SOC13		SOC12	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
SOC11		SOC10		SOC9		SOC8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 20-27. ADCINTSOCSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	SOC15	R/W	0h	SOC15 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC15. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC15. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC15. 10 ADCINT2 will trigger SOC15. 11 Invalid selection. Reset type: SYSRSn
13-12	SOC14	R/W	0h	SOC14 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC14. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC14. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC14. 10 ADCINT2 will trigger SOC14. 11 Invalid selection. Reset type: SYSRSn
11-10	SOC13	R/W	0h	SOC13 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC13. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC13. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC13. 10 ADCINT2 will trigger SOC13. 11 Invalid selection. Reset type: SYSRSn
9-8	SOC12	R/W	0h	SOC12 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC12. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC12. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC12. 10 ADCINT2 will trigger SOC12. 11 Invalid selection. Reset type: SYSRSn

Table 20-27. ADCINTSOCSEL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	SOC11	R/W	0h	SOC11 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC11. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC11. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC11. 10 ADCINT2 will trigger SOC11. 11 Invalid selection. Reset type: SYSRSn
5-4	SOC10	R/W	0h	SOC10 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC10. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC10. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC10. 10 ADCINT2 will trigger SOC10. 11 Invalid selection. Reset type: SYSRSn
3-2	SOC9	R/W	0h	SOC9 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC9. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC9. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC9. 10 ADCINT2 will trigger SOC9. 11 Invalid selection. Reset type: SYSRSn
1-0	SOC8	R/W	0h	SOC8 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC8. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC8. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC8. 10 ADCINT2 will trigger SOC8. 11 Invalid selection. Reset type: SYSRSn

20.15.2.13 ADCSOCFLG1 Register (Offset = Ch) [Reset = 0h]

ADCSOCFLG1 is shown in Figure 20-38 and described in Table 20-28.

Return to the Summary Table.

ADC SOC Flag 1 Register

Figure 20-38. ADCSOCFLG1 Register

15	14	13	12	11	10	9	8
SOC15	SOC14	SOC13	SOC12	SOC11	SOC10	SOC9	SOC8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
SOC7	SOC6	SOC5	SOC4	SOC3	SOC2	SOC1	SOC0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 20-28. ADCSOCFLG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SOC15	R	0h	SOC15 Start of Conversion Flag. Indicates the state of SOC15 conversions. 0 No sample pending for SOC15. 1 Trigger has been received and sample is pending for SOC15. This bit will be automatically cleared when the SOC15 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
14	SOC14	R	0h	SOC14 Start of Conversion Flag. Indicates the state of SOC14 conversions. 0 No sample pending for SOC14. 1 Trigger has been received and sample is pending for SOC14. This bit will be automatically cleared when the SOC14 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
13	SOC13	R	0h	SOC13 Start of Conversion Flag. Indicates the state of SOC13 conversions. 0 No sample pending for SOC13. 1 Trigger has been received and sample is pending for SOC13. This bit will be automatically cleared when the SOC13 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn

Table 20-28. ADCSOCFLG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	SOC12	R	0h	SOC12 Start of Conversion Flag. Indicates the state of SOC12 conversions. 0 No sample pending for SOC12. 1 Trigger has been received and sample is pending for SOC12. This bit will be automatically cleared when the SOC12 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
11	SOC11	R	0h	SOC11 Start of Conversion Flag. Indicates the state of SOC11 conversions. 0 No sample pending for SOC11. 1 Trigger has been received and sample is pending for SOC11. This bit will be automatically cleared when the SOC11 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
10	SOC10	R	0h	SOC10 Start of Conversion Flag. Indicates the state of SOC10 conversions. 0 No sample pending for SOC10. 1 Trigger has been received and sample is pending for SOC10. This bit will be automatically cleared when the SOC10 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
9	SOC9	R	0h	SOC9 Start of Conversion Flag. Indicates the state of SOC9 conversions. 0 No sample pending for SOC9. 1 Trigger has been received and sample is pending for SOC9. This bit will be automatically cleared when the SOC9 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
8	SOC8	R	0h	SOC8 Start of Conversion Flag. Indicates the state of SOC8 conversions. 0 No sample pending for SOC8. 1 Trigger has been received and sample is pending for SOC8. This bit will be automatically cleared when the SOC8 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn

Table 20-28. ADCSOCFLG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	SOC7	R	0h	SOC7 Start of Conversion Flag. Indicates the state of SOC7 conversions. 0 No sample pending for SOC7. 1 Trigger has been received and sample is pending for SOC7. This bit will be automatically cleared when the SOC7 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
6	SOC6	R	0h	SOC6 Start of Conversion Flag. Indicates the state of SOC6 conversions. 0 No sample pending for SOC6. 1 Trigger has been received and sample is pending for SOC6. This bit will be automatically cleared when the SOC6 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
5	SOC5	R	0h	SOC5 Start of Conversion Flag. Indicates the state of SOC5 conversions. 0 No sample pending for SOC5. 1 Trigger has been received and sample is pending for SOC5. This bit will be automatically cleared when the SOC5 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
4	SOC4	R	0h	SOC4 Start of Conversion Flag. Indicates the state of SOC4 conversions. 0 No sample pending for SOC4. 1 Trigger has been received and sample is pending for SOC4. This bit will be automatically cleared when the SOC4 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
3	SOC3	R	0h	SOC3 Start of Conversion Flag. Indicates the state of SOC3 conversions. 0 No sample pending for SOC3. 1 Trigger has been received and sample is pending for SOC3. This bit will be automatically cleared when the SOC3 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn

Table 20-28. ADCSOCFLG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	SOC2	R	0h	SOC2 Start of Conversion Flag. Indicates the state of SOC2 conversions. 0 No sample pending for SOC2. 1 Trigger has been received and sample is pending for SOC2. This bit will be automatically cleared when the SOC2 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
1	SOC1	R	0h	SOC1 Start of Conversion Flag. Indicates the state of SOC1 conversions. 0 No sample pending for SOC1. 1 Trigger has been received and sample is pending for SOC1. This bit will be automatically cleared when the SOC1 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
0	SOC0	R	0h	SOC0 Start of Conversion Flag. Indicates the state of SOC0 conversions. 0 No sample pending for SOC0. 1 Trigger has been received and sample is pending for SOC0. This bit will be automatically cleared when the SOC0 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn

20.15.2.14 ADCSOCFRC1 Register (Offset = Dh) [Reset = 0h]

ADCSOCFRC1 is shown in Figure 20-39 and described in Table 20-29.

Return to the Summary Table.

ADC SOC Force 1 Register

Figure 20-39. ADCSOCFRC1 Register

15	14	13	12	11	10	9	8
SOC15	SOC14	SOC13	SOC12	SOC11	SOC10	SOC9	SOC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
SOC7	SOC6	SOC5	SOC4	SOC3	SOC2	SOC1	SOC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 20-29. ADCSOCFRC1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SOC15	R-0/W1S	0h	SOC15 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC15 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC15 flag bit to 1. This will cause a conversion to start once priority is given to SOC15. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC15 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
14	SOC14	R-0/W1S	0h	SOC14 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC14 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC14 flag bit to 1. This will cause a conversion to start once priority is given to SOC14. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC14 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
13	SOC13	R-0/W1S	0h	SOC13 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC13 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC13 flag bit to 1. This will cause a conversion to start once priority is given to SOC13. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC13 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn

Table 20-29. ADCSOCFRC1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	SOC12	R-0/W1S	0h	SOC12 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC12 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC12 flag bit to 1. This will cause a conversion to start once priority is given to SOC12. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC12 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
11	SOC11	R-0/W1S	0h	SOC11 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC11 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC11 flag bit to 1. This will cause a conversion to start once priority is given to SOC11. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC11 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
10	SOC10	R-0/W1S	0h	SOC10 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC10 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC10 flag bit to 1. This will cause a conversion to start once priority is given to SOC10. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC10 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
9	SOC9	R-0/W1S	0h	SOC9 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC9 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC9 flag bit to 1. This will cause a conversion to start once priority is given to SOC9. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC9 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn

Table 20-29. ADCSOCFRC1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	SOC8	R-0/W1S	0h	SOC8 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC8 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC8 flag bit to 1. This will cause a conversion to start once priority is given to SOC8. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC8 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
7	SOC7	R-0/W1S	0h	SOC7 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC7 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC7 flag bit to 1. This will cause a conversion to start once priority is given to SOC7. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC7 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
6	SOC6	R-0/W1S	0h	SOC6 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC6 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC6 flag bit to 1. This will cause a conversion to start once priority is given to SOC6. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC6 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
5	SOC5	R-0/W1S	0h	SOC5 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC5 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC5 flag bit to 1. This will cause a conversion to start once priority is given to SOC5. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC5 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn

Table 20-29. ADCSOCFRC1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	SOC4	R-0/W1S	0h	SOC4 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC4 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC4 flag bit to 1. This will cause a conversion to start once priority is given to SOC4. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC4 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
3	SOC3	R-0/W1S	0h	SOC3 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC3 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC3 flag bit to 1. This will cause a conversion to start once priority is given to SOC3. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC3 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
2	SOC2	R-0/W1S	0h	SOC2 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC2 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC2 flag bit to 1. This will cause a conversion to start once priority is given to SOC2. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC2 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
1	SOC1	R-0/W1S	0h	SOC1 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC1 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC1 flag bit to 1. This will cause a conversion to start once priority is given to SOC1. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC1 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn

Table 20-29. ADCSOCFRC1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	SOC0	R-0/W1S	0h	SOC0 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC0 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC0 flag bit to 1. This will cause a conversion to start once priority is given to SOC0. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC0 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn

20.15.2.15 ADCSOCOVF1 Register (Offset = Eh) [Reset = 0h]

ADCSOCOVF1 is shown in Figure 20-40 and described in Table 20-30.

Return to the Summary Table.

ADC SOC Overflow 1 Register

Figure 20-40. ADCSOCOVF1 Register

15	14	13	12	11	10	9	8
SOC15	SOC14	SOC13	SOC12	SOC11	SOC10	SOC9	SOC8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
SOC7	SOC6	SOC5	SOC4	SOC3	SOC2	SOC1	SOC0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 20-30. ADCSOCOVF1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SOC15	R	0h	SOC15 Start of Conversion Overflow Flag. Indicates an SOC15 event was generated in hardware while an existing SOC15 event was already pending. 0 No SOC15 event overflow. 1 SOC15 event overflow. An overflow condition does not stop SOC15 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
14	SOC14	R	0h	SOC14 Start of Conversion Overflow Flag. Indicates an SOC14 event was generated in hardware while an existing SOC14 event was already pending. 0 No SOC14 event overflow. 1 SOC14 event overflow. An overflow condition does not stop SOC14 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
13	SOC13	R	0h	SOC13 Start of Conversion Overflow Flag. Indicates an SOC13 event was generated in hardware while an existing SOC13 event was already pending. 0 No SOC13 event overflow. 1 SOC13 event overflow. An overflow condition does not stop SOC13 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
12	SOC12	R	0h	SOC12 Start of Conversion Overflow Flag. Indicates an SOC12 event was generated in hardware while an existing SOC12 event was already pending. 0 No SOC12 event overflow. 1 SOC12 event overflow. An overflow condition does not stop SOC12 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn

Table 20-30. ADCSOCOVF1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	SOC11	R	0h	SOC11 Start of Conversion Overflow Flag. Indicates an SOC11 event was generated in hardware while an existing SOC11 event was already pending. 0 No SOC11 event overflow. 1 SOC11 event overflow. An overflow condition does not stop SOC11 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
10	SOC10	R	0h	SOC10 Start of Conversion Overflow Flag. Indicates an SOC10 event was generated in hardware while an existing SOC10 event was already pending. 0 No SOC10 event overflow. 1 SOC10 event overflow. An overflow condition does not stop SOC10 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
9	SOC9	R	0h	SOC9 Start of Conversion Overflow Flag. Indicates an SOC9 event was generated in hardware while an existing SOC9 event was already pending. 0 No SOC9 event overflow. 1 SOC9 event overflow. An overflow condition does not stop SOC9 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
8	SOC8	R	0h	SOC8 Start of Conversion Overflow Flag. Indicates an SOC8 event was generated in hardware while an existing SOC8 event was already pending. 0 No SOC8 event overflow. 1 SOC8 event overflow. An overflow condition does not stop SOC8 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
7	SOC7	R	0h	SOC7 Start of Conversion Overflow Flag. Indicates an SOC7 event was generated in hardware while an existing SOC7 event was already pending. 0 No SOC7 event overflow. 1 SOC7 event overflow. An overflow condition does not stop SOC7 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
6	SOC6	R	0h	SOC6 Start of Conversion Overflow Flag. Indicates an SOC6 event was generated in hardware while an existing SOC6 event was already pending. 0 No SOC6 event overflow. 1 SOC6 event overflow. An overflow condition does not stop SOC6 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
5	SOC5	R	0h	SOC5 Start of Conversion Overflow Flag. Indicates an SOC5 event was generated in hardware while an existing SOC5 event was already pending. 0 No SOC5 event overflow. 1 SOC5 event overflow. An overflow condition does not stop SOC5 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn

Table 20-30. ADCSOCOVF1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	SOC4	R	0h	SOC4 Start of Conversion Overflow Flag. Indicates an SOC4 event was generated in hardware while an existing SOC4 event was already pending. 0 No SOC4 event overflow. 1 SOC4 event overflow. An overflow condition does not stop SOC4 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
3	SOC3	R	0h	SOC3 Start of Conversion Overflow Flag. Indicates an SOC3 event was generated in hardware while an existing SOC3 event was already pending. 0 No SOC3 event overflow. 1 SOC3 event overflow. An overflow condition does not stop SOC3 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
2	SOC2	R	0h	SOC2 Start of Conversion Overflow Flag. Indicates an SOC2 event was generated in hardware while an existing SOC2 event was already pending. 0 No SOC2 event overflow. 1 SOC2 event overflow. An overflow condition does not stop SOC2 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
1	SOC1	R	0h	SOC1 Start of Conversion Overflow Flag. Indicates an SOC1 event was generated in hardware while an existing SOC1 event was already pending. 0 No SOC1 event overflow. 1 SOC1 event overflow. An overflow condition does not stop SOC1 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
0	SOC0	R	0h	SOC0 Start of Conversion Overflow Flag. Indicates an SOC0 event was generated in hardware while an existing SOC0 event was already pending. 0 No SOC0 event overflow. 1 SOC0 event overflow. An overflow condition does not stop SOC0 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn

20.15.2.16 ADCSOCOVFCLR1 Register (Offset = Fh) [Reset = 0h]

ADCSOCOVFCLR1 is shown in Figure 20-41 and described in Table 20-31.

Return to the Summary Table.

ADC SOC Overflow Clear 1 Register

Figure 20-41. ADCSOCOVFCLR1 Register

15	14	13	12	11	10	9	8
SOC15	SOC14	SOC13	SOC12	SOC11	SOC10	SOC9	SOC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
SOC7	SOC6	SOC5	SOC4	SOC3	SOC2	SOC1	SOC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 20-31. ADCSOCOVFCLR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SOC15	R-0/W1S	0h	SOC15 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC15 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC15 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
14	SOC14	R-0/W1S	0h	SOC14 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC14 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC14 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
13	SOC13	R-0/W1S	0h	SOC13 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC13 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC13 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
12	SOC12	R-0/W1S	0h	SOC12 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC12 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC12 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn

Table 20-31. ADCSOCOVFCLR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	SOC11	R-0/W1S	0h	SOC11 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC11 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC11 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
10	SOC10	R-0/W1S	0h	SOC10 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC10 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC10 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
9	SOC9	R-0/W1S	0h	SOC9 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC9 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC9 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
8	SOC8	R-0/W1S	0h	SOC8 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC8 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC8 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
7	SOC7	R-0/W1S	0h	SOC7 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC7 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC7 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
6	SOC6	R-0/W1S	0h	SOC6 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC6 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC6 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
5	SOC5	R-0/W1S	0h	SOC5 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC5 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC5 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn

Table 20-31. ADCSOCOVFCLR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	SOC4	R-0/W1S	0h	SOC4 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC4 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC4 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
3	SOC3	R-0/W1S	0h	SOC3 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC3 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC3 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
2	SOC2	R-0/W1S	0h	SOC2 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC2 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC2 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
1	SOC1	R-0/W1S	0h	SOC1 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC1 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC1 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
0	SOC0	R-0/W1S	0h	SOC0 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC0 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC0 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn

20.15.2.17 ADCSOC0CTL Register (Offset = 10h) [Reset = 0h]

ADCSOC0CTL is shown in Figure 20-42 and described in Table 20-32.

Return to the Summary Table.

ADC SOC0 Control Register

Figure 20-42. ADCSOC0CTL Register

31	30	29	28	27	26	25	24
RESERVED						TRIGSEL	
R-0h						R/W-0h	
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED		CHSEL	
R/W-0h				R-0h		R/W-0h	
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 20-32. ADCSOC0CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved

Table 20-32. ADCSOC0CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-20	TRIGSEL	R/W	0h	SOC0 Trigger Source Select. Along with the SOC0 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC0 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h ADCTRIG21 - ePWM9, ADCSOCA 16h ADCTRIG22 - ePWM9, ADCSOCB 17h ADCTRIG23 - ePWM10, ADCSOCA 18h ADCTRIG24 - ePWM10, ADCSOCB 19h ADCTRIG25 - ePWM11, ADCSOCA 1Ah ADCTRIG26 - ePWM11, ADCSOCB 1Bh ADCTRIG27 - ePWM12, ADCSOCA 1Ch ADCTRIG28 - ePWM12, ADCSOCB 1Dh ADCTRIG29 - CPU2 Timer 0, TINT0n 1Eh ADCTRIG30 - CPU2 Timer 1, TINT1n 1Fh ADCTRIG31 - CPU2 Timer 2, TINT2n 20h ADCTRIM32 - ePWM13, ADCSOCA 21h ADCTRIM33 - ePWM13, ADCSOCB 22h ADCTRIM34 - ePWM14, ADCSOCA 23h ADCTRIM35 - ePWM14, ADCSOCB 24h ADCTRIM36 - ePWM15, ADCSOCA 25h ADCTRIM37 - ePWM15, ADCSOCB 26h ADCTRIM38 - ePWM16, ADCSOCA 27h ADCTRIM39 - ePWM16, ADCSOCB 28h - 3Fh Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 20-32. ADCSOC0CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC0 Channel Select. Selects the channel to be converted when SOC0 is received by the ADC. Single-ended Signaling Mode (SIGNALMODE = 0): 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Differential Signaling Mode (SIGNALMODE = 1): 0h ADCIN0 (non-inverting) and ADCIN1 (inverting) 1h ADCIN0 (non-inverting) and ADCIN1 (inverting) 2h ADCIN2 (non-inverting) and ADCIN3 (inverting) 3h ADCIN2 (non-inverting) and ADCIN3 (inverting) 4h ADCIN4 (non-inverting) and ADCIN5 (inverting) 5h ADCIN4 (non-inverting) and ADCIN5 (inverting) 6h ADCIN6 (non-inverting) and ADCIN7 (inverting) 7h ADCIN6 (non-inverting) and ADCIN7 (inverting) 8h ADCIN8 (non-inverting) and ADCIN9 (inverting) 9h ADCIN8 (non-inverting) and ADCIN9 (inverting) Ah ADCIN10 (non-inverting) and ADCIN11 (inverting) Bh ADCIN10 (non-inverting) and ADCIN11 (inverting) Ch ADCIN12 (non-inverting) and ADCIN13 (inverting) Dh ADCIN12 (non-inverting) and ADCIN13 (inverting) Eh ADCIN14 (non-inverting) and ADCIN15 (inverting) Fh ADCIN14 (non-inverting) and ADCIN15 (inverting) Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC0 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

20.15.2.18 ADCSOC1CTL Register (Offset = 12h) [Reset = 0h]

ADCSOC1CTL is shown in Figure 20-43 and described in Table 20-33.

Return to the Summary Table.

ADC SOC1 Control Register

Figure 20-43. ADCSOC1CTL Register

31	30	29	28	27	26	25	24
RESERVED						TRIGSEL	
R-0h						R/W-0h	
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 20-33. ADCSOC1CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved

Table 20-33. ADCSOC1CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-20	TRIGSEL	R/W	0h	SOC1 Trigger Source Select. Along with the SOC1 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC1 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h ADCTRIG21 - ePWM9, ADCSOCA 16h ADCTRIG22 - ePWM9, ADCSOCB 17h ADCTRIG23 - ePWM10, ADCSOCA 18h ADCTRIG24 - ePWM10, ADCSOCB 19h ADCTRIG25 - ePWM11, ADCSOCA 1Ah ADCTRIG26 - ePWM11, ADCSOCB 1Bh ADCTRIG27 - ePWM12, ADCSOCA 1Ch ADCTRIG28 - ePWM12, ADCSOCB 1Dh ADCTRIG29 - CPU2 Timer 0, TINT0n 1Eh ADCTRIG30 - CPU2 Timer 1, TINT1n 1Fh ADCTRIG31 - CPU2 Timer 2, TINT2n 20h ADCTRIM32 - ePWM13, ADCSOCA 21h ADCTRIM33 - ePWM13, ADCSOCB 22h ADCTRIM34 - ePWM14, ADCSOCA 23h ADCTRIM35 - ePWM14, ADCSOCB 24h ADCTRIM36 - ePWM15, ADCSOCA 25h ADCTRIM37 - ePWM15, ADCSOCB 26h ADCTRIM38 - ePWM16, ADCSOCA 27h ADCTRIM39 - ePWM16, ADCSOCB 28h - 3Fh Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 20-33. ADCSOC1CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC1 Channel Select. Selects the channel to be converted when SOC1 is received by the ADC. Single-ended Signaling Mode (SIGNALMODE = 0): 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Differential Signaling Mode (SIGNALMODE = 1): 0h ADCIN0 (non-inverting) and ADCIN1 (inverting) 1h ADCIN0 (non-inverting) and ADCIN1 (inverting) 2h ADCIN2 (non-inverting) and ADCIN3 (inverting) 3h ADCIN2 (non-inverting) and ADCIN3 (inverting) 4h ADCIN4 (non-inverting) and ADCIN5 (inverting) 5h ADCIN4 (non-inverting) and ADCIN5 (inverting) 6h ADCIN6 (non-inverting) and ADCIN7 (inverting) 7h ADCIN6 (non-inverting) and ADCIN7 (inverting) 8h ADCIN8 (non-inverting) and ADCIN9 (inverting) 9h ADCIN8 (non-inverting) and ADCIN9 (inverting) Ah ADCIN10 (non-inverting) and ADCIN11 (inverting) Bh ADCIN10 (non-inverting) and ADCIN11 (inverting) Ch ADCIN12 (non-inverting) and ADCIN13 (inverting) Dh ADCIN12 (non-inverting) and ADCIN13 (inverting) Eh ADCIN14 (non-inverting) and ADCIN15 (inverting) Fh ADCIN14 (non-inverting) and ADCIN15 (inverting) Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC1 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

20.15.2.19 ADCSOC2CTL Register (Offset = 14h) [Reset = 0h]

ADCSOC2CTL is shown in Figure 20-44 and described in Table 20-34.

Return to the Summary Table.

ADC SOC2 Control Register

Figure 20-44. ADCSOC2CTL Register

31	30	29	28	27	26	25	24
RESERVED						TRIGSEL	
R-0h						R/W-0h	
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED		CHSEL	
R/W-0h				R-0h		R/W-0h	
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 20-34. ADCSOC2CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved

Table 20-34. ADCSOC2CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-20	TRIGSEL	R/W	0h	SOC2 Trigger Source Select. Along with the SOC2 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC2 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h ADCTRIG21 - ePWM9, ADCSOCA 16h ADCTRIG22 - ePWM9, ADCSOCB 17h ADCTRIG23 - ePWM10, ADCSOCA 18h ADCTRIG24 - ePWM10, ADCSOCB 19h ADCTRIG25 - ePWM11, ADCSOCA 1Ah ADCTRIG26 - ePWM11, ADCSOCB 1Bh ADCTRIG27 - ePWM12, ADCSOCA 1Ch ADCTRIG28 - ePWM12, ADCSOCB 1Dh ADCTRIG29 - CPU2 Timer 0, TINT0n 1Eh ADCTRIG30 - CPU2 Timer 1, TINT1n 1Fh ADCTRIG31 - CPU2 Timer 2, TINT2n 20h ADCTRIM32 - ePWM13, ADCSOCA 21h ADCTRIM33 - ePWM13, ADCSOCB 22h ADCTRIM34 - ePWM14, ADCSOCA 23h ADCTRIM35 - ePWM14, ADCSOCB 24h ADCTRIM36 - ePWM15, ADCSOCA 25h ADCTRIM37 - ePWM15, ADCSOCB 26h ADCTRIM38 - ePWM16, ADCSOCA 27h ADCTRIM39 - ePWM16, ADCSOCB 28h - 3Fh Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 20-34. ADCSOC2CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC2 Channel Select. Selects the channel to be converted when SOC2 is received by the ADC. Single-ended Signaling Mode (SIGNALMODE = 0): 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Differential Signaling Mode (SIGNALMODE = 1): 0h ADCIN0 (non-inverting) and ADCIN1 (inverting) 1h ADCIN0 (non-inverting) and ADCIN1 (inverting) 2h ADCIN2 (non-inverting) and ADCIN3 (inverting) 3h ADCIN2 (non-inverting) and ADCIN3 (inverting) 4h ADCIN4 (non-inverting) and ADCIN5 (inverting) 5h ADCIN4 (non-inverting) and ADCIN5 (inverting) 6h ADCIN6 (non-inverting) and ADCIN7 (inverting) 7h ADCIN6 (non-inverting) and ADCIN7 (inverting) 8h ADCIN8 (non-inverting) and ADCIN9 (inverting) 9h ADCIN8 (non-inverting) and ADCIN9 (inverting) Ah ADCIN10 (non-inverting) and ADCIN11 (inverting) Bh ADCIN10 (non-inverting) and ADCIN11 (inverting) Ch ADCIN12 (non-inverting) and ADCIN13 (inverting) Dh ADCIN12 (non-inverting) and ADCIN13 (inverting) Eh ADCIN14 (non-inverting) and ADCIN15 (inverting) Fh ADCIN14 (non-inverting) and ADCIN15 (inverting) Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC2 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

20.15.2.20 ADCSOC3CTL Register (Offset = 16h) [Reset = 0h]

ADCSOC3CTL is shown in Figure 20-45 and described in Table 20-35.

Return to the Summary Table.

ADC SOC3 Control Register

Figure 20-45. ADCSOC3CTL Register

31	30	29	28	27	26	25	24
RESERVED						TRIGSEL	
R-0h						R/W-0h	
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 20-35. ADCSOC3CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved

Table 20-35. ADCSOC3CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-20	TRIGSEL	R/W	0h	SOC3 Trigger Source Select. Along with the SOC3 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC3 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h ADCTRIG21 - ePWM9, ADCSOCA 16h ADCTRIG22 - ePWM9, ADCSOCB 17h ADCTRIG23 - ePWM10, ADCSOCA 18h ADCTRIG24 - ePWM10, ADCSOCB 19h ADCTRIG25 - ePWM11, ADCSOCA 1Ah ADCTRIG26 - ePWM11, ADCSOCB 1Bh ADCTRIG27 - ePWM12, ADCSOCA 1Ch ADCTRIG28 - ePWM12, ADCSOCB 1Dh ADCTRIG29 - CPU2 Timer 0, TINT0n 1Eh ADCTRIG30 - CPU2 Timer 1, TINT1n 1Fh ADCTRIG31 - CPU2 Timer 2, TINT2n 20h ADCTRIM32 - ePWM13, ADCSOCA 21h ADCTRIM33 - ePWM13, ADCSOCB 22h ADCTRIM34 - ePWM14, ADCSOCA 23h ADCTRIM35 - ePWM14, ADCSOCB 24h ADCTRIM36 - ePWM15, ADCSOCA 25h ADCTRIM37 - ePWM15, ADCSOCB 26h ADCTRIM38 - ePWM16, ADCSOCA 27h ADCTRIM39 - ePWM16, ADCSOCB 28h - 3Fh Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 20-35. ADCSOC3CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC3 Channel Select. Selects the channel to be converted when SOC3 is received by the ADC. Single-ended Signaling Mode (SIGNALMODE = 0): 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Differential Signaling Mode (SIGNALMODE = 1): 0h ADCIN0 (non-inverting) and ADCIN1 (inverting) 1h ADCIN0 (non-inverting) and ADCIN1 (inverting) 2h ADCIN2 (non-inverting) and ADCIN3 (inverting) 3h ADCIN2 (non-inverting) and ADCIN3 (inverting) 4h ADCIN4 (non-inverting) and ADCIN5 (inverting) 5h ADCIN4 (non-inverting) and ADCIN5 (inverting) 6h ADCIN6 (non-inverting) and ADCIN7 (inverting) 7h ADCIN6 (non-inverting) and ADCIN7 (inverting) 8h ADCIN8 (non-inverting) and ADCIN9 (inverting) 9h ADCIN8 (non-inverting) and ADCIN9 (inverting) Ah ADCIN10 (non-inverting) and ADCIN11 (inverting) Bh ADCIN10 (non-inverting) and ADCIN11 (inverting) Ch ADCIN12 (non-inverting) and ADCIN13 (inverting) Dh ADCIN12 (non-inverting) and ADCIN13 (inverting) Eh ADCIN14 (non-inverting) and ADCIN15 (inverting) Fh ADCIN14 (non-inverting) and ADCIN15 (inverting) Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC3 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

20.15.2.21 ADCSOC4CTL Register (Offset = 18h) [Reset = 0h]

ADCSOC4CTL is shown in Figure 20-46 and described in Table 20-36.

Return to the Summary Table.

ADC SOC4 Control Register

Figure 20-46. ADCSOC4CTL Register

31	30	29	28	27	26	25	24
RESERVED						TRIGSEL	
R-0h						R/W-0h	
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED		CHSEL	
R/W-0h				R-0h		R/W-0h	
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 20-36. ADCSOC4CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved

Table 20-36. ADCSOC4CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-20	TRIGSEL	R/W	0h	SOC4 Trigger Source Select. Along with the SOC4 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC4 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h ADCTRIG21 - ePWM9, ADCSOCA 16h ADCTRIG22 - ePWM9, ADCSOCB 17h ADCTRIG23 - ePWM10, ADCSOCA 18h ADCTRIG24 - ePWM10, ADCSOCB 19h ADCTRIG25 - ePWM11, ADCSOCA 1Ah ADCTRIG26 - ePWM11, ADCSOCB 1Bh ADCTRIG27 - ePWM12, ADCSOCA 1Ch ADCTRIG28 - ePWM12, ADCSOCB 1Dh ADCTRIG29 - CPU2 Timer 0, TINT0n 1Eh ADCTRIG30 - CPU2 Timer 1, TINT1n 1Fh ADCTRIG31 - CPU2 Timer 2, TINT2n 20h ADCTRIM32 - ePWM13, ADCSOCA 21h ADCTRIM33 - ePWM13, ADCSOCB 22h ADCTRIM34 - ePWM14, ADCSOCA 23h ADCTRIM35 - ePWM14, ADCSOCB 24h ADCTRIM36 - ePWM15, ADCSOCA 25h ADCTRIM37 - ePWM15, ADCSOCB 26h ADCTRIM38 - ePWM16, ADCSOCA 27h ADCTRIM39 - ePWM16, ADCSOCB 28h - 3Fh Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 20-36. ADCSOC4CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC4 Channel Select. Selects the channel to be converted when SOC4 is received by the ADC. Single-ended Signaling Mode (SIGNALMODE = 0): 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Differential Signaling Mode (SIGNALMODE = 1): 0h ADCIN0 (non-inverting) and ADCIN1 (inverting) 1h ADCIN0 (non-inverting) and ADCIN1 (inverting) 2h ADCIN2 (non-inverting) and ADCIN3 (inverting) 3h ADCIN2 (non-inverting) and ADCIN3 (inverting) 4h ADCIN4 (non-inverting) and ADCIN5 (inverting) 5h ADCIN4 (non-inverting) and ADCIN5 (inverting) 6h ADCIN6 (non-inverting) and ADCIN7 (inverting) 7h ADCIN6 (non-inverting) and ADCIN7 (inverting) 8h ADCIN8 (non-inverting) and ADCIN9 (inverting) 9h ADCIN8 (non-inverting) and ADCIN9 (inverting) Ah ADCIN10 (non-inverting) and ADCIN11 (inverting) Bh ADCIN10 (non-inverting) and ADCIN11 (inverting) Ch ADCIN12 (non-inverting) and ADCIN13 (inverting) Dh ADCIN12 (non-inverting) and ADCIN13 (inverting) Eh ADCIN14 (non-inverting) and ADCIN15 (inverting) Fh ADCIN14 (non-inverting) and ADCIN15 (inverting) Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC4 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

20.15.2.22 ADCSOC5CTL Register (Offset = 1Ah) [Reset = 0h]

ADCSOC5CTL is shown in Figure 20-47 and described in Table 20-37.

Return to the Summary Table.

ADC SOC5 Control Register

Figure 20-47. ADCSOC5CTL Register

31	30	29	28	27	26	25	24
RESERVED						TRIGSEL	
R-0h						R/W-0h	
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 20-37. ADCSOC5CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved

Table 20-37. ADCSOC5CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-20	TRIGSEL	R/W	0h	SOC5 Trigger Source Select. Along with the SOC5 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC5 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h ADCTRIG21 - ePWM9, ADCSOCA 16h ADCTRIG22 - ePWM9, ADCSOCB 17h ADCTRIG23 - ePWM10, ADCSOCA 18h ADCTRIG24 - ePWM10, ADCSOCB 19h ADCTRIG25 - ePWM11, ADCSOCA 1Ah ADCTRIG26 - ePWM11, ADCSOCB 1Bh ADCTRIG27 - ePWM12, ADCSOCA 1Ch ADCTRIG28 - ePWM12, ADCSOCB 1Dh ADCTRIG29 - CPU2 Timer 0, TINT0n 1Eh ADCTRIG30 - CPU2 Timer 1, TINT1n 1Fh ADCTRIG31 - CPU2 Timer 2, TINT2n 20h ADCTRIM32 - ePWM13, ADCSOCA 21h ADCTRIM33 - ePWM13, ADCSOCB 22h ADCTRIM34 - ePWM14, ADCSOCA 23h ADCTRIM35 - ePWM14, ADCSOCB 24h ADCTRIM36 - ePWM15, ADCSOCA 25h ADCTRIM37 - ePWM15, ADCSOCB 26h ADCTRIM38 - ePWM16, ADCSOCA 27h ADCTRIM39 - ePWM16, ADCSOCB 28h - 3Fh Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 20-37. ADCSOC5CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC5 Channel Select. Selects the channel to be converted when SOC5 is received by the ADC. Single-ended Signaling Mode (SIGNALMODE = 0): 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Differential Signaling Mode (SIGNALMODE = 1): 0h ADCIN0 (non-inverting) and ADCIN1 (inverting) 1h ADCIN0 (non-inverting) and ADCIN1 (inverting) 2h ADCIN2 (non-inverting) and ADCIN3 (inverting) 3h ADCIN2 (non-inverting) and ADCIN3 (inverting) 4h ADCIN4 (non-inverting) and ADCIN5 (inverting) 5h ADCIN4 (non-inverting) and ADCIN5 (inverting) 6h ADCIN6 (non-inverting) and ADCIN7 (inverting) 7h ADCIN6 (non-inverting) and ADCIN7 (inverting) 8h ADCIN8 (non-inverting) and ADCIN9 (inverting) 9h ADCIN8 (non-inverting) and ADCIN9 (inverting) Ah ADCIN10 (non-inverting) and ADCIN11 (inverting) Bh ADCIN10 (non-inverting) and ADCIN11 (inverting) Ch ADCIN12 (non-inverting) and ADCIN13 (inverting) Dh ADCIN12 (non-inverting) and ADCIN13 (inverting) Eh ADCIN14 (non-inverting) and ADCIN15 (inverting) Fh ADCIN14 (non-inverting) and ADCIN15 (inverting) Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC5 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

20.15.2.23 ADCSOC6CTL Register (Offset = 1Ch) [Reset = 0h]

ADCSOC6CTL is shown in Figure 20-48 and described in Table 20-38.

Return to the Summary Table.

ADC SOC6 Control Register

Figure 20-48. ADCSOC6CTL Register

31	30	29	28	27	26	25	24
RESERVED						TRIGSEL	
R-0h						R/W-0h	
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED		CHSEL	
R/W-0h				R-0h		R/W-0h	
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 20-38. ADCSOC6CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved

Table 20-38. ADCSOC6CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-20	TRIGSEL	R/W	0h	SOC6 Trigger Source Select. Along with the SOC6 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC6 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h ADCTRIG21 - ePWM9, ADCSOCA 16h ADCTRIG22 - ePWM9, ADCSOCB 17h ADCTRIG23 - ePWM10, ADCSOCA 18h ADCTRIG24 - ePWM10, ADCSOCB 19h ADCTRIG25 - ePWM11, ADCSOCA 1Ah ADCTRIG26 - ePWM11, ADCSOCB 1Bh ADCTRIG27 - ePWM12, ADCSOCA 1Ch ADCTRIG28 - ePWM12, ADCSOCB 1Dh ADCTRIG29 - CPU2 Timer 0, TINT0n 1Eh ADCTRIG30 - CPU2 Timer 1, TINT1n 1Fh ADCTRIG31 - CPU2 Timer 2, TINT2n 20h ADCTRIM32 - ePWM13, ADCSOCA 21h ADCTRIM33 - ePWM13, ADCSOCB 22h ADCTRIM34 - ePWM14, ADCSOCA 23h ADCTRIM35 - ePWM14, ADCSOCB 24h ADCTRIM36 - ePWM15, ADCSOCA 25h ADCTRIM37 - ePWM15, ADCSOCB 26h ADCTRIM38 - ePWM16, ADCSOCA 27h ADCTRIM39 - ePWM16, ADCSOCB 28h - 3Fh Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 20-38. ADCSOC6CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC6 Channel Select. Selects the channel to be converted when SOC6 is received by the ADC. Single-ended Signaling Mode (SIGNALMODE = 0): 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Differential Signaling Mode (SIGNALMODE = 1): 0h ADCIN0 (non-inverting) and ADCIN1 (inverting) 1h ADCIN0 (non-inverting) and ADCIN1 (inverting) 2h ADCIN2 (non-inverting) and ADCIN3 (inverting) 3h ADCIN2 (non-inverting) and ADCIN3 (inverting) 4h ADCIN4 (non-inverting) and ADCIN5 (inverting) 5h ADCIN4 (non-inverting) and ADCIN5 (inverting) 6h ADCIN6 (non-inverting) and ADCIN7 (inverting) 7h ADCIN6 (non-inverting) and ADCIN7 (inverting) 8h ADCIN8 (non-inverting) and ADCIN9 (inverting) 9h ADCIN8 (non-inverting) and ADCIN9 (inverting) Ah ADCIN10 (non-inverting) and ADCIN11 (inverting) Bh ADCIN10 (non-inverting) and ADCIN11 (inverting) Ch ADCIN12 (non-inverting) and ADCIN13 (inverting) Dh ADCIN12 (non-inverting) and ADCIN13 (inverting) Eh ADCIN14 (non-inverting) and ADCIN15 (inverting) Fh ADCIN14 (non-inverting) and ADCIN15 (inverting) Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC6 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

20.15.2.24 ADCSOC7CTL Register (Offset = 1Eh) [Reset = 0h]

ADCSOC7CTL is shown in Figure 20-49 and described in Table 20-39.

Return to the Summary Table.

ADC SOC7 Control Register

Figure 20-49. ADCSOC7CTL Register

31	30	29	28	27	26	25	24
RESERVED						TRIGSEL	
R-0h						R/W-0h	
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 20-39. ADCSOC7CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved

Table 20-39. ADCSOC7CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-20	TRIGSEL	R/W	0h	SOC7 Trigger Source Select. Along with the SOC7 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC7 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h ADCTRIG21 - ePWM9, ADCSOCA 16h ADCTRIG22 - ePWM9, ADCSOCB 17h ADCTRIG23 - ePWM10, ADCSOCA 18h ADCTRIG24 - ePWM10, ADCSOCB 19h ADCTRIG25 - ePWM11, ADCSOCA 1Ah ADCTRIG26 - ePWM11, ADCSOCB 1Bh ADCTRIG27 - ePWM12, ADCSOCA 1Ch ADCTRIG28 - ePWM12, ADCSOCB 1Dh ADCTRIG29 - CPU2 Timer 0, TINT0n 1Eh ADCTRIG30 - CPU2 Timer 1, TINT1n 1Fh ADCTRIG31 - CPU2 Timer 2, TINT2n 20h ADCTRIM32 - ePWM13, ADCSOCA 21h ADCTRIM33 - ePWM13, ADCSOCB 22h ADCTRIM34 - ePWM14, ADCSOCA 23h ADCTRIM35 - ePWM14, ADCSOCB 24h ADCTRIM36 - ePWM15, ADCSOCA 25h ADCTRIM37 - ePWM15, ADCSOCB 26h ADCTRIM38 - ePWM16, ADCSOCA 27h ADCTRIM39 - ePWM16, ADCSOCB 28h - 3Fh Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 20-39. ADCSOC7CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC7 Channel Select. Selects the channel to be converted when SOC7 is received by the ADC. Single-ended Signaling Mode (SIGNALMODE = 0): 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Differential Signaling Mode (SIGNALMODE = 1): 0h ADCIN0 (non-inverting) and ADCIN1 (inverting) 1h ADCIN0 (non-inverting) and ADCIN1 (inverting) 2h ADCIN2 (non-inverting) and ADCIN3 (inverting) 3h ADCIN2 (non-inverting) and ADCIN3 (inverting) 4h ADCIN4 (non-inverting) and ADCIN5 (inverting) 5h ADCIN4 (non-inverting) and ADCIN5 (inverting) 6h ADCIN6 (non-inverting) and ADCIN7 (inverting) 7h ADCIN6 (non-inverting) and ADCIN7 (inverting) 8h ADCIN8 (non-inverting) and ADCIN9 (inverting) 9h ADCIN8 (non-inverting) and ADCIN9 (inverting) Ah ADCIN10 (non-inverting) and ADCIN11 (inverting) Bh ADCIN10 (non-inverting) and ADCIN11 (inverting) Ch ADCIN12 (non-inverting) and ADCIN13 (inverting) Dh ADCIN12 (non-inverting) and ADCIN13 (inverting) Eh ADCIN14 (non-inverting) and ADCIN15 (inverting) Fh ADCIN14 (non-inverting) and ADCIN15 (inverting) Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC7 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

20.15.2.25 ADCSOC8CTL Register (Offset = 20h) [Reset = 0h]

ADCSOC8CTL is shown in Figure 20-50 and described in Table 20-40.

Return to the Summary Table.

ADC SOC8 Control Register

Figure 20-50. ADCSOC8CTL Register

31	30	29	28	27	26	25	24
RESERVED						TRIGSEL	
R-0h						R/W-0h	
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 20-40. ADCSOC8CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved

Table 20-40. ADCSOC8CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-20	TRIGSEL	R/W	0h	SOC8 Trigger Source Select. Along with the SOC8 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC8 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h ADCTRIG21 - ePWM9, ADCSOCA 16h ADCTRIG22 - ePWM9, ADCSOCB 17h ADCTRIG23 - ePWM10, ADCSOCA 18h ADCTRIG24 - ePWM10, ADCSOCB 19h ADCTRIG25 - ePWM11, ADCSOCA 1Ah ADCTRIG26 - ePWM11, ADCSOCB 1Bh ADCTRIG27 - ePWM12, ADCSOCA 1Ch ADCTRIG28 - ePWM12, ADCSOCB 1Dh ADCTRIG29 - CPU2 Timer 0, TINT0n 1Eh ADCTRIG30 - CPU2 Timer 1, TINT1n 1Fh ADCTRIG31 - CPU2 Timer 2, TINT2n 20h ADCTRIM32 - ePWM13, ADCSOCA 21h ADCTRIM33 - ePWM13, ADCSOCB 22h ADCTRIM34 - ePWM14, ADCSOCA 23h ADCTRIM35 - ePWM14, ADCSOCB 24h ADCTRIM36 - ePWM15, ADCSOCA 25h ADCTRIM37 - ePWM15, ADCSOCB 26h ADCTRIM38 - ePWM16, ADCSOCA 27h ADCTRIM39 - ePWM16, ADCSOCB 28h - 3Fh Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 20-40. ADCSOC8CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC8 Channel Select. Selects the channel to be converted when SOC8 is received by the ADC. Single-ended Signaling Mode (SIGNALMODE = 0): 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Differential Signaling Mode (SIGNALMODE = 1): 0h ADCIN0 (non-inverting) and ADCIN1 (inverting) 1h ADCIN0 (non-inverting) and ADCIN1 (inverting) 2h ADCIN2 (non-inverting) and ADCIN3 (inverting) 3h ADCIN2 (non-inverting) and ADCIN3 (inverting) 4h ADCIN4 (non-inverting) and ADCIN5 (inverting) 5h ADCIN4 (non-inverting) and ADCIN5 (inverting) 6h ADCIN6 (non-inverting) and ADCIN7 (inverting) 7h ADCIN6 (non-inverting) and ADCIN7 (inverting) 8h ADCIN8 (non-inverting) and ADCIN9 (inverting) 9h ADCIN8 (non-inverting) and ADCIN9 (inverting) Ah ADCIN10 (non-inverting) and ADCIN11 (inverting) Bh ADCIN10 (non-inverting) and ADCIN11 (inverting) Ch ADCIN12 (non-inverting) and ADCIN13 (inverting) Dh ADCIN12 (non-inverting) and ADCIN13 (inverting) Eh ADCIN14 (non-inverting) and ADCIN15 (inverting) Fh ADCIN14 (non-inverting) and ADCIN15 (inverting) Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC8 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

20.15.2.26 ADCSOC9CTL Register (Offset = 22h) [Reset = 0h]

ADCSOC9CTL is shown in Figure 20-51 and described in Table 20-41.

Return to the Summary Table.

ADC SOC9 Control Register

Figure 20-51. ADCSOC9CTL Register

31	30	29	28	27	26	25	24
RESERVED						TRIGSEL	
R-0h						R/W-0h	
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 20-41. ADCSOC9CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved

Table 20-41. ADCSOC9CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-20	TRIGSEL	R/W	0h	SOC9 Trigger Source Select. Along with the SOC9 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC9 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h ADCTRIG21 - ePWM9, ADCSOCA 16h ADCTRIG22 - ePWM9, ADCSOCB 17h ADCTRIG23 - ePWM10, ADCSOCA 18h ADCTRIG24 - ePWM10, ADCSOCB 19h ADCTRIG25 - ePWM11, ADCSOCA 1Ah ADCTRIG26 - ePWM11, ADCSOCB 1Bh ADCTRIG27 - ePWM12, ADCSOCA 1Ch ADCTRIG28 - ePWM12, ADCSOCB 1Dh ADCTRIG29 - CPU2 Timer 0, TINT0n 1Eh ADCTRIG30 - CPU2 Timer 1, TINT1n 1Fh ADCTRIG31 - CPU2 Timer 2, TINT2n 20h ADCTRIM32 - ePWM13, ADCSOCA 21h ADCTRIM33 - ePWM13, ADCSOCB 22h ADCTRIM34 - ePWM14, ADCSOCA 23h ADCTRIM35 - ePWM14, ADCSOCB 24h ADCTRIM36 - ePWM15, ADCSOCA 25h ADCTRIM37 - ePWM15, ADCSOCB 26h ADCTRIM38 - ePWM16, ADCSOCA 27h ADCTRIM39 - ePWM16, ADCSOCB 28h - 3Fh Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 20-41. ADCSOC9CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC9 Channel Select. Selects the channel to be converted when SOC9 is received by the ADC. Single-ended Signaling Mode (SIGNALMODE = 0): 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Differential Signaling Mode (SIGNALMODE = 1): 0h ADCIN0 (non-inverting) and ADCIN1 (inverting) 1h ADCIN0 (non-inverting) and ADCIN1 (inverting) 2h ADCIN2 (non-inverting) and ADCIN3 (inverting) 3h ADCIN2 (non-inverting) and ADCIN3 (inverting) 4h ADCIN4 (non-inverting) and ADCIN5 (inverting) 5h ADCIN4 (non-inverting) and ADCIN5 (inverting) 6h ADCIN6 (non-inverting) and ADCIN7 (inverting) 7h ADCIN6 (non-inverting) and ADCIN7 (inverting) 8h ADCIN8 (non-inverting) and ADCIN9 (inverting) 9h ADCIN8 (non-inverting) and ADCIN9 (inverting) Ah ADCIN10 (non-inverting) and ADCIN11 (inverting) Bh ADCIN10 (non-inverting) and ADCIN11 (inverting) Ch ADCIN12 (non-inverting) and ADCIN13 (inverting) Dh ADCIN12 (non-inverting) and ADCIN13 (inverting) Eh ADCIN14 (non-inverting) and ADCIN15 (inverting) Fh ADCIN14 (non-inverting) and ADCIN15 (inverting) Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC9 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

20.15.2.27 ADCSOC10CTL Register (Offset = 24h) [Reset = 0h]

ADCSOC10CTL is shown in Figure 20-52 and described in Table 20-42.

Return to the Summary Table.

ADC SOC10 Control Register

Figure 20-52. ADCSOC10CTL Register

31	30	29	28	27	26	25	24
RESERVED						TRIGSEL	
R-0h						R/W-0h	
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED		CHSEL	
R/W-0h				R-0h		R/W-0h	
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 20-42. ADCSOC10CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved

Table 20-42. ADCSOC10CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-20	TRIGSEL	R/W	0h	SOC10 Trigger Source Select. Along with the SOC10 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC10 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h ADCTRIG21 - ePWM9, ADCSOCA 16h ADCTRIG22 - ePWM9, ADCSOCB 17h ADCTRIG23 - ePWM10, ADCSOCA 18h ADCTRIG24 - ePWM10, ADCSOCB 19h ADCTRIG25 - ePWM11, ADCSOCA 1Ah ADCTRIG26 - ePWM11, ADCSOCB 1Bh ADCTRIG27 - ePWM12, ADCSOCA 1Ch ADCTRIG28 - ePWM12, ADCSOCB 1Dh ADCTRIG29 - CPU2 Timer 0, TINT0n 1Eh ADCTRIG30 - CPU2 Timer 1, TINT1n 1Fh ADCTRIG31 - CPU2 Timer 2, TINT2n 20h ADCTRIM32 - ePWM13, ADCSOCA 21h ADCTRIM33 - ePWM13, ADCSOCB 22h ADCTRIM34 - ePWM14, ADCSOCA 23h ADCTRIM35 - ePWM14, ADCSOCB 24h ADCTRIM36 - ePWM15, ADCSOCA 25h ADCTRIM37 - ePWM15, ADCSOCB 26h ADCTRIM38 - ePWM16, ADCSOCA 27h ADCTRIM39 - ePWM16, ADCSOCB 28h - 3Fh Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 20-42. ADCSOC10CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC10 Channel Select. Selects the channel to be converted when SOC10 is received by the ADC. Single-ended Signaling Mode (SIGNALMODE = 0): 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Differential Signaling Mode (SIGNALMODE = 1): 0h ADCIN0 (non-inverting) and ADCIN1 (inverting) 1h ADCIN0 (non-inverting) and ADCIN1 (inverting) 2h ADCIN2 (non-inverting) and ADCIN3 (inverting) 3h ADCIN2 (non-inverting) and ADCIN3 (inverting) 4h ADCIN4 (non-inverting) and ADCIN5 (inverting) 5h ADCIN4 (non-inverting) and ADCIN5 (inverting) 6h ADCIN6 (non-inverting) and ADCIN7 (inverting) 7h ADCIN6 (non-inverting) and ADCIN7 (inverting) 8h ADCIN8 (non-inverting) and ADCIN9 (inverting) 9h ADCIN8 (non-inverting) and ADCIN9 (inverting) Ah ADCIN10 (non-inverting) and ADCIN11 (inverting) Bh ADCIN10 (non-inverting) and ADCIN11 (inverting) Ch ADCIN12 (non-inverting) and ADCIN13 (inverting) Dh ADCIN12 (non-inverting) and ADCIN13 (inverting) Eh ADCIN14 (non-inverting) and ADCIN15 (inverting) Fh ADCIN14 (non-inverting) and ADCIN15 (inverting) Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC10 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

20.15.2.28 ADCSOC11CTL Register (Offset = 26h) [Reset = 0h]

ADCSOC11CTL is shown in Figure 20-53 and described in Table 20-43.

Return to the Summary Table.

ADC SOC11 Control Register

Figure 20-53. ADCSOC11CTL Register

31	30	29	28	27	26	25	24
RESERVED						TRIGSEL	
R-0h						R/W-0h	
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED		CHSEL	
R/W-0h				R-0h		R/W-0h	
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 20-43. ADCSOC11CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved

Table 20-43. ADCSOC11CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-20	TRIGSEL	R/W	0h	SOC11 Trigger Source Select. Along with the SOC11 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC11 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h ADCTRIG21 - ePWM9, ADCSOCA 16h ADCTRIG22 - ePWM9, ADCSOCB 17h ADCTRIG23 - ePWM10, ADCSOCA 18h ADCTRIG24 - ePWM10, ADCSOCB 19h ADCTRIG25 - ePWM11, ADCSOCA 1Ah ADCTRIG26 - ePWM11, ADCSOCB 1Bh ADCTRIG27 - ePWM12, ADCSOCA 1Ch ADCTRIG28 - ePWM12, ADCSOCB 1Dh ADCTRIG29 - CPU2 Timer 0, TINT0n 1Eh ADCTRIG30 - CPU2 Timer 1, TINT1n 1Fh ADCTRIG31 - CPU2 Timer 2, TINT2n 20h ADCTRIM32 - ePWM13, ADCSOCA 21h ADCTRIM33 - ePWM13, ADCSOCB 22h ADCTRIM34 - ePWM14, ADCSOCA 23h ADCTRIM35 - ePWM14, ADCSOCB 24h ADCTRIM36 - ePWM15, ADCSOCA 25h ADCTRIM37 - ePWM15, ADCSOCB 26h ADCTRIM38 - ePWM16, ADCSOCA 27h ADCTRIM39 - ePWM16, ADCSOCB 28h - 3Fh Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 20-43. ADCSOC11CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC11 Channel Select. Selects the channel to be converted when SOC11 is received by the ADC. Single-ended Signaling Mode (SIGNALMODE = 0): 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Differential Signaling Mode (SIGNALMODE = 1): 0h ADCIN0 (non-inverting) and ADCIN1 (inverting) 1h ADCIN0 (non-inverting) and ADCIN1 (inverting) 2h ADCIN2 (non-inverting) and ADCIN3 (inverting) 3h ADCIN2 (non-inverting) and ADCIN3 (inverting) 4h ADCIN4 (non-inverting) and ADCIN5 (inverting) 5h ADCIN4 (non-inverting) and ADCIN5 (inverting) 6h ADCIN6 (non-inverting) and ADCIN7 (inverting) 7h ADCIN6 (non-inverting) and ADCIN7 (inverting) 8h ADCIN8 (non-inverting) and ADCIN9 (inverting) 9h ADCIN8 (non-inverting) and ADCIN9 (inverting) Ah ADCIN10 (non-inverting) and ADCIN11 (inverting) Bh ADCIN10 (non-inverting) and ADCIN11 (inverting) Ch ADCIN12 (non-inverting) and ADCIN13 (inverting) Dh ADCIN12 (non-inverting) and ADCIN13 (inverting) Eh ADCIN14 (non-inverting) and ADCIN15 (inverting) Fh ADCIN14 (non-inverting) and ADCIN15 (inverting) Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC11 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

20.15.2.29 ADCSOC12CTL Register (Offset = 28h) [Reset = 0h]

ADCSOC12CTL is shown in Figure 20-54 and described in Table 20-44.

Return to the Summary Table.

ADC SOC12 Control Register

Figure 20-54. ADCSOC12CTL Register

31	30	29	28	27	26	25	24
RESERVED						TRIGSEL	
R-0h						R/W-0h	
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED		CHSEL	
R/W-0h				R-0h		R/W-0h	
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 20-44. ADCSOC12CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved

Table 20-44. ADCSOC12CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-20	TRIGSEL	R/W	0h	SOC12 Trigger Source Select. Along with the SOC12 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC12 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h ADCTRIG21 - ePWM9, ADCSOCA 16h ADCTRIG22 - ePWM9, ADCSOCB 17h ADCTRIG23 - ePWM10, ADCSOCA 18h ADCTRIG24 - ePWM10, ADCSOCB 19h ADCTRIG25 - ePWM11, ADCSOCA 1Ah ADCTRIG26 - ePWM11, ADCSOCB 1Bh ADCTRIG27 - ePWM12, ADCSOCA 1Ch ADCTRIG28 - ePWM12, ADCSOCB 1Dh ADCTRIG29 - CPU2 Timer 0, TINT0n 1Eh ADCTRIG30 - CPU2 Timer 1, TINT1n 1Fh ADCTRIG31 - CPU2 Timer 2, TINT2n 20h ADCTRIM32 - ePWM13, ADCSOCA 21h ADCTRIM33 - ePWM13, ADCSOCB 22h ADCTRIM34 - ePWM14, ADCSOCA 23h ADCTRIM35 - ePWM14, ADCSOCB 24h ADCTRIM36 - ePWM15, ADCSOCA 25h ADCTRIM37 - ePWM15, ADCSOCB 26h ADCTRIM38 - ePWM16, ADCSOCA 27h ADCTRIM39 - ePWM16, ADCSOCB 28h - 3Fh Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 20-44. ADCSOC12CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC12 Channel Select. Selects the channel to be converted when SOC12 is received by the ADC. Single-ended Signaling Mode (SIGNALMODE = 0): 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Differential Signaling Mode (SIGNALMODE = 1): 0h ADCIN0 (non-inverting) and ADCIN1 (inverting) 1h ADCIN0 (non-inverting) and ADCIN1 (inverting) 2h ADCIN2 (non-inverting) and ADCIN3 (inverting) 3h ADCIN2 (non-inverting) and ADCIN3 (inverting) 4h ADCIN4 (non-inverting) and ADCIN5 (inverting) 5h ADCIN4 (non-inverting) and ADCIN5 (inverting) 6h ADCIN6 (non-inverting) and ADCIN7 (inverting) 7h ADCIN6 (non-inverting) and ADCIN7 (inverting) 8h ADCIN8 (non-inverting) and ADCIN9 (inverting) 9h ADCIN8 (non-inverting) and ADCIN9 (inverting) Ah ADCIN10 (non-inverting) and ADCIN11 (inverting) Bh ADCIN10 (non-inverting) and ADCIN11 (inverting) Ch ADCIN12 (non-inverting) and ADCIN13 (inverting) Dh ADCIN12 (non-inverting) and ADCIN13 (inverting) Eh ADCIN14 (non-inverting) and ADCIN15 (inverting) Fh ADCIN14 (non-inverting) and ADCIN15 (inverting) Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC12 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

20.15.2.30 ADCSOC13CTL Register (Offset = 2Ah) [Reset = 0h]

ADCSOC13CTL is shown in Figure 20-55 and described in Table 20-45.

Return to the Summary Table.

ADC SOC13 Control Register

Figure 20-55. ADCSOC13CTL Register

31	30	29	28	27	26	25	24
RESERVED						TRIGSEL	
R-0h						R/W-0h	
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 20-45. ADCSOC13CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved

Table 20-45. ADCSOC13CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-20	TRIGSEL	R/W	0h	SOC13 Trigger Source Select. Along with the SOC13 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC13 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h ADCTRIG21 - ePWM9, ADCSOCA 16h ADCTRIG22 - ePWM9, ADCSOCB 17h ADCTRIG23 - ePWM10, ADCSOCA 18h ADCTRIG24 - ePWM10, ADCSOCB 19h ADCTRIG25 - ePWM11, ADCSOCA 1Ah ADCTRIG26 - ePWM11, ADCSOCB 1Bh ADCTRIG27 - ePWM12, ADCSOCA 1Ch ADCTRIG28 - ePWM12, ADCSOCB 1Dh ADCTRIG29 - CPU2 Timer 0, TINT0n 1Eh ADCTRIG30 - CPU2 Timer 1, TINT1n 1Fh ADCTRIG31 - CPU2 Timer 2, TINT2n 20h ADCTRIM32 - ePWM13, ADCSOCA 21h ADCTRIM33 - ePWM13, ADCSOCB 22h ADCTRIM34 - ePWM14, ADCSOCA 23h ADCTRIM35 - ePWM14, ADCSOCB 24h ADCTRIM36 - ePWM15, ADCSOCA 25h ADCTRIM37 - ePWM15, ADCSOCB 26h ADCTRIM38 - ePWM16, ADCSOCA 27h ADCTRIM39 - ePWM16, ADCSOCB 28h - 3Fh Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 20-45. ADCSOC13CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC13 Channel Select. Selects the channel to be converted when SOC13 is received by the ADC. Single-ended Signaling Mode (SIGNALMODE = 0): 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Differential Signaling Mode (SIGNALMODE = 1): 0h ADCIN0 (non-inverting) and ADCIN1 (inverting) 1h ADCIN0 (non-inverting) and ADCIN1 (inverting) 2h ADCIN2 (non-inverting) and ADCIN3 (inverting) 3h ADCIN2 (non-inverting) and ADCIN3 (inverting) 4h ADCIN4 (non-inverting) and ADCIN5 (inverting) 5h ADCIN4 (non-inverting) and ADCIN5 (inverting) 6h ADCIN6 (non-inverting) and ADCIN7 (inverting) 7h ADCIN6 (non-inverting) and ADCIN7 (inverting) 8h ADCIN8 (non-inverting) and ADCIN9 (inverting) 9h ADCIN8 (non-inverting) and ADCIN9 (inverting) Ah ADCIN10 (non-inverting) and ADCIN11 (inverting) Bh ADCIN10 (non-inverting) and ADCIN11 (inverting) Ch ADCIN12 (non-inverting) and ADCIN13 (inverting) Dh ADCIN12 (non-inverting) and ADCIN13 (inverting) Eh ADCIN14 (non-inverting) and ADCIN15 (inverting) Fh ADCIN14 (non-inverting) and ADCIN15 (inverting) Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC13 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

20.15.2.31 ADCSOC14CTL Register (Offset = 2Ch) [Reset = 0h]

ADCSOC14CTL is shown in Figure 20-56 and described in Table 20-46.

Return to the Summary Table.

ADC SOC14 Control Register

Figure 20-56. ADCSOC14CTL Register

31	30	29	28	27	26	25	24
RESERVED						TRIGSEL	
R-0h						R/W-0h	
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED		CHSEL	
R/W-0h				R-0h		R/W-0h	
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 20-46. ADCSOC14CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved

Table 20-46. ADCSOC14CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-20	TRIGSEL	R/W	0h	SOC14 Trigger Source Select. Along with the SOC14 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC14 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h ADCTRIG21 - ePWM9, ADCSOCA 16h ADCTRIG22 - ePWM9, ADCSOCB 17h ADCTRIG23 - ePWM10, ADCSOCA 18h ADCTRIG24 - ePWM10, ADCSOCB 19h ADCTRIG25 - ePWM11, ADCSOCA 1Ah ADCTRIG26 - ePWM11, ADCSOCB 1Bh ADCTRIG27 - ePWM12, ADCSOCA 1Ch ADCTRIG28 - ePWM12, ADCSOCB 1Dh ADCTRIG29 - CPU2 Timer 0, TINT0n 1Eh ADCTRIG30 - CPU2 Timer 1, TINT1n 1Fh ADCTRIG31 - CPU2 Timer 2, TINT2n 20h ADCTRIM32 - ePWM13, ADCSOCA 21h ADCTRIM33 - ePWM13, ADCSOCB 22h ADCTRIM34 - ePWM14, ADCSOCA 23h ADCTRIM35 - ePWM14, ADCSOCB 24h ADCTRIM36 - ePWM15, ADCSOCA 25h ADCTRIM37 - ePWM15, ADCSOCB 26h ADCTRIM38 - ePWM16, ADCSOCA 27h ADCTRIM39 - ePWM16, ADCSOCB 28h - 3Fh Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 20-46. ADCSOC14CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC14 Channel Select. Selects the channel to be converted when SOC14 is received by the ADC. Single-ended Signaling Mode (SIGNALMODE = 0): 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Differential Signaling Mode (SIGNALMODE = 1): 0h ADCIN0 (non-inverting) and ADCIN1 (inverting) 1h ADCIN0 (non-inverting) and ADCIN1 (inverting) 2h ADCIN2 (non-inverting) and ADCIN3 (inverting) 3h ADCIN2 (non-inverting) and ADCIN3 (inverting) 4h ADCIN4 (non-inverting) and ADCIN5 (inverting) 5h ADCIN4 (non-inverting) and ADCIN5 (inverting) 6h ADCIN6 (non-inverting) and ADCIN7 (inverting) 7h ADCIN6 (non-inverting) and ADCIN7 (inverting) 8h ADCIN8 (non-inverting) and ADCIN9 (inverting) 9h ADCIN8 (non-inverting) and ADCIN9 (inverting) Ah ADCIN10 (non-inverting) and ADCIN11 (inverting) Bh ADCIN10 (non-inverting) and ADCIN11 (inverting) Ch ADCIN12 (non-inverting) and ADCIN13 (inverting) Dh ADCIN12 (non-inverting) and ADCIN13 (inverting) Eh ADCIN14 (non-inverting) and ADCIN15 (inverting) Fh ADCIN14 (non-inverting) and ADCIN15 (inverting) Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC14 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

20.15.2.32 ADCSOC15CTL Register (Offset = 2Eh) [Reset = 0h]

ADCSOC15CTL is shown in Figure 20-57 and described in Table 20-47.

Return to the Summary Table.

ADC SOC15 Control Register

Figure 20-57. ADCSOC15CTL Register

31	30	29	28	27	26	25	24
RESERVED						TRIGSEL	
R-0h						R/W-0h	
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 20-47. ADCSOC15CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved

Table 20-47. ADCSOC15CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-20	TRIGSEL	R/W	0h	SOC15 Trigger Source Select. Along with the SOC15 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC15 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h ADCTRIG21 - ePWM9, ADCSOCA 16h ADCTRIG22 - ePWM9, ADCSOCB 17h ADCTRIG23 - ePWM10, ADCSOCA 18h ADCTRIG24 - ePWM10, ADCSOCB 19h ADCTRIG25 - ePWM11, ADCSOCA 1Ah ADCTRIG26 - ePWM11, ADCSOCB 1Bh ADCTRIG27 - ePWM12, ADCSOCA 1Ch ADCTRIG28 - ePWM12, ADCSOCB 1Dh ADCTRIG29 - CPU2 Timer 0, TINT0n 1Eh ADCTRIG30 - CPU2 Timer 1, TINT1n 1Fh ADCTRIG31 - CPU2 Timer 2, TINT2n 20h ADCTRIM32 - ePWM13, ADCSOCA 21h ADCTRIM33 - ePWM13, ADCSOCB 22h ADCTRIM34 - ePWM14, ADCSOCA 23h ADCTRIM35 - ePWM14, ADCSOCB 24h ADCTRIM36 - ePWM15, ADCSOCA 25h ADCTRIM37 - ePWM15, ADCSOCB 26h ADCTRIM38 - ePWM16, ADCSOCA 27h ADCTRIM39 - ePWM16, ADCSOCB 28h - 3Fh Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 20-47. ADCSOC15CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC15 Channel Select. Selects the channel to be converted when SOC15 is received by the ADC. Single-ended Signaling Mode (SIGNALMODE = 0): 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Differential Signaling Mode (SIGNALMODE = 1): 0h ADCIN0 (non-inverting) and ADCIN1 (inverting) 1h ADCIN0 (non-inverting) and ADCIN1 (inverting) 2h ADCIN2 (non-inverting) and ADCIN3 (inverting) 3h ADCIN2 (non-inverting) and ADCIN3 (inverting) 4h ADCIN4 (non-inverting) and ADCIN5 (inverting) 5h ADCIN4 (non-inverting) and ADCIN5 (inverting) 6h ADCIN6 (non-inverting) and ADCIN7 (inverting) 7h ADCIN6 (non-inverting) and ADCIN7 (inverting) 8h ADCIN8 (non-inverting) and ADCIN9 (inverting) 9h ADCIN8 (non-inverting) and ADCIN9 (inverting) Ah ADCIN10 (non-inverting) and ADCIN11 (inverting) Bh ADCIN10 (non-inverting) and ADCIN11 (inverting) Ch ADCIN12 (non-inverting) and ADCIN13 (inverting) Dh ADCIN12 (non-inverting) and ADCIN13 (inverting) Eh ADCIN14 (non-inverting) and ADCIN15 (inverting) Fh ADCIN14 (non-inverting) and ADCIN15 (inverting) Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC15 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

20.15.2.33 ADCEVTSTAT Register (Offset = 30h) [Reset = 0h]

ADCEVTSTAT is shown in Figure 20-58 and described in Table 20-48.

Return to the Summary Table.

ADC Event Status Register

Figure 20-58. ADCEVTSTAT Register

15	14	13	12	11	10	9	8
RESERVED	PPB4ZERO	PPB4TRIPLO	PPB4TRIPHI	RESERVED	PPB3ZERO	PPB3TRIPLO	PPB3TRIPHI
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	PPB2ZERO	PPB2TRIPLO	PPB2TRIPHI	RESERVED	PPB1ZERO	PPB1TRIPLO	PPB1TRIPHI
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 20-48. ADCEVTSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	PPB4ZERO	R	0h	Post Processing Block 4 Zero Crossing Flag. When set indicates the ADCPPB4RESULT register has changed sign. This bit is gated by EOC signal. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
13	PPB4TRIPLO	R	0h	Post Processing Block 4 Trip Low Flag. When set indicates a digital compare trip low event has occurred. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
12	PPB4TRIPHI	R	0h	Post Processing Block 4 Trip High Flag. When set indicates a digital compare trip high event has occurred. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
11	RESERVED	R	0h	Reserved
10	PPB3ZERO	R	0h	Post Processing Block 3 Zero Crossing Flag. When set indicates the ADCPPB3RESULT register has changed sign. This bit is gated by EOC signal. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn

Table 20-48. ADCEVTSTAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	PPB3TRIPLO	R	0h	Post Processing Block 3 Trip Low Flag. When set indicates a digital compare trip low event has occurred. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
8	PPB3TRIPHI	R	0h	Post Processing Block 3 Trip High Flag. When set indicates a digital compare trip high event has occurred. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6	PPB2ZERO	R	0h	Post Processing Block 2 Zero Crossing Flag. When set indicates the ADCPPB2RESULT register has changed sign. This bit is gated by EOC signal. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
5	PPB2TRIPLO	R	0h	Post Processing Block 2 Trip Low Flag. When set indicates a digital compare trip low event has occurred. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
4	PPB2TRIPHI	R	0h	Post Processing Block 2 Trip High Flag. When set indicates a digital compare trip high event has occurred. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
3	RESERVED	R	0h	Reserved
2	PPB1ZERO	R	0h	Post Processing Block 1 Zero Crossing Flag. When set indicates the ADCPPB1RESULT register has changed sign. This bit is gated by EOC signal. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn

Table 20-48. ADCEVTSTAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	PPB1TRIPLO	R	0h	Post Processing Block 1 Trip Low Flag. When set indicates a digital compare trip low event has occurred. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
0	PPB1TRIPHI	R	0h	Post Processing Block 1 Trip High Flag. When set indicates a digital compare trip high event has occurred. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn

20.15.2.34 ADCEVTCLR Register (Offset = 32h) [Reset = 0h]

ADCEVTCLR is shown in Figure 20-59 and described in Table 20-49.

Return to the Summary Table.

ADC Event Clear Register

Figure 20-59. ADCEVTCLR Register

15	14	13	12	11	10	9	8
RESERVED	PPB4ZERO	PPB4TRIPLO	PPB4TRIPHI	RESERVED	PPB3ZERO	PPB3TRIPLO	PPB3TRIPHI
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
RESERVED	PPB2ZERO	PPB2TRIPLO	PPB2TRIPHI	RESERVED	PPB1ZERO	PPB1TRIPLO	PPB1TRIPHI
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 20-49. ADCEVTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	PPB4ZERO	R-0/W1S	0h	Post Processing Block 4 Zero Crossing Clear. Clears the corresponding zero crossing flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
13	PPB4TRIPLO	R-0/W1S	0h	Post Processing Block 4 Trip Low Clear. Clears the corresponding trip low flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
12	PPB4TRIPHI	R-0/W1S	0h	Post Processing Block 4 Trip High Clear. Clears the corresponding trip high flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
11	RESERVED	R	0h	Reserved
10	PPB3ZERO	R-0/W1S	0h	Post Processing Block 3 Zero Crossing Clear. Clears the corresponding zero crossing flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
9	PPB3TRIPLO	R-0/W1S	0h	Post Processing Block 3 Trip Low Clear. Clears the corresponding trip low flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
8	PPB3TRIPHI	R-0/W1S	0h	Post Processing Block 3 Trip High Clear. Clears the corresponding trip high flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6	PPB2ZERO	R-0/W1S	0h	Post Processing Block 2 Zero Crossing Clear. Clears the corresponding zero crossing flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn

Table 20-49. ADCEVTCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	PPB2TRIPLO	R-0/W1S	0h	Post Processing Block 2 Trip Low Clear. Clears the corresponding trip low flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
4	PPB2TRIPHI	R-0/W1S	0h	Post Processing Block 2 Trip High Clear. Clears the corresponding trip high flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
3	RESERVED	R	0h	Reserved
2	PPB1ZERO	R-0/W1S	0h	Post Processing Block 1 Zero Crossing Clear. Clears the corresponding zero crossing flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
1	PPB1TRIPLO	R-0/W1S	0h	Post Processing Block 1 Trip Low Clear. Clears the corresponding trip low flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
0	PPB1TRIPHI	R-0/W1S	0h	Post Processing Block 1 Trip High Clear. Clears the corresponding trip high flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn

20.15.2.35 ADCEVTSEL Register (Offset = 34h) [Reset = 0h]

ADCEVTSEL is shown in Figure 20-60 and described in Table 20-50.

Return to the Summary Table.

ADC Event Selection Register

Figure 20-60. ADCEVTSEL Register

15	14	13	12	11	10	9	8
RESERVED	PPB4ZERO	PPB4TRIPLO	PPB4TRIPHI	RESERVED	PPB3ZERO	PPB3TRIPLO	PPB3TRIPHI
R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	PPB2ZERO	PPB2TRIPLO	PPB2TRIPHI	RESERVED	PPB1ZERO	PPB1TRIPLO	PPB1TRIPHI
R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h

Table 20-50. ADCEVTSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	PPB4ZERO	R/W	0h	Post Processing Block 4 Zero Crossing Event Enable. Setting this bit allows the corresponding rising zero crossing flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
13	PPB4TRIPLO	R/W	0h	Post Processing Block 4 Trip Low Event Enable. Setting this bit allows the corresponding rising trip low flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
12	PPB4TRIPHI	R/W	0h	Post Processing Block 4 Trip High Event Enable. Setting this bit allows the corresponding rising trip high flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
11	RESERVED	R	0h	Reserved
10	PPB3ZERO	R/W	0h	Post Processing Block 3 Zero Crossing Event Enable. Setting this bit allows the corresponding rising zero crossing flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
9	PPB3TRIPLO	R/W	0h	Post Processing Block 3 Trip Low Event Enable. Setting this bit allows the corresponding rising trip low flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
8	PPB3TRIPHI	R/W	0h	Post Processing Block 3 Trip High Event Enable. Setting this bit allows the corresponding rising trip high flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6	PPB2ZERO	R/W	0h	Post Processing Block 2 Zero Crossing Event Enable. Setting this bit allows the corresponding rising zero crossing flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn

Table 20-50. ADCEVTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	PPB2TRIPLO	R/W	0h	Post Processing Block 2 Trip Low Event Enable. Setting this bit allows the corresponding rising trip low flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
4	PPB2TRIPHI	R/W	0h	Post Processing Block 2 Trip High Event Enable. Setting this bit allows the corresponding rising trip high flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
3	RESERVED	R	0h	Reserved
2	PPB1ZERO	R/W	0h	Post Processing Block 1 Zero Crossing Event Enable. Setting this bit allows the corresponding rising zero crossing flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
1	PPB1TRIPLO	R/W	0h	Post Processing Block 1 Trip Low Event Enable. Setting this bit allows the corresponding rising trip low flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
0	PPB1TRIPHI	R/W	0h	Post Processing Block 1 Trip High Event Enable. Setting this bit allows the corresponding rising trip high flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn

20.15.2.36 ADCEVTINTSEL Register (Offset = 36h) [Reset = 0h]

ADCEVTINTSEL is shown in Figure 20-61 and described in Table 20-51.

Return to the Summary Table.

ADC Event Interrupt Selection Register

Figure 20-61. ADCEVTINTSEL Register

15	14	13	12	11	10	9	8
RESERVED	PPB4ZERO	PPB4TRIPLO	PPB4TRIPHI	RESERVED	PPB3ZERO	PPB3TRIPLO	PPB3TRIPHI
R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	PPB2ZERO	PPB2TRIPLO	PPB2TRIPHI	RESERVED	PPB1ZERO	PPB1TRIPLO	PPB1TRIPHI
R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h

Table 20-51. ADCEVTINTSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	PPB4ZERO	R/W	0h	Post Processing Block 4 Zero Crossing Interrupt Enable. Setting this bit allows the corresponding rising zero crossing flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
13	PPB4TRIPLO	R/W	0h	Post Processing Block 4 Trip Low Interrupt Enable. Setting this bit allows the corresponding rising trip low flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
12	PPB4TRIPHI	R/W	0h	Post Processing Block 4 Trip High Interrupt Enable. Setting this bit allows the corresponding rising trip high flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
11	RESERVED	R	0h	Reserved
10	PPB3ZERO	R/W	0h	Post Processing Block 3 Zero Crossing Interrupt Enable. Setting this bit allows the corresponding rising zero crossing flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
9	PPB3TRIPLO	R/W	0h	Post Processing Block 3 Trip Low Interrupt Enable. Setting this bit allows the corresponding rising trip low flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
8	PPB3TRIPHI	R/W	0h	Post Processing Block 3 Trip High Interrupt Enable. Setting this bit allows the corresponding rising trip high flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6	PPB2ZERO	R/W	0h	Post Processing Block 2 Zero Crossing Interrupt Enable. Setting this bit allows the corresponding rising zero crossing flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn

Table 20-51. ADCEVTINTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	PPB2TRIPLO	R/W	0h	Post Processing Block 2 Trip Low Interrupt Enable. Setting this bit allows the corresponding rising trip low flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
4	PPB2TRIPHI	R/W	0h	Post Processing Block 2 Trip High Interrupt Enable. Setting this bit allows the corresponding rising trip high flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
3	RESERVED	R	0h	Reserved
2	PPB1ZERO	R/W	0h	Post Processing Block 1 Zero Crossing Interrupt Enable. Setting this bit allows the corresponding rising zero crossing flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
1	PPB1TRIPLO	R/W	0h	Post Processing Block 1 Trip Low Interrupt Enable. Setting this bit allows the corresponding rising trip low flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
0	PPB1TRIPHI	R/W	0h	Post Processing Block 1 Trip High Interrupt Enable. Setting this bit allows the corresponding rising trip high flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn

20.15.2.37 ADCOSDETECT Register (Offset = 38h) [Reset = 0h]

ADCOSDETECT is shown in Figure 20-62 and described in Table 20-52.

Return to the Summary Table.

ADC Open and Shorts Detect Register

Figure 20-62. ADCOSDETECT Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					DETECTCFG		
R-0h					R/W-0h		

Table 20-52. ADCOSDETECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	Reserved
2-0	DETECTCFG	R/W	0h	ADC Opens and Shorts Detect Configuration. This bit field defines the open/shorts detection circuit state. 0h Open/Shorts detection circuit is disabled. 1h Open/Shorts detection circuit is enabled at zero scale. 2h Open/Shorts detection circuit is enabled at full scale. 3h Open/Shorts detection circuit is enabled at (nominal) 5/12 scale. 4h Open/Shorts detection circuit is enabled at (nominal) 7/12 scale. 5h Open/Shorts detection circuit is enabled with a (nominal) 5K pulldown to VSSA. 6h Open/Shorts detection circuit is enabled with a (nominal) 5K pullup to VDDA. 7h Open/Shorts detection circuit is enabled with a (nominal) 7K pulldown to VSSA. Reset type: SYSRSn

20.15.2.38 ADCCOUNTER Register (Offset = 39h) [Reset = 0h]

ADCCOUNTER is shown in Figure 20-63 and described in Table 20-53.

Return to the Summary Table.

ADC Counter Register

Figure 20-63. ADCCOUNTER Register

15	14	13	12	11	10	9	8
RESERVED				FREECOUNT			
R-0h				R-0h			
7	6	5	4	3	2	1	0
FREECOUNT							
R-0h							

Table 20-53. ADCCOUNTER Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	FREECOUNT	R	0h	ADC Free Running Counter Value. This bit field reflects the status of the free running ADC counter. Reset type: SYSRSn

20.15.2.39 ADCREV Register (Offset = 3Ah) [Reset = 105h]

ADCREV is shown in Figure 20-64 and described in Table 20-54.

Return to the Summary Table.

ADC Revision Register

Figure 20-64. ADCREV Register

15	14	13	12	11	10	9	8
REV							
R-1h							
7	6	5	4	3	2	1	0
TYPE							
R-5h							

Table 20-54. ADCREV Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	REV	R	1h	ADC Revision. To allow documentation of differences between revisions. First version is labeled as 00h. Reset type: SYSRSn
7-0	TYPE	R	5h	ADC Type. Always set to 5 for this ADC. Reset type: SYSRSn

20.15.2.40 ADCOFFTRIM Register (Offset = 3Bh) [Reset = 0h]

ADCOFFTRIM is shown in Figure 20-65 and described in Table 20-55.

Return to the Summary Table.

ADC Offset Trim Register

Figure 20-65. ADCOFFTRIM Register

15	14	13	12	11	10	9	8
RESERVED				RESERVED			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
OFFTRIM							
R/W-0h							

Table 20-55. ADCOFFTRIM Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-8	RESERVED	R/W	0h	Reserved
7-0	OFFTRIM	R/W	0h	ADC Offset Trim. Adjusts the conversion results of the converter up or down to account for offset error in the ADC. A different offset trim is required for each combination of resolution and signal mode. Using the AdcSetMode function to set the resolution and signal mode will ensure that the correct offset trim is loaded. Range is +127 steps to -128 steps (2's compliment format). Regardless of the converter resolution, the size of each trim step is (VREFHI-VREFLO)/65536. Reset type: SYSRSn

20.15.2.41 ADCPPB1CONFIG Register (Offset = 40h) [Reset = 0h]

ADCPPB1CONFIG is shown in Figure 20-66 and described in Table 20-56.

Return to the Summary Table.

ADC PPB1 Config Register

Figure 20-66. ADCPPB1CONFIG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		CBCEN	TWOSCOMPEN	CONFIG			
R-0h		R/W-0h	R/W-0h	R/W-0h			

Table 20-56. ADCPPB1CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-6	RESERVED	R	0h	Reserved
5	CBCEN	R/W	0h	ADC Post Processing Block Cycle By Cycle Enable. When set, this bit enables the post conversion hardware processing circuit to automatically clear the ADCEVTSTAT on a conversion if the event condition is no longer present. Reset type: SYSRSn
4	TWOSCOMPEN	R/W	0h	ADC Post Processing Block 1 Two's Complement Enable. When set this bit enables the post conversion hardware processing circuit that performs a two's complement on the output of the offset/reference subtraction unit before storing the result in the ADCPPB1RESULT register. 0 ADCPPB1RESULT = ADCRESULTx - ADCPPB1OFFREF 1 ADCPPB1RESULT = ADCPPB1OFFREF - ADCRESULTx Reset type: SYSRSn

Table 20-56. ADCPPB1CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	CONFIG	R/W	0h	ADC Post Processing Block 1 Configuration. This bit field defines which SOC/EOC/RESULT is assocatied with this post processing block. 0000 SOC0/EOC0/RESULT0 is associated with post processing block 1 0001 SOC1/EOC1/RESULT1 is associated with post processing block 1 0010 SOC2/EOC2/RESULT2 is associated with post processing block 1 0011 SOC3/EOC3/RESULT3 is associated with post processing block 1 0100 SOC4/EOC4/RESULT4 is associated with post processing block 1 0101 SOC5/EOC5/RESULT5 is associated with post processing block 1 0110 SOC6/EOC6/RESULT6 is associated with post processing block 1 0111 SOC7/EOC7/RESULT7 is associated with post processing block 1 1000 SOC8/EOC8/RESULT8 is associated with post processing block 1 1001 SOC9/EOC9/RESULT9 is associated with post processing block 1 1010 SOC10/EOC10/RESULT10 is associated with post processing block 1 1011 SOC11/EOC11/RESULT11 is associated with post processing block 1 1100 SOC12/EOC12/RESULT12 is associated with post processing block 1 1101 SOC13/EOC13/RESULT13 is associated with post processing block 1 1110 SOC14/EOC14/RESULT14 is associated with post processing block 1 1111 SOC15/EOC15/RESULT15 is associated with post processing block 1 Reset type: SYSRSn

20.15.2.42 ADCPPB1STAMP Register (Offset = 41h) [Reset = 0h]

ADCPPB1STAMP is shown in Figure 20-67 and described in Table 20-57.

Return to the Summary Table.

ADC PPB1 Sample Delay Time Stamp Register

Figure 20-67. ADCPPB1STAMP Register

15	14	13	12	11	10	9	8
RESERVED				DLYSTAMP			
R-0h				R-0h			
7	6	5	4	3	2	1	0
DLYSTAMP							
R-0h							

Table 20-57. ADCPPB1STAMP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DLYSTAMP	R	0h	ADC Post Processing Block 1 Delay Time Stamp. When an SOC starts sampling the value contained in REQSTAMP is subtracted from the value in ADCCOUNTER.FREECOUNT and loaded into this bit field, thereby giving the number of system clock cycles delay between the SOC trigger and the actual start of the sample. Reset type: SYSRSn

20.15.2.43 ADCPPB1OFFCAL Register (Offset = 42h) [Reset = 0h]

ADCPPB1OFFCAL is shown in Figure 20-68 and described in Table 20-58.

Return to the Summary Table.

ADC PPB1 Offset Calibration Register

Figure 20-68. ADCPPB1OFFCAL Register

15	14	13	12	11	10	9	8
RESERVED						OFFCAL	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
OFFCAL							
R/W-0h							

Table 20-58. ADCPPB1OFFCAL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	OFFCAL	R/W	0h	ADC Post Processing Block 1 Offset Correction. This bit field can be used to digitally remove any system level offset inherent in the ADCIN circuit. This 10-bit signed value is subtracted from the ADC output before being stored in the ADCRESULT register. 000h No change. The ADC output is stored directly into ADCRESULT. 001h ADC output - 1 is stored into ADCRESULT. 002h ADC output - 2 is stored into ADCRESULT. ... 200h ADC output + 512 is stored into ADCRESULT. ... 3FFh ADC output + 1 is stored into ADCRESULT. NOTE: In 16-bit mode, the subtraction will saturate at 0000h and FFFFh before being stored into the ADCRESULT register. In 12-bit mode, the subtraction will saturate at 0000h and 0FFFh before being stored into the ADCRESULT register. Note: in the case that multiple PPBs point to the same SOC, only the OFFCAL of the highest numbered PPB will be applied. Reset type: SYSRSn

20.15.2.44 ADCPPB1OFFREF Register (Offset = 43h) [Reset = 0h]

ADCPPB1OFFREF is shown in Figure 20-69 and described in Table 20-59.

Return to the Summary Table.

ADC PPB1 Offset Reference Register

Figure 20-69. ADCPPB1OFFREF Register

15	14	13	12	11	10	9	8
OFFREF							
R/W-0h							
7	6	5	4	3	2	1	0
OFFREF							
R/W-0h							

Table 20-59. ADCPPB1OFFREF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	OFFREF	R/W	0h	ADC Post Processing Block 1 Offset Correction. This bit field can be used to either calculate the feedback error or convert a unipolar signal to bipolar by subtracting a reference value. This 16-bit unsigned value is subtracted from the ADCRESULT register before being passed through an optional two's complement function and stored in the ADCPPB1RESULT register. This subtraction is not saturated. 0000h No change. The ADCRESULT value is passed on. 0001h ADCRESULT - 1 is passed on. 0002h ADCRESULT - 2 is passed on. ... 8000h ADCRESULT - 32,768 is passed on. ... FFFFh ADCRESULT - 65,535 is passed on. NOTE: In 12-bit mode the size of this register does not change from 16-bits. It is the user's responsibility to ensure that only a 12-bit value is written to this register when in 12-bit mode. Reset type: SYSRSn

20.15.2.45 ADCPPB1TRIPHI Register (Offset = 44h) [Reset = 0h]

ADCPPB1TRIPHI is shown in Figure 20-70 and described in Table 20-60.

Return to the Summary Table.

ADC PPB1 Trip High Register

Figure 20-70. ADCPPB1TRIPHI Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							HSIGN
R-0h							R/W-0h
15	14	13	12	11	10	9	8
LIMITHI							
R/W-0h							
7	6	5	4	3	2	1	0
LIMITHI							
R/W-0h							

Table 20-60. ADCPPB1TRIPHI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	HSIGN	R/W	0h	High Limit Sign Bit. This is the sign bit (17th bit) to the LIMITHI bit field when in 16-bit ADC mode. Reset type: SYSRSn
15-0	LIMITHI	R/W	0h	ADC Post Processing Block 1 Trip High Limit. This value sets the digital comparator trip high limit. In 16-bit mode all 17 bits will be compared against the 17 bits of the PPBRESULT bit field of the ADCPPB1RESULT register. In 12-bit mode bits 12:0 will be compared against bits 12:0 of the PPBRESULT bit field of the ADCPPB1RESULT register. Reset type: SYSRSn

20.15.2.46 ADCPPB1TRIPLO Register (Offset = 46h) [Reset = 0h]

ADCPPB1TRIPLO is shown in Figure 20-71 and described in Table 20-61.

Return to the Summary Table.

ADC PPB1 Trip Low/Trigger Time Stamp Register

Figure 20-71. ADCPPB1TRIPLO Register

31	30	29	28	27	26	25	24
REQSTAMP							
R-0h							
23	22	21	20	19	18	17	16
REQSTAMP				RESERVED			LSIGN
R-0h				R-0h			R/W-0h
15	14	13	12	11	10	9	8
LIMITLO							
R/W-0h							
7	6	5	4	3	2	1	0
LIMITLO							
R/W-0h							

Table 20-61. ADCPPB1TRIPLO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	REQSTAMP	R	0h	ADC Post Processing Block 1 Request Time Stamp. When a trigger sets the associated SOC flag in the ADCSOCFLG1 register the value of ADCCOUNTER.FREECOUNT is loaded into this bit field. Reset type: SYSRSn
19-17	RESERVED	R	0h	Reserved
16	LSIGN	R/W	0h	Low Limit Sign Bit. This is the sign bit (17th bit) to the LIMITLO bit field when in 16-bit ADC mode. Reset type: SYSRSn
15-0	LIMITLO	R/W	0h	ADC Post Processing Block 1 Trip Low Limit. This value sets the digital comparator trip low limit. In 16-bit mode all 17 bits will be compared against the 17 bits of the PPBRESULT bit field of the ADCPPB1RESULT register. In 12-bit mode bits 12:0 will be compared against bits 12:0 of the PPBRSULT bit field of the ADCPPB1RESULT register. Reset type: SYSRSn

20.15.2.47 ADCPPB2CONFIG Register (Offset = 48h) [Reset = 0h]

ADCPPB2CONFIG is shown in Figure 20-72 and described in Table 20-62.

Return to the Summary Table.

ADC PPB2 Config Register

Figure 20-72. ADCPPB2CONFIG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		CBCEN	TWOSCOMPEN	CONFIG			
R-0h		R/W-0h	R/W-0h	R/W-0h			

Table 20-62. ADCPPB2CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-6	RESERVED	R	0h	Reserved
5	CBCEN	R/W	0h	ADC Post Processing Block Cycle By Cycle Enable. When set, this bit enables the post conversion hardware processing circuit to automatically clear the ADCEVTSTAT on a conversion if the event condition is no longer present. Reset type: SYSRSn
4	TWOSCOMPEN	R/W	0h	ADC Post Processing Block 2 Two's Complement Enable. When set this bit enables the post conversion hardware processing circuit that performs a two's complement on the output of the offset/reference subtraction unit before storing the result in the ADCPPB2RESULT register. 0 ADCPPB2RESULT = ADCRESULTx - ADCPPB2OFFREF 1 ADCPPB2RESULT = ADCPPB2OFFREF - ADCRESULTx Reset type: SYSRSn

Table 20-62. ADCPPB2CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	CONFIG	R/W	0h	ADC Post Processing Block 2 Configuration. This bit field defines which SOC/EOC/RESULT is assocatied with this post processing block. 0000 SOC0/EOC0/RESULT0 is associated with post processing block 2 0001 SOC1/EOC1/RESULT1 is associated with post processing block 2 0010 SOC2/EOC2/RESULT2 is associated with post processing block 2 0011 SOC3/EOC3/RESULT3 is associated with post processing block 2 0100 SOC4/EOC4/RESULT4 is associated with post processing block 2 0101 SOC5/EOC5/RESULT5 is associated with post processing block 2 0110 SOC6/EOC6/RESULT6 is associated with post processing block 2 0111 SOC7/EOC7/RESULT7 is associated with post processing block 2 1000 SOC8/EOC8/RESULT8 is associated with post processing block 2 1001 SOC9/EOC9/RESULT9 is associated with post processing block 2 1010 SOC10/EOC10/RESULT10 is associated with post processing block 2 1011 SOC11/EOC11/RESULT11 is associated with post processing block 2 1100 SOC12/EOC12/RESULT12 is associated with post processing block 2 1101 SOC13/EOC13/RESULT13 is associated with post processing block 2 1110 SOC14/EOC14/RESULT14 is associated with post processing block 2 1111 SOC15/EOC15/RESULT15 is associated with post processing block 2 Reset type: SYSRSn

20.15.2.48 ADCPPB2STAMP Register (Offset = 49h) [Reset = 0h]

ADCPPB2STAMP is shown in Figure 20-73 and described in Table 20-63.

Return to the Summary Table.

ADC PPB2 Sample Delay Time Stamp Register

Figure 20-73. ADCPPB2STAMP Register

15	14	13	12	11	10	9	8
RESERVED				DLYSTAMP			
R-0h				R-0h			
7	6	5	4	3	2	1	0
DLYSTAMP							
R-0h							

Table 20-63. ADCPPB2STAMP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DLYSTAMP	R	0h	ADC Post Processing Block 2 Delay Time Stamp. When an SOC starts sampling the value contained in REQSTAMP is subtracted from the value in ADCCOUNTER.FREECOUNT and loaded into this bit field, thereby giving the number of system clock cycles delay between the SOC trigger and the actual start of the sample. Reset type: SYSRSn

20.15.2.49 ADCPPB2OFFCAL Register (Offset = 4Ah) [Reset = 0h]

ADCPPB2OFFCAL is shown in Figure 20-74 and described in Table 20-64.

Return to the Summary Table.

ADC PPB2 Offset Calibration Register

Figure 20-74. ADCPPB2OFFCAL Register

15	14	13	12	11	10	9	8
RESERVED						OFFCAL	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
OFFCAL							
R/W-0h							

Table 20-64. ADCPPB2OFFCAL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	OFFCAL	R/W	0h	ADC Post Processing Block 2 Offset Correction. This bit field can be used to digitally remove any system level offset inherent in the ADCIN circuit. This 10-bit signed value is subtracted from the ADC output before being stored in the ADCRESULT register. 000h No change. The ADC output is stored directly into ADCRESULT. 001h ADC output - 1 is stored into ADCRESULT. 002h ADC output - 2 is stored into ADCRESULT. ... 200h ADC output + 512 is stored into ADCRESULT. ... 3FFh ADC output + 1 is stored into ADCRESULT. NOTE: In 16-bit mode, the subtraction will saturate at 0000h and FFFFh before being stored into the ADCRESULT register. In 12-bit mode, the subtraction will saturate at 0000h and 0FFFh before being stored into the ADCRESULT register. Note: in the case that multiple PPBs point to the same SOC, only the OFFCAL of the highest numbered PPB will be applied. Reset type: SYSRSn

20.15.2.50 ADCPPB2OFFREF Register (Offset = 4Bh) [Reset = 0h]

ADCPPB2OFFREF is shown in Figure 20-75 and described in Table 20-65.

Return to the Summary Table.

ADC PPB2 Offset Reference Register

Figure 20-75. ADCPPB2OFFREF Register

15	14	13	12	11	10	9	8
OFFREF							
R/W-0h							
7	6	5	4	3	2	1	0
OFFREF							
R/W-0h							

Table 20-65. ADCPPB2OFFREF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	OFFREF	R/W	0h	ADC Post Processing Block 2 Offset Correction. This bit field can be used to either calculate the feedback error or convert a unipolar signal to bipolar by subtracting a reference value. This 16-bit unsigned value is subtracted from the ADCRESULT register before being passed through an optional two's complement function and stored in the ADCPPB2RESULT register. This subtraction is not saturated. 0000h No change. The ADCRESULT value is passed on. 0001h ADCRESULT - 1 is passed on. 0002h ADCRESULT - 2 is passed on. ... 8000h ADCRESULT - 32,768 is passed on. ... FFFFh ADCRESULT - 65,535 is passed on. NOTE: In 12-bit mode the size of this register does not change from 16-bits. It is the user's responsibility to ensure that only a 12-bit value is written to this register when in 12-bit mode. Reset type: SYSRSn

20.15.2.51 ADCPPB2TRIPHI Register (Offset = 4Ch) [Reset = 0h]

ADCPPB2TRIPHI is shown in Figure 20-76 and described in Table 20-66.

Return to the Summary Table.

ADC PPB2 Trip High Register

Figure 20-76. ADCPPB2TRIPHI Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							HSIGN
R-0h							R/W-0h
15	14	13	12	11	10	9	8
LIMITHI							
R/W-0h							
7	6	5	4	3	2	1	0
LIMITHI							
R/W-0h							

Table 20-66. ADCPPB2TRIPHI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	HSIGN	R/W	0h	High Limit Sign Bit. This is the sign bit (17th bit) to the LIMITHI bit field when in 16-bit ADC mode. Reset type: SYSRSn
15-0	LIMITHI	R/W	0h	ADC Post Processing Block 2 Trip High Limit. This value sets the digital comparator trip high limit. In 16-bit mode all 17 bits will be compared against the 17 bits of the PPBRESULT bit field of the ADCPPB2RESULT register. In 12-bit mode bits 12:0 will be compared against bits 12:0 of the PPBRESULT bit field of the ADCPPB2RESULT register. Reset type: SYSRSn

20.15.2.52 ADCPPB2TRIPLO Register (Offset = 4Eh) [Reset = 0h]

ADCPPB2TRIPLO is shown in Figure 20-77 and described in Table 20-67.

Return to the Summary Table.

ADC PPB2 Trip Low/Trigger Time Stamp Register

Figure 20-77. ADCPPB2TRIPLO Register

31	30	29	28	27	26	25	24
REQSTAMP							
R-0h							
23	22	21	20	19	18	17	16
REQSTAMP				RESERVED			LSIGN
R-0h				R-0h			R/W-0h
15	14	13	12	11	10	9	8
LIMITLO							
R/W-0h							
7	6	5	4	3	2	1	0
LIMITLO							
R/W-0h							

Table 20-67. ADCPPB2TRIPLO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	REQSTAMP	R	0h	ADC Post Processing Block 2 Request Time Stamp. When a trigger sets the associated SOC flag in the ADCSOCFLG1 register the value of ADCCOUNTER.FREECOUNT is loaded into this bit field. Reset type: SYSRSn
19-17	RESERVED	R	0h	Reserved
16	LSIGN	R/W	0h	Low Limit Sign Bit. This is the sign bit (17th bit) to the LIMITLO bit field when in 16-bit ADC mode. Reset type: SYSRSn
15-0	LIMITLO	R/W	0h	ADC Post Processing Block 2 Trip Low Limit. This value sets the digital comparator trip low limit. In 16-bit mode all 17 bits will be compared against the 17 bits of the PPBRESULT bit field of the ADCPPB2RESULT register. In 12-bit mode bits 12:0 will be compared against bits 12:0 of the PPBRSULT bit field of the ADCPPB2RESULT register. Reset type: SYSRSn

20.15.2.53 ADCPPB3CONFIG Register (Offset = 50h) [Reset = 0h]

ADCPPB3CONFIG is shown in Figure 20-78 and described in Table 20-68.

Return to the Summary Table.

ADC PPB3 Config Register

Figure 20-78. ADCPPB3CONFIG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		CBCEN	TWOSCOMPEN	CONFIG			
R-0h		R/W-0h	R/W-0h	R/W-0h			

Table 20-68. ADCPPB3CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-6	RESERVED	R	0h	Reserved
5	CBCEN	R/W	0h	ADC Post Processing Block Cycle By Cycle Enable. When set, this bit enables the post conversion hardware processing circuit to automatically clear the ADCEVTSTAT on a conversion if the event condition is no longer present. Reset type: SYSRSn
4	TWOSCOMPEN	R/W	0h	ADC Post Processing Block 3 Two's Complement Enable. When set this bit enables the post conversion hardware processing circuit that performs a two's complement on the output of the offset/reference subtraction unit before storing the result in the ADCPPB3RESULT register. 0 ADCPPB3RESULT = ADCRESULTx - ADCPPB3OFFREF 1 ADCPPB3RESULT = ADCPPB3OFFREF - ADCRESULTx Reset type: SYSRSn

Table 20-68. ADCPPB3CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	CONFIG	R/W	0h	ADC Post Processing Block 3 Configuration. This bit field defines which SOC/EOC/RESULT is assocatied with this post processing block. 0000 SOC0/EOC0/RESULT0 is associated with post processing block 3 0001 SOC1/EOC1/RESULT1 is associated with post processing block 3 0010 SOC2/EOC2/RESULT2 is associated with post processing block 3 0011 SOC3/EOC3/RESULT3 is associated with post processing block 3 0100 SOC4/EOC4/RESULT4 is associated with post processing block 3 0101 SOC5/EOC5/RESULT5 is associated with post processing block 3 0110 SOC6/EOC6/RESULT6 is associated with post processing block 3 0111 SOC7/EOC7/RESULT7 is associated with post processing block 3 1000 SOC8/EOC8/RESULT8 is associated with post processing block 3 1001 SOC9/EOC9/RESULT9 is associated with post processing block 3 1010 SOC10/EOC10/RESULT10 is associated with post processing block 3 1011 SOC11/EOC11/RESULT11 is associated with post processing block 3 1100 SOC12/EOC12/RESULT12 is associated with post processing block 3 1101 SOC13/EOC13/RESULT13 is associated with post processing block 3 1110 SOC14/EOC14/RESULT14 is associated with post processing block 3 1111 SOC15/EOC15/RESULT15 is associated with post processing block 3 Reset type: SYSRSn

20.15.2.54 ADCPPB3STAMP Register (Offset = 51h) [Reset = 0h]

ADCPPB3STAMP is shown in Figure 20-79 and described in Table 20-69.

Return to the Summary Table.

ADC PPB3 Sample Delay Time Stamp Register

Figure 20-79. ADCPPB3STAMP Register

15	14	13	12	11	10	9	8
RESERVED				DLYSTAMP			
R-0h				R-0h			
7	6	5	4	3	2	1	0
DLYSTAMP							
R-0h							

Table 20-69. ADCPPB3STAMP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DLYSTAMP	R	0h	ADC Post Processing Block 3 Delay Time Stamp. When an SOC starts sampling the value contained in REQSTAMP is subtracted from the value in ADCCOUNTER.FREECOUNT and loaded into this bit field, thereby giving the number of system clock cycles delay between the SOC trigger and the actual start of the sample. Reset type: SYSRSn

20.15.2.55 ADCPPB3OFFCAL Register (Offset = 52h) [Reset = 0h]

ADCPPB3OFFCAL is shown in Figure 20-80 and described in Table 20-70.

Return to the Summary Table.

ADC PPB3 Offset Calibration Register

Figure 20-80. ADCPPB3OFFCAL Register

15	14	13	12	11	10	9	8
RESERVED						OFFCAL	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
OFFCAL							
R/W-0h							

Table 20-70. ADCPPB3OFFCAL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	OFFCAL	R/W	0h	ADC Post Processing Block 3 Offset Correction. This bit field can be used to digitally remove any system level offset inherent in the ADCIN circuit. This 10-bit signed value is subtracted from the ADC output before being stored in the ADCRESULT register. 000h No change. The ADC output is stored directly into ADCRESULT. 001h ADC output - 1 is stored into ADCRESULT. 002h ADC output - 2 is stored into ADCRESULT. ... 200h ADC output + 512 is stored into ADCRESULT. ... 3FFh ADC output + 1 is stored into ADCRESULT. NOTE: In 16-bit mode, the subtraction will saturate at 0000h and FFFFh before being stored into the ADCRESULT register. In 12-bit mode, the subtraction will saturate at 0000h and 0FFFh before being stored into the ADCRESULT register. Note: in the case that multiple PPBs point to the same SOC, only the OFFCAL of the highest numbered PPB will be applied. Reset type: SYSRSn

20.15.2.56 ADCPPB3OFFREF Register (Offset = 53h) [Reset = 0h]

ADCPPB3OFFREF is shown in Figure 20-81 and described in Table 20-71.

Return to the Summary Table.

ADC PPB3 Offset Reference Register

Figure 20-81. ADCPPB3OFFREF Register

15	14	13	12	11	10	9	8
OFFREF							
R/W-0h							
7	6	5	4	3	2	1	0
OFFREF							
R/W-0h							

Table 20-71. ADCPPB3OFFREF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	OFFREF	R/W	0h	ADC Post Processing Block 3 Offset Correction. This bit field can be used to either calculate the feedback error or convert a unipolar signal to bipolar by subtracting a reference value. This 16-bit unsigned value is subtracted from the ADCRESULT register before being passed through an optional two's complement function and stored in the ADCPPB3RESULT register. This subtraction is not saturated. 0000h No change. The ADCRESULT value is passed on. 0001h ADCRESULT - 1 is passed on. 0002h ADCRESULT - 2 is passed on. ... 8000h ADCRESULT - 32,768 is passed on. ... FFFFh ADCRESULT - 65,535 is passed on. NOTE: In 12-bit mode the size of this register does not change from 16-bits. It is the user's responsibility to ensure that only a 12-bit value is written to this register when in 12-bit mode. Reset type: SYSRSn

20.15.2.57 ADCPPB3TRIPHI Register (Offset = 54h) [Reset = 0h]

ADCPPB3TRIPHI is shown in Figure 20-82 and described in Table 20-72.

Return to the Summary Table.

ADC PPB3 Trip High Register

Figure 20-82. ADCPPB3TRIPHI Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							HSIGN
R-0h							R/W-0h
15	14	13	12	11	10	9	8
LIMITHI							
R/W-0h							
7	6	5	4	3	2	1	0
LIMITHI							
R/W-0h							

Table 20-72. ADCPPB3TRIPHI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	HSIGN	R/W	0h	High Limit Sign Bit. This is the sign bit (17th bit) to the LIMITHI bit field when in 16-bit ADC mode. Reset type: SYSRSn
15-0	LIMITHI	R/W	0h	ADC Post Processing Block 3 Trip High Limit. This value sets the digital comparator trip high limit. In 16-bit mode all 17 bits will be compared against the 17 bits of the PPBRESULT bit field of the ADCPPB3RESULT register. In 12-bit mode bits 12:0 will be compared against bits 12:0 of the PPBRESULT bit field of the ADCPPB3RESULT register. Reset type: SYSRSn

20.15.2.58 ADCPPB3TRIPLO Register (Offset = 56h) [Reset = 0h]

ADCPPB3TRIPLO is shown in Figure 20-83 and described in Table 20-73.

Return to the Summary Table.

ADC PPB3 Trip Low/Trigger Time Stamp Register

Figure 20-83. ADCPPB3TRIPLO Register

31	30	29	28	27	26	25	24
REQSTAMP							
R-0h							
23	22	21	20	19	18	17	16
REQSTAMP				RESERVED			LSIGN
R-0h				R-0h			R/W-0h
15	14	13	12	11	10	9	8
LIMITLO							
R/W-0h							
7	6	5	4	3	2	1	0
LIMITLO							
R/W-0h							

Table 20-73. ADCPPB3TRIPLO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	REQSTAMP	R	0h	ADC Post Processing Block 3 Request Time Stamp. When a trigger sets the associated SOC flag in the ADCSOCFLG1 register the value of ADCCOUNTER.FREECOUNT is loaded into this bit field. Reset type: SYSRSn
19-17	RESERVED	R	0h	Reserved
16	LSIGN	R/W	0h	Low Limit Sign Bit. This is the sign bit (17th bit) to the LIMITLO bit field when in 16-bit ADC mode. Reset type: SYSRSn
15-0	LIMITLO	R/W	0h	ADC Post Processing Block 3 Trip Low Limit. This value sets the digital comparator trip low limit. In 16-bit mode all 17 bits will be compared against the 17 bits of the PPBRESULT bit field of the ADCPPB3RESULT register. In 12-bit mode bits 12:0 will be compared against bits 12:0 of the PPBRSULT bit field of the ADCPPB3RESULT register. Reset type: SYSRSn

20.15.2.59 ADCPPB4CONFIG Register (Offset = 58h) [Reset = 0h]

ADCPPB4CONFIG is shown in Figure 20-84 and described in Table 20-74.

Return to the Summary Table.

ADC PPB4 Config Register

Figure 20-84. ADCPPB4CONFIG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		CBCEN	TWOSCOMPEN	CONFIG			
R-0h		R/W-0h	R/W-0h	R/W-0h			

Table 20-74. ADCPPB4CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-6	RESERVED	R	0h	Reserved
5	CBCEN	R/W	0h	ADC Post Processing Block Cycle By Cycle Enable. When set, this bit enables the post conversion hardware processing circuit to automatically clear the ADCEVTSTAT on a conversion if the event condition is no longer present. Reset type: SYSRSn
4	TWOSCOMPEN	R/W	0h	ADC Post Processing Block 4 Two's Complement Enable. When set this bit enables the post conversion hardware processing circuit that performs a two's complement on the output of the offset/reference subtraction unit before storing the result in the ADCPPB4RESULT register. 0 ADCPPB4RESULT = ADCRESULTx - ADCPPB4OFFREF 1 ADCPPB4RESULT = ADCPPB4OFFREF - ADCRESULTx Reset type: SYSRSn

Table 20-74. ADCPPB4CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	CONFIG	R/W	0h	ADC Post Processing Block 4 Configuration. This bit field defines which SOC/EOC/RESULT is assocatied with this post processing block. 0000 SOC0/EOC0/RESULT0 is associated with post processing block 4 0001 SOC1/EOC1/RESULT1 is associated with post processing block 4 0010 SOC2/EOC2/RESULT2 is associated with post processing block 4 0011 SOC3/EOC3/RESULT3 is associated with post processing block 4 0100 SOC4/EOC4/RESULT4 is associated with post processing block 4 0101 SOC5/EOC5/RESULT5 is associated with post processing block 4 0110 SOC6/EOC6/RESULT6 is associated with post processing block 4 0111 SOC7/EOC7/RESULT7 is associated with post processing block 4 1000 SOC8/EOC8/RESULT8 is associated with post processing block 4 1001 SOC9/EOC9/RESULT9 is associated with post processing block 4 1010 SOC10/EOC10/RESULT10 is associated with post processing block 4 1011 SOC11/EOC11/RESULT11 is associated with post processing block 4 1100 SOC12/EOC12/RESULT12 is associated with post processing block 4 1101 SOC13/EOC13/RESULT13 is associated with post processing block 4 1110 SOC14/EOC14/RESULT14 is associated with post processing block 4 1111 SOC15/EOC15/RESULT15 is associated with post processing block 4 Reset type: SYSRSn

20.15.2.60 ADCPPB4STAMP Register (Offset = 59h) [Reset = 0h]

ADCPPB4STAMP is shown in Figure 20-85 and described in Table 20-75.

Return to the Summary Table.

ADC PPB4 Sample Delay Time Stamp Register

Figure 20-85. ADCPPB4STAMP Register

15	14	13	12	11	10	9	8
RESERVED				DLYSTAMP			
R-0h				R-0h			
7	6	5	4	3	2	1	0
DLYSTAMP							
R-0h							

Table 20-75. ADCPPB4STAMP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DLYSTAMP	R	0h	ADC Post Processing Block 4 Delay Time Stamp. When an SOC starts sampling the value contained in REQSTAMP is subtracted from the value in ADCCOUNTER.FREECOUNT and loaded into this bit field, thereby giving the number of system clock cycles delay between the SOC trigger and the actual start of the sample. Reset type: SYSRSn

20.15.2.61 ADCPPB4OFFCAL Register (Offset = 5Ah) [Reset = 0h]

ADCPPB4OFFCAL is shown in Figure 20-86 and described in Table 20-76.

Return to the Summary Table.

ADC PPB4 Offset Calibration Register

Figure 20-86. ADCPPB4OFFCAL Register

15	14	13	12	11	10	9	8
RESERVED						OFFCAL	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
OFFCAL							
R/W-0h							

Table 20-76. ADCPPB4OFFCAL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	OFFCAL	R/W	0h	ADC Post Processing Block 4 Offset Correction. This bit field can be used to digitally remove any system level offset inherent in the ADCIN circuit. This 10-bit signed value is subtracted from the ADC output before being stored in the ADCRESULT register. 000h No change. The ADC output is stored directly into ADCRESULT. 001h ADC output - 1 is stored into ADCRESULT. 002h ADC output - 2 is stored into ADCRESULT. ... 200h ADC output + 512 is stored into ADCRESULT. ... 3FFh ADC output + 1 is stored into ADCRESULT. NOTE: In 16-bit mode, the subtraction will saturate at 0000h and FFFFh before being stored into the ADCRESULT register. In 12-bit mode, the subtraction will saturate at 0000h and 0FFFh before being stored into the ADCRESULT register. Note: in the case that multiple PPBs point to the same SOC, only the OFFCAL of the highest numbered PPB will be applied. Reset type: SYSRSn

20.15.2.62 ADCPPB4OFFREF Register (Offset = 5Bh) [Reset = 0h]

ADCPPB4OFFREF is shown in Figure 20-87 and described in Table 20-77.

Return to the Summary Table.

ADC PPB4 Offset Reference Register

Figure 20-87. ADCPPB4OFFREF Register

15	14	13	12	11	10	9	8
OFFREF							
R/W-0h							
7	6	5	4	3	2	1	0
OFFREF							
R/W-0h							

Table 20-77. ADCPPB4OFFREF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	OFFREF	R/W	0h	ADC Post Processing Block 4 Offset Correction. This bit field can be used to either calculate the feedback error or convert a unipolar signal to bipolar by subtracting a reference value. This 16-bit unsigned value is subtracted from the ADCRESULT register before being passed through an optional two's complement function and stored in the ADCPPB4RESULT register. This subtraction is not saturated. 0000h No change. The ADCRESULT value is passed on. 0001h ADCRESULT - 1 is passed on. 0002h ADCRESULT - 2 is passed on. ... 8000h ADCRESULT - 32,768 is passed on. ... FFFFh ADCRESULT - 65,535 is passed on. NOTE: In 12-bit mode the size of this register does not change from 16-bits. It is the user's responsibility to ensure that only a 12-bit value is written to this register when in 12-bit mode. Reset type: SYSRSn

20.15.2.63 ADCPPB4TRIPHI Register (Offset = 5Ch) [Reset = 0h]

ADCPPB4TRIPHI is shown in Figure 20-88 and described in Table 20-78.

Return to the Summary Table.

ADC PPB4 Trip High Register

Figure 20-88. ADCPPB4TRIPHI Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							HSIGN
R-0h							R/W-0h
15	14	13	12	11	10	9	8
LIMITHI							
R/W-0h							
7	6	5	4	3	2	1	0
LIMITHI							
R/W-0h							

Table 20-78. ADCPPB4TRIPHI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	HSIGN	R/W	0h	High Limit Sign Bit. This is the sign bit (17th bit) to the LIMITHI bit field when in 16-bit ADC mode. Reset type: SYSRSn
15-0	LIMITHI	R/W	0h	ADC Post Processing Block 4 Trip High Limit. This value sets the digital comparator trip high limit. In 16-bit mode all 17 bits will be compared against the 17 bits of the PPBRESULT bit field of the ADCPPB4RESULT register. In 12-bit mode bits 12:0 will be compared against bits 12:0 of the PPBRESULT bit field of the ADCPPB4RESULT register. Reset type: SYSRSn

20.15.2.64 ADCPPB4TRIPLO Register (Offset = 5Eh) [Reset = 0h]

ADCPPB4TRIPLO is shown in Figure 20-89 and described in Table 20-79.

Return to the Summary Table.

ADC PPB4 Trip Low/Trigger Time Stamp Register

Figure 20-89. ADCPPB4TRIPLO Register

31	30	29	28	27	26	25	24
REQSTAMP							
R-0h							
23	22	21	20	19	18	17	16
REQSTAMP				RESERVED			LSIGN
R-0h				R-0h			R/W-0h
15	14	13	12	11	10	9	8
LIMITLO							
R/W-0h							
7	6	5	4	3	2	1	0
LIMITLO							
R/W-0h							

Table 20-79. ADCPPB4TRIPLO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	REQSTAMP	R	0h	ADC Post Processing Block 4 Request Time Stamp. When a trigger sets the associated SOC flag in the ADCSOCFLG1 register the value of ADCCOUNTER.FREECOUNT is loaded into this bit field. Reset type: SYSRSn
19-17	RESERVED	R	0h	Reserved
16	LSIGN	R/W	0h	Low Limit Sign Bit. This is the sign bit (17th bit) to the LIMITLO bit field when in 16-bit ADC mode. Reset type: SYSRSn
15-0	LIMITLO	R/W	0h	ADC Post Processing Block 4 Trip Low Limit. This value sets the digital comparator trip low limit. In 16-bit mode all 17 bits will be compared against the 17 bits of the PPBRESULT bit field of the ADCPPB4RESULT register. In 12-bit mode bits 12:0 will be compared against bits 12:0 of the PPBRSULT bit field of the ADCPPB4RESULT register. Reset type: SYSRSn

20.15.2.65 ADCINTCYCLE Register (Offset = 6Fh) [Reset = 0h]

ADCINTCYCLE is shown in Figure 20-90 and described in Table 20-80.

Return to the Summary Table.

ADC Early Interrupt Generation Cycle

Figure 20-90. ADCINTCYCLE Register

15	14	13	12	11	10	9	8
DELAY							
R/W-0h							
7	6	5	4	3	2	1	0
DELAY							
R/W-0h							

Table 20-80. ADCINTCYCLE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DELAY	R/W	0h	ADC Early Interrupt Generation Cycle Delay: Defines the delay from the fall edge of ADCSOC in terms of system clock cycles, for the interrupt to be generated. Reset type: SYSRSn

20.15.2.66 ADCINLTRIM1 Register (Offset = 70h) [Reset = X]

ADCINLTRIM1 is shown in Figure 20-91 and described in Table 20-81.

Return to the Summary Table.

ADC Linearity Trim 1 Register

Figure 20-91. ADCINLTRIM1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INLTRIM31TO0																															
R/W-X																															

Table 20-81. ADCINLTRIM1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INLTRIM31TO0	R/W	X	ADC Linearity Trim Bits 31-0. This register should not be modified unless specifically indicated by TI Errata or other documentation. Modifying the contents of this register could cause this module to operate outside of datasheet specifications. Reset type: SYSRSn

20.15.2.67 ADCINLTRIM2 Register (Offset = 72h) [Reset = X]

ADCINLTRIM2 is shown in Figure 20-92 and described in Table 20-82.

Return to the Summary Table.

ADC Linearity Trim 2 Register

Figure 20-92. ADCINLTRIM2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INLTRIM63TO32																															
R/W-X																															

Table 20-82. ADCINLTRIM2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INLTRIM63TO32	R/W	X	ADC Linearity Trim Bits 63-32. This register should not be modified unless specifically indicated by TI Errata or other documentation. Modifying the contents of this register could cause this module to operate outside of datasheet specifications. Reset type: SYSRSn

20.15.2.68 ADCINLTRIM3 Register (Offset = 74h) [Reset = X]

ADCINLTRIM3 is shown in Figure 20-93 and described in Table 20-83.

Return to the Summary Table.

ADC Linearity Trim 3 Register

Figure 20-93. ADCINLTRIM3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INLTRIM95TO64																															
R/W-X																															

Table 20-83. ADCINLTRIM3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INLTRIM95TO64	R/W	X	ADC Linearity Trim Bits 95-64. This register should not be modified unless specifically indicated by TI Errata or other documentation. Modifying the contents of this register could cause this module to operate outside of datasheet specifications. Reset type: SYSRSn

20.15.2.69 ADCINLTRIM4 Register (Offset = 76h) [Reset = X]

ADCINLTRIM4 is shown in Figure 20-94 and described in Table 20-84.

Return to the Summary Table.

ADC Linearity Trim 4 Register

Figure 20-94. ADCINLTRIM4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INLTRIM127TO96																															
R/W-X																															

Table 20-84. ADCINLTRIM4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INLTRIM127TO96	R/W	X	ADC Linearity Trim Bits 127-96. This register should not be modified unless specifically indicated by TI Errata or other documentation. Modifying the contents of this register could cause this module to operate outside of datasheet specifications. Reset type: SYSRSn

20.15.2.70 ADCINLTRIM5 Register (Offset = 78h) [Reset = X]

ADCINLTRIM5 is shown in Figure 20-95 and described in Table 20-85.

Return to the Summary Table.

ADC Linearity Trim 5 Register

Figure 20-95. ADCINLTRIM5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INLTRIM159TO128																															
R/W-X																															

Table 20-85. ADCINLTRIM5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INLTRIM159TO128	R/W	X	ADC Linearity Trim Bits 159-128. This register should not be modified unless specifically indicated by TI Errata or other documentation. Modifying the contents of this register could cause this module to operate outside of datasheet specifications. Reset type: SYSRSn

20.15.2.71 ADCINLTRIM6 Register (Offset = 7Ah) [Reset = X]

ADCINLTRIM6 is shown in Figure 20-96 and described in Table 20-86.

Return to the Summary Table.

ADC Linearity Trim 6 Register

Figure 20-96. ADCINLTRIM6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INLTRIM191TO160																															
R/W-X																															

Table 20-86. ADCINLTRIM6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INLTRIM191TO160	R/W	X	ADC Linearity Trim Bits 191-160. This register should not be modified unless specifically indicated by TI Errata or other documentation. Modifying the contents of this register could cause this module to operate outside of datasheet specifications. Reset type: SYSRSn

20.15.3 ADC_RESULT_REGS Registers

Table 20-87 lists the memory-mapped registers for the ADC_RESULT_REGS registers. All register offset addresses not listed in Table 20-87 should be considered as reserved locations and the register contents should not be modified.

Table 20-87. ADC_RESULT_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	ADCRESULT0	ADC Result 0 Register		Go
1h	ADCRESULT1	ADC Result 1 Register		Go
2h	ADCRESULT2	ADC Result 2 Register		Go
3h	ADCRESULT3	ADC Result 3 Register		Go
4h	ADCRESULT4	ADC Result 4 Register		Go
5h	ADCRESULT5	ADC Result 5 Register		Go
6h	ADCRESULT6	ADC Result 6 Register		Go
7h	ADCRESULT7	ADC Result 7 Register		Go
8h	ADCRESULT8	ADC Result 8 Register		Go
9h	ADCRESULT9	ADC Result 9 Register		Go
Ah	ADCRESULT10	ADC Result 10 Register		Go
Bh	ADCRESULT11	ADC Result 11 Register		Go
Ch	ADCRESULT12	ADC Result 12 Register		Go
Dh	ADCRESULT13	ADC Result 13 Register		Go
Eh	ADCRESULT14	ADC Result 14 Register		Go
Fh	ADCRESULT15	ADC Result 15 Register		Go
10h	ADCPPB1RESULT	ADC Post Processing Block 1 Result Register		Go
12h	ADCPPB2RESULT	ADC Post Processing Block 2 Result Register		Go
14h	ADCPPB3RESULT	ADC Post Processing Block 3 Result Register		Go
16h	ADCPPB4RESULT	ADC Post Processing Block 4 Result Register		Go

Complex bit access types are encoded to fit into small table cells. Table 20-88 shows the codes that are used for access types in this section.

Table 20-88. ADC_RESULT_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 20-88. ADC_RESULT_REGS Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

20.15.3.1 ADCRESULT0 Register (Offset = 0h) [Reset = 0h]

ADCRESULT0 is shown in Figure 20-97 and described in Table 20-89.

Return to the Summary Table.

ADC Result 0 Register

Figure 20-97. ADCRESULT0 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 20-89. ADCRESULT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 0 16-bit ADC result. After the ADC completes a conversion of SOC0, the digital result is placed in this bit field. Reset type: SYSRSn

20.15.3.2 ADCRESULT1 Register (Offset = 1h) [Reset = 0h]

ADCRESULT1 is shown in Figure 20-98 and described in Table 20-90.

Return to the Summary Table.

ADC Result 1 Register

Figure 20-98. ADCRESULT1 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 20-90. ADCRESULT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 1 16-bit ADC result. After the ADC completes a conversion of SOC1, the digital result is placed in this bit field. Reset type: SYSRSn

20.15.3.3 ADCRESULT2 Register (Offset = 2h) [Reset = 0h]

ADCRESULT2 is shown in Figure 20-99 and described in Table 20-91.

Return to the Summary Table.

ADC Result 2 Register

Figure 20-99. ADCRESULT2 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 20-91. ADCRESULT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 2 16-bit ADC result. After the ADC completes a conversion of SOC2, the digital result is placed in this bit field. Reset type: SYSRSn

20.15.3.4 ADCRESULT3 Register (Offset = 3h) [Reset = 0h]

ADCRESULT3 is shown in Figure 20-100 and described in Table 20-92.

Return to the Summary Table.

ADC Result 3 Register

Figure 20-100. ADCRESULT3 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 20-92. ADCRESULT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 3 16-bit ADC result. After the ADC completes a conversion of SOC3, the digital result is placed in this bit field. Reset type: SYSRSn

20.15.3.5 ADCRESULT4 Register (Offset = 4h) [Reset = 0h]

ADCRESULT4 is shown in Figure 20-101 and described in Table 20-93.

Return to the Summary Table.

ADC Result 4 Register

Figure 20-101. ADCRESULT4 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 20-93. ADCRESULT4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 4 16-bit ADC result. After the ADC completes a conversion of SOC4, the digital result is placed in this bit field. Reset type: SYSRSn

20.15.3.6 ADCRESULT5 Register (Offset = 5h) [Reset = 0h]

ADCRESULT5 is shown in Figure 20-102 and described in Table 20-94.

Return to the Summary Table.

ADC Result 5 Register

Figure 20-102. ADCRESULT5 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 20-94. ADCRESULT5 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 5 16-bit ADC result. After the ADC completes a conversion of SOC5, the digital result is placed in this bit field. Reset type: SYSRSn

20.15.3.7 ADCRESULT6 Register (Offset = 6h) [Reset = 0h]

ADCRESULT6 is shown in Figure 20-103 and described in Table 20-95.

Return to the Summary Table.

ADC Result 6 Register

Figure 20-103. ADCRESULT6 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 20-95. ADCRESULT6 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 6 16-bit ADC result. After the ADC completes a conversion of SOC6, the digital result is placed in this bit field. Reset type: SYSRSn

20.15.3.8 ADCRESULT7 Register (Offset = 7h) [Reset = 0h]

ADCRESULT7 is shown in Figure 20-104 and described in Table 20-96.

Return to the Summary Table.

ADC Result 7 Register

Figure 20-104. ADCRESULT7 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 20-96. ADCRESULT7 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 7 16-bit ADC result. After the ADC completes a conversion of SOC7, the digital result is placed in this bit field. Reset type: SYSRSn

20.15.3.9 ADCRESULT8 Register (Offset = 8h) [Reset = 0h]

ADCRESULT8 is shown in Figure 20-105 and described in Table 20-97.

Return to the Summary Table.

ADC Result 8 Register

Figure 20-105. ADCRESULT8 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 20-97. ADCRESULT8 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 8 16-bit ADC result. After the ADC completes a conversion of SOC8, the digital result is placed in this bit field. Reset type: SYSRSn

20.15.3.10 ADCRESULT9 Register (Offset = 9h) [Reset = 0h]

ADCRESULT9 is shown in Figure 20-106 and described in Table 20-98.

Return to the Summary Table.

ADC Result 9 Register

Figure 20-106. ADCRESULT9 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 20-98. ADCRESULT9 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 9 16-bit ADC result. After the ADC completes a conversion of SOC9, the digital result is placed in this bit field. Reset type: SYSRSn

20.15.3.11 ADCRESULT10 Register (Offset = Ah) [Reset = 0h]

ADCRESULT10 is shown in Figure 20-107 and described in Table 20-99.

Return to the Summary Table.

ADC Result 10 Register

Figure 20-107. ADCRESULT10 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 20-99. ADCRESULT10 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 10 16-bit ADC result. After the ADC completes a conversion of SOC10, the digital result is placed in this bit field. Reset type: SYSRSn

20.15.3.12 ADCRESULT11 Register (Offset = Bh) [Reset = 0h]

ADCRESULT11 is shown in Figure 20-108 and described in Table 20-100.

Return to the Summary Table.

ADC Result 11 Register

Figure 20-108. ADCRESULT11 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 20-100. ADCRESULT11 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 11 16-bit ADC result. After the ADC completes a conversion of SOC11, the digital result is placed in this bit field. Reset type: SYSRSn

20.15.3.13 ADCRESULT12 Register (Offset = Ch) [Reset = 0h]

ADCRESULT12 is shown in Figure 20-109 and described in Table 20-101.

Return to the Summary Table.

ADC Result 12 Register

Figure 20-109. ADCRESULT12 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 20-101. ADCRESULT12 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 12 16-bit ADC result. After the ADC completes a conversion of SOC12, the digital result is placed in this bit field. Reset type: SYSRSn

20.15.3.14 ADCRESULT13 Register (Offset = Dh) [Reset = 0h]

ADCRESULT13 is shown in Figure 20-110 and described in Table 20-102.

Return to the Summary Table.

ADC Result 13 Register

Figure 20-110. ADCRESULT13 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 20-102. ADCRESULT13 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 13 16-bit ADC result. After the ADC completes a conversion of SOC13, the digital result is placed in this bit field. Reset type: SYSRSn

20.15.3.15 ADCRESULT14 Register (Offset = Eh) [Reset = 0h]

ADCRESULT14 is shown in Figure 20-111 and described in Table 20-103.

Return to the Summary Table.

ADC Result 14 Register

Figure 20-111. ADCRESULT14 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 20-103. ADCRESULT14 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 14 16-bit ADC result. After the ADC completes a conversion of SOC14, the digital result is placed in this bit field. Reset type: SYSRSn

20.15.3.16 ADCRESULT15 Register (Offset = Fh) [Reset = 0h]

ADCRESULT15 is shown in Figure 20-112 and described in Table 20-104.

Return to the Summary Table.

ADC Result 15 Register

Figure 20-112. ADCRESULT15 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 20-104. ADCRESULT15 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 15 16-bit ADC result. After the ADC completes a conversion of SOC15, the digital result is placed in this bit field. Reset type: SYSRSn

20.15.3.17 ADCPPB1RESULT Register (Offset = 10h) [Reset = 0h]

ADCPPB1RESULT is shown in Figure 20-113 and described in Table 20-105.

Return to the Summary Table.

ADC Post Processing Block 1 Result Register

Figure 20-113. ADCPPB1RESULT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIGN																PPBRESULT															
R-0h																R-0h															

Table 20-105. ADCPPB1RESULT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	SIGN	R	0h	Sign Extended Bits. These bits reflect the same value as bit 16. NOTE: If the conversion associated with this Post Processing Block is a 12-bit conversion, the SIGN bits extend down to bit 12, and all reflect the same value as bit 12. Reset type: SYSRSn
15-0	PPBRESULT	R	0h	ADC Post Processing Block Result 1 The result of the offset/reference subtraction post conversion processing is stored in this register. This result is available 1 SYSCLK cycle after the associated ADCRESULT is available. If ADCINTFLG is polled to determine when to read the PPBRESULT, it may be necessary to add a NOP instruction to ensure that the updated post conversion processing result has posted to the register. NOTE: If the conversion associated with this Post Processing Block is a 12-bit conversion, the PPBRESULT bits are limited to bits 12:0. Reset type: SYSRSn

20.15.3.18 ADCPPB2RESULT Register (Offset = 12h) [Reset = 0h]

ADCPPB2RESULT is shown in Figure 20-114 and described in Table 20-106.

Return to the Summary Table.

ADC Post Processing Block 2 Result Register

Figure 20-114. ADCPPB2RESULT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIGN																PPBRESULT															
R-0h																R-0h															

Table 20-106. ADCPPB2RESULT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	SIGN	R	0h	Sign Extended Bits. These bits reflect the same value as bit 16. NOTE: If the conversion associated with this Post Processing Block is a 12-bit conversion, the SIGN bits extend down to bit 12, and all reflect the same value as bit 12. Reset type: SYSRSn
15-0	PPBRESULT	R	0h	ADC Post Processing Block Result 2 The result of the offset/reference subtraction post conversion processing is stored in this register. This result is available 1 SYSCLK cycle after the associated ADCRESULT is available. If ADCINTFLG is polled to determine when to read the PPBRESULT, it may be necessary to add a NOP instruction to ensure that the updated post conversion processing result has posted to the register. NOTE: If the conversion associated with this Post Processing Block is a 12-bit conversion, the PPBRESULT bits are limited to bits 12:0. Reset type: SYSRSn

20.15.3.19 ADCPPB3RESULT Register (Offset = 14h) [Reset = 0h]

ADCPPB3RESULT is shown in Figure 20-115 and described in Table 20-107.

Return to the Summary Table.

ADC Post Processing Block 3 Result Register

Figure 20-115. ADCPPB3RESULT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIGN																PPBRESULT															
R-0h																R-0h															

Table 20-107. ADCPPB3RESULT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	SIGN	R	0h	Sign Extended Bits. These bits reflect the same value as bit 16. NOTE: If the conversion associated with this Post Processing Block is a 12-bit conversion, the SIGN bits extend down to bit 12, and all reflect the same value as bit 12. Reset type: SYSRSn
15-0	PPBRESULT	R	0h	ADC Post Processing Block Result 3 The result of the offset/reference subtraction post conversion processing is stored in this register. This result is available 1 SYSCLK cycle after the associated ADCRESULT is available. If ADCINTFLG is polled to determine when to read the PPBRESULT, it may be necessary to add a NOP instruction to ensure that the updated post conversion processing result has posted to the register. NOTE: If the conversion associated with this Post Processing Block is a 12-bit conversion, the PPBRESULT bits are limited to bits 12:0. Reset type: SYSRSn

20.15.3.20 ADCPPB4RESULT Register (Offset = 16h) [Reset = 0h]

ADCPPB4RESULT is shown in Figure 20-116 and described in Table 20-108.

Return to the Summary Table.

ADC Post Processing Block 4 Result Register

Figure 20-116. ADCPPB4RESULT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIGN																PPBRESULT															
R-0h																R-0h															

Table 20-108. ADCPPB4RESULT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	SIGN	R	0h	Sign Extended Bits. These bits reflect the same value as bit 16. NOTE: If the conversion associated with this Post Processing Block is a 12-bit conversion, the SIGN bits extend down to bit 12, and all reflect the same value as bit 12. Reset type: SYSRSn
15-0	PPBRESULT	R	0h	ADC Post Processing Block Result 4 The result of the offset/reference subtraction post conversion processing is stored in this register. This result is available 1 SYSCLK cycle after the associated ADCRESULT is available. If ADCINTFLG is polled to determine when to read the PPBRESULT, it may be necessary to add a NOP instruction to ensure that the updated post conversion processing result has posted to the register. NOTE: If the conversion associated with this Post Processing Block is a 12-bit conversion, the PPBRESULT bits are limited to bits 12:0. Reset type: SYSRSn

20.15.4 ADC Registers to Driverlib Functions

Table 20-109. ADC Registers to Driverlib Functions

File	Driverlib Function
ADCCTL1	
adc.h	ADC_setInterruptPulseMode
adc.h	ADC_enableConverter
adc.h	ADC_disableConverter
adc.h	ADC_isBusy
ADCCTL2	
adc.c	ADC_setMode
adc.c	ADC_setINLTrim
adc.c	ADC_setOffsetTrim
adc.h	ADC_setPrescaler
ADCBURSTCTL	
adc.h	ADC_setBurstModeConfig
adc.h	ADC_enableBurstMode
adc.h	ADC_disableBurstMode
ADCINTFLG	
adc.h	ADC_getInterruptStatus
adc.h	ADC_clearInterruptStatus
ADCINTFLGCLR	
adc.h	ADC_clearInterruptStatus
ADCINTOVF	

Table 20-109. ADC Registers to Driverlib Functions (continued)

File	Driverlib Function
adc.h	ADC_getInterruptOverflowStatus
adc.h	ADC_clearInterruptOverflowStatus
ADCINTOVFCLR	
adc.h	ADC_clearInterruptOverflowStatus
ADCINTSEL1N2	
adc.h	ADC_enableInterrupt
adc.h	ADC_disableInterrupt
adc.h	ADC_setInterruptSource
adc.h	ADC_enableContinuousMode
adc.h	ADC_disableContinuousMode
ADCINTSEL3N4	
-	See INTSEL1N2
ADCSOCPRICTL	
adc.h	ADC_setSOCPriority
ADCINTSOCSEL1	
adc.h	ADC_setInterruptSOCTrigger
ADCINTSOCSEL2	
-	See INTSOCSEL1
ADCSOCFLG1	
-	
ADCSOCFRC1	
adc.h	ADC_forceSOC
adc.h	ADC_forceMultipleSOC
ADCSOCOVF1	
-	
ADCSOCOVFCLR1	
-	
ADCSOC0CTL	
adc.h	ADC_setupSOC
ADCSOC1CTL	
-	See SOC0CTL
ADCSOC2CTL	
-	See SOC0CTL
ADCSOC3CTL	
-	See SOC0CTL
ADCSOC4CTL	
-	See SOC0CTL
ADCSOC5CTL	
-	See SOC0CTL
ADCSOC6CTL	
-	See SOC0CTL
ADCSOC7CTL	
-	See SOC0CTL
ADCSOC8CTL	
-	See SOC0CTL

Table 20-109. ADC Registers to Driverlib Functions (continued)

File	Driverlib Function
ADCSOC9CTL	
-	See SOC0CTL
ADCSOC10CTL	
-	See SOC0CTL
ADCSOC11CTL	
-	See SOC0CTL
ADCSOC12CTL	
-	See SOC0CTL
ADCSOC13CTL	
-	See SOC0CTL
ADCSOC14CTL	
-	See SOC0CTL
ADCSOC15CTL	
-	See SOC0CTL
ADCEVTSTAT	
adc.h	ADC_getPPBEventStatus
ADCEVTCLR	
adc.h	ADC_clearPPBEventStatus
ADCEVTSEL	
adc.h	ADC_enablePPBEvent
adc.h	ADC_disablePPBEvent
ADCEVTINTSEL	
adc.h	ADC_enablePPBEventInterrupt
adc.h	ADC_disablePPBEventInterrupt
ADCOSDETECT	
adc.h	ADC_configOSDetectMode
ADCCOUNTER	
-	
ADCREV	
-	
ADCOFFTRIM	
adc.c	ADC_setOffsetTrim
ADCPPB1CONFIG	
adc.h	ADC_setupPPB
adc.h	ADC_enablePPBEventCBCClear
adc.h	ADC_disablePPBEventCBCClear
adc.h	ADC_enablePPBTwosComplement
adc.h	ADC_disablePPBTwosComplement
ADCPPB1STAMP	
adc.h	ADC_getPPBDelayTimeStamp
ADCPPB1OFFCAL	
adc.h	ADC_setPPBCalibrationOffset
ADCPPB1OFFREF	
adc.h	ADC_setPPBReferenceOffset
ADCPPB1TRIPHI	

Table 20-109. ADC Registers to Driverlib Functions (continued)

File	Driverlib Function
adc.c	ADC_setPPBTripLimits
ADCPPB1TRIPLO	
adc.c	ADC_setPPBTripLimits
ADCPPB2CONFIG	
-	See PPB1CONFIG
ADCPPB2STAMP	
-	See PPB1STAMP
ADCPPB2OFFCAL	
-	See PPB1OFFCAL
ADCPPB2OFFREF	
-	See PPB1OFFREF
ADCPPB2TRIPHI	
-	See PPB1TRIPHI
ADCPPB2TRIPLO	
-	See PPB1TRIPLO
ADCPPB3CONFIG	
-	See PPB1CONFIG
ADCPPB3STAMP	
-	See PPB1STAMP
ADCPPB3OFFCAL	
-	See PPB1OFFCAL
ADCPPB3OFFREF	
-	See PPB1OFFREF
ADCPPB3TRIPHI	
-	See PPB1TRIPHI
ADCPPB3TRIPLO	
-	See PPB1TRIPLO
ADCPPB4CONFIG	
-	See PPB1CONFIG
ADCPPB4STAMP	
-	See PPB1STAMP
ADCPPB4OFFCAL	
-	See PPB1OFFCAL
ADCPPB4OFFREF	
-	See PPB1OFFREF
ADCPPB4TRIPHI	
-	See PPB1TRIPHI
ADCPPB4TRIPLO	
-	See PPB1TRIPLO
ADCINTCYCLE	
adc.h	ADC_setInterruptCycleOffset
ADCINLTRIM1	
adc.c	ADC_setINLTrim
ADCINLTRIM2	
adc.c	ADC_setINLTrim

Table 20-109. ADC Registers to Driverlib Functions (continued)

File	Driverlib Function
ADCINLTRIM3	
-	
ADCINLTRIM4	
adc.c	ADC_setINLTrim
ADCINLTRIM5	
adc.c	ADC_setINLTrim
ADCINLTRIM6	
-	
ADCRESULT0	
adc.h	ADC_readResult
ADCRESULT1	
-	See RESULT0
ADCRESULT2	
-	See RESULT0
ADCRESULT3	
-	See RESULT0
ADCRESULT4	
-	See RESULT0
ADCRESULT5	
-	See RESULT0
ADCRESULT6	
-	See RESULT0
ADCRESULT7	
-	See RESULT0
ADCRESULT8	
-	See RESULT0
ADCRESULT9	
-	See RESULT0
ADCRESULT10	
-	See RESULT0
ADCRESULT11	
-	See RESULT0
ADCRESULT12	
-	See RESULT0
ADCRESULT13	
-	See RESULT0
ADCRESULT14	
-	See RESULT0
ADCRESULT15	
-	See RESULT0
ADCPPB1RESULT	
adc.h	ADC_readPPBResult
ADCPPB2RESULT	
-	See PPB1RESULT
ADCPPB3RESULT	

Table 20-109. ADC Registers to Driverlib Functions (continued)

File	Driverlib Function
-	See PPB1RESULT
ADCPPB4RESULT	
-	See PPB1RESULT

Chapter 21

Buffered Digital-to-Analog Converter (DAC)

The buffered digital-to-analog converter (DAC) is an analog module that can output a programmable, arbitrary reference voltage.

21.1 Introduction

The buffered DAC module consists of an internal 12-bit DAC and an analog output buffer that is capable of driving an external load. An integrated pull-down resistor on the DAC output helps to provide a known pin voltage when the output buffer is disabled. This pull-down resistor cannot be disabled and remains as a passive component on the pin, even for other shared pinmux functions. The buffered DAC is a general-purpose DAC that can be used to generate a DC voltage in addition to AC waveforms such as sine waves, square waves, triangle waves and so forth. Software writes to the DAC value register can take effect immediately or can be synchronized with EPWMSYNCPER events.

21.1.1 DAC Related Collateral

Foundational Materials

- C2000 Academy - DAC
- *High Speed, Digital to Analog Converters Basics Application Report*
- *Real-Time Control Reference Guide*
  - Refer to the DAC section
- *Understanding Data Converters Application Report*

Getting Started Materials

- MathWorks F2807x/F2837xD/F2837xS/F28004x/F2838x DAC
  - NOTE: This is a non-TI (third party) site.

21.1.2 Features

Each buffered DAC has the following features:

- 12-bit programmable internal DAC
- Selectable reference voltage source
- Pull-down resistor on output
- Ability to synchronize with EPWMSYNCPER

21.1.3 Block Diagram

The block diagram for the buffered DAC is shown in Figure 21-1.

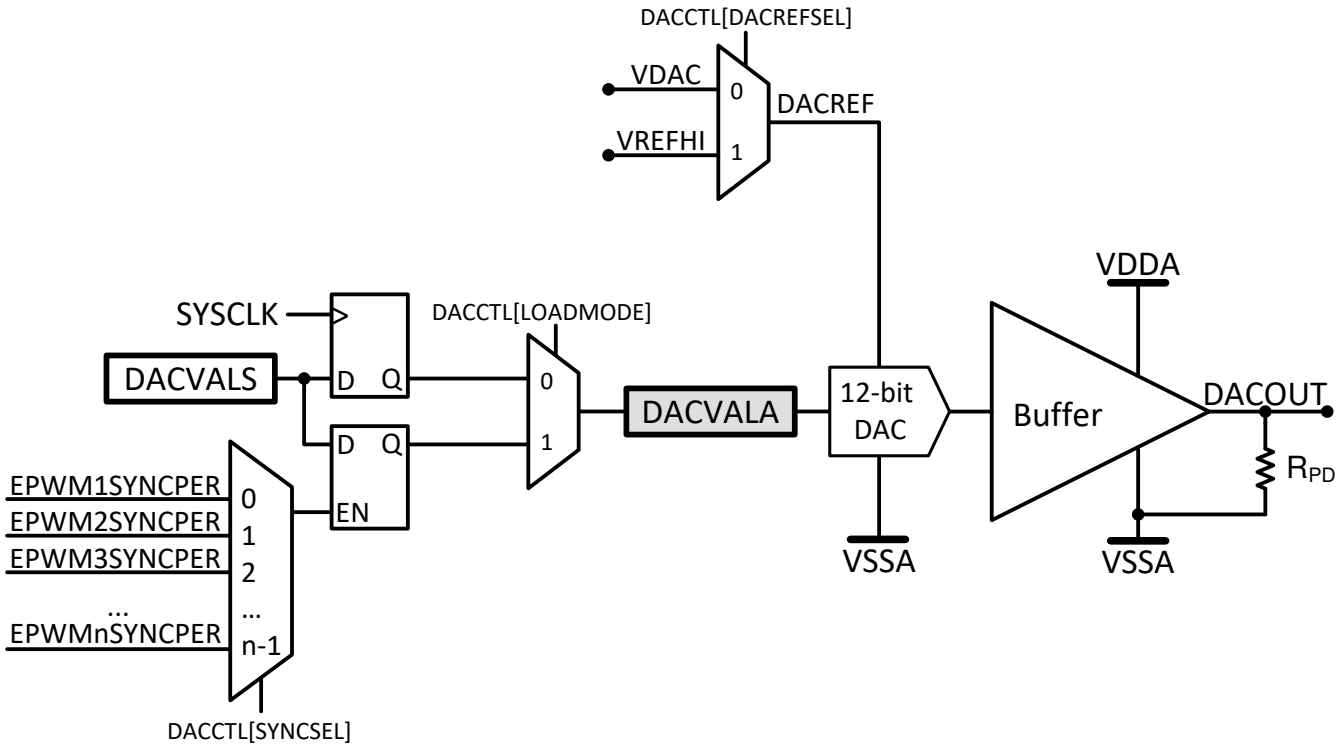


Figure 21-1. DAC Module Block Diagram

21.2 Using the DAC

The internal DAC's reference voltage source, DACREF, is selectable between VDAC and VREFHI.

Two sets of DACVAL registers, DACVALA and DACVALS, are present in the buffered DAC module. DACVALA is a read-only register that actively controls the buffered DAC value. DACVALS is a writable shadow register that loads into DACVALA either immediately or synchronized with the next EPWMSYNCPER event. If the clock to the buffered DAC is disabled while the buffered DAC is outputting a voltage, the output voltage remains unaffected, but DACVALA and DACVALS is no longer updated with register writes. Enabling the clock to the buffered DAC restores the DAC to the state before the clock was disabled.

The ideal output of the internal DAC is calculated with the following equation:

$$DACOUT = \frac{DACVALA * DACREF}{4096} \quad (7)$$

The output buffer of the buffered DAC can exhibit non-linear behavior near the supply rails (VDDA/VSSA). To determine the linear range of the buffered DAC, see the device-specific data sheet.

21.2.1 Initialization Sequence

1. Enable the buffered DAC clock.
2. Set DACREF with DACREFSEL.
3. Power up the buffered DAC with DACOUTEN.
4. Wait for the power-up time to elapse before outputting a voltage. To determine the power-up time of the buffered DAC, see the device data manual.
5. For predictable behavior of the buffered DAC, consecutive writes to DACVALS must be spaced apart according to the settling time of the buffered DAC. To determine the settling time of the buffered DAC, see the device data manual.

21.2.2 DAC Offset Adjustment

Zero offset error is defined as the difference between the voltage at midcode (2048) and 1.25V (for 2.5-V reference voltage). DAC offset error is calibrated at 2.5-V reference voltage and loaded into the DAC offset trim register as part of the Device_cal() function. If the DAC is used at any reference voltage other than 2.5V, the offset trim must be adjusted to ensure the offset error performance stays within the device-specific data sheet limits. The DAC offset register is a 16-bit register that contains the 8-bit signed offset trim in the lower half of the register. Use the function call DAC_tuneOffsetTrim() found in C2000Ware to adjust the offset.

21.2.3 EPWMSYNCPER Signal

EPWMSYNCPER comes from the Time-Base submodule of the EPWM. For a detailed description of how this signal is generated, refer to the *Time-Base Submodule* section of the *Enhanced Pulse Width Modulator (ePWM)* chapter.

The EPWMSYNCPER signal that loads DACVALA when DACCTL [LOADMODE] = 1 is a level trigger load. If TBCTR and TBPRD of the EPWM are both 0, EPWMSYNCPER is held at a high level and DACVALA is immediately loaded from DACVALS irrespective of the value of DACCTL [LOADMODE]. Due to this, configure the EPWM first before setting DACCTL [LOADMODE] to 1.

Note

The name of the sync signal that the GPDAC receives from the EPWM has been updated from PWMSYNC to EPWMSYNCPER (SYNCPER/PWMSYNCPER/EPWMxSYNCPER) to avoid confusion with the other EPWM sync signals EPWMSYNCI and EPWMSYNCO. For a description of these signals, see the *Enhanced Pulse Width Modulator (ePWM)* chapter.

21.3 Lock Registers

A DACLOCK register is provided to prevent spurious writes from modifying the DACCTL, DACVALS, and DACOUTEN registers. Once a register is protected through DACLOCK, write access are locked out until the device is reset.

21.4 Software

21.4.1 DAC Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/dac

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

21.4.1.1 Buffered DAC Enable

FILE: buffdac_ex1_enable.c

This example generates a voltage on the buffered DAC output, DACOUTA/ADCINA0 and uses the default DAC reference setting of VDAC.

External Connections

- When the DAC reference is set to VDAC, an external reference voltage must be applied to the VDAC pin. This can be accomplished by connecting a jumper wire from 3.3V to ADCINB0.

Watch Variables

- None.

21.4.1.2 Buffered DAC Random

FILE: buffdac_ex2_random.c

This example generates random voltages on the buffered DAC output, DACOUTA/ADCINA0 and uses the default DAC reference setting of VDAC.

External Connections

- When the DAC reference is set to VDAC, an external reference voltage must be applied to the VDAC pin. This can accomplished by connecting a jumper wire from 3.3V to ADCINB0.

Watch Variables

- None.

21.4.1.3 Buffered DAC Sine (buffdac_sine)

FILE: buffdac_ex3_waveform.c

This example generates a sine wave on the buffered DAC output, DACOUTA/ADCINA0 (HSEC Pin 9) and uses the default DAC reference setting of VDAC.

When the DAC reference is set to VDAC, an external reference voltage must be applied to the VDAC pin. This can accomplished by connecting a jumper wire from 3.3V to ADCINB0.

Run the included .js file to add the watch variables. This example uses the SGEN module. Documentation for the SGEN module can be found in the SGEN library directory.

The generated waveform can be adjusted with the following variables while running:

- *waveformGain* : Adjust the magnitude of the waveform. Range is from 0.0 to 1.0. The default value of 0.8003 centers the waveform within the linear range of the DAC
- *waveformOffset* : Adjust the offset of the waveform. Range is from -1.0 to 1.0. The default value of 0 centers the waveform
- *outputFreq_hz* : Adjust the output frequency of the waveform. Range is from 0 to maxOutputFreq_hz
- *maxOutputFreq_hz* : Adjust the max output frequency of the waveform. Range - See SGEN module documentation for how this affects other parameters

The generated waveform can be adjusted with the following variables/macros but require recompile:

- *samplingFreq_hz* : Adjust the rate at which the DAC is updated. Range - See SGEN module documentation for how this affects other parameters
- *SINEWAVE_TYPE* : The type of sine generated. Range - LOW_THD_SINE, HIGH_PRECISION_SINE

The following variables give additional information about the generated waveform: See SGEN module documentation for details

- *freqResolution_hz*
- *maxOutput_lsb* : Maximum value written to the DAC.
- *minOutput_lsb* : Minimum value written to the DAC.
- *pk_to_pk_lsb* : Magnitude of generated waveform.
- *cpuPeriod_us* : Period of cpu.
- *samplingPeriod_us* : The rate at which the DAC is updated. Note that samplingPeriod_us has to be greater than the DAC settling time.
- *interruptCycles* : Interrupt duration in cycles.
- *interruptDuration_us* : Interrupt duration in uS.
- *sgen* : The SGEN module instance.
- *DataLog* : Circular log of writes to the DAC.

21.5 DAC Registers

This section describes the Buffered Digital to Analog Converter registers.

21.5.1 DAC Base Address Table (C28)

Table 21-1. DAC Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
DacaRegs	DAC_REGS	DACA_BASE	0x0000_5C00	YES	YES	YES	YES	YES
DacbRegs	DAC_REGS	DACB_BASE	0x0000_5C10	YES	YES	YES	YES	YES
DaccRegs	DAC_REGS	DACC_BASE	0x0000_5C20	YES	YES	YES	YES	YES

21.5.2 DAC_REGS Registers

Table 21-2 lists the memory-mapped registers for the DAC_REGS registers. All register offset addresses not listed in Table 21-2 should be considered as reserved locations and the register contents should not be modified.

Table 21-2. DAC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	DACREV	DAC Revision Register		Go
1h	DACCTL	DAC Control Register	EALLOW	Go
2h	DACVALA	DAC Value Register - Active		Go
3h	DACVALS	DAC Value Register - Shadow		Go
4h	DACOUTEN	DAC Output Enable Register	EALLOW	Go
5h	DACLOCK	DAC Lock Register	EALLOW	Go
6h	DACTRIM	DAC Trim Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 21-3 shows the codes that are used for access types in this section.

Table 21-3. DAC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

21.5.2.1 DACREV Register (Offset = 0h) [Reset = 0h]

DACREV is shown in Figure 21-2 and described in Table 21-4.

Return to the Summary Table.

DAC Revision Register

Figure 21-2. DACREV Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
REV							
R-0h							

Table 21-4. DACREV Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	REV	R	0h	DAC Revision Reset type: SYSRSn

21.5.2.2 DACCTL Register (Offset = 1h) [Reset = 0h]

DACCTL is shown in Figure 21-3 and described in Table 21-5.

Return to the Summary Table.

DAC Control Register

Figure 21-3. DACCTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
SYNCSEL				RESERVED	LOADMODE	RESERVED	DACREFSEL
R/W-0h				R-0h	R/W-0h	R-0h	R/W-0h

Table 21-5. DACCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-4	SYNCSEL	R/W	0h	DAC EPWMSYNCPER select. Determines which EPWMSYNCPER signal will update the DACVALA register. Where n represents the maximum number of EPWMSYNCPER signals available on the device: 0 EPWM1SYNCPER 1 EPWM2SYNCPER 2 EPWM3SYNCPER ... n-1 EPWMnSYNCPER Reset type: SYSRSn
3	RESERVED	R	0h	Reserved
2	LOADMODE	R/W	0h	DACVALA load mode. Determines when the DACVALA register is updated with the value from DACVALS. 0 Load on next SYSCLK 1 Load on next EPWMSYNCPER specified by SYNCSEL Reset type: SYSRSn
1	RESERVED	R	0h	Reserved
0	DACREFSEL	R/W	0h	DAC reference select. Selects which voltage references are used by the DAC. 0 VDAC/VSSA are the reference voltages 1 ADC VREFHI/VSSA are the reference voltages Reset type: SYSRSn

21.5.2.3 DACVALA Register (Offset = 2h) [Reset = 0h]

DACVALA is shown in Figure 21-4 and described in Table 21-6.

Return to the Summary Table.

DAC Value Register - Active

Figure 21-4. DACVALA Register

15	14	13	12	11	10	9	8
RESERVED				DACVALA			
R-0h				R-0h			
7	6	5	4	3	2	1	0
DACVALA							
R-0h							

Table 21-6. DACVALA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DACVALA	R	0h	Active output code currently driven by the DAC Reset type: SYSRSn

21.5.2.4 DACVALS Register (Offset = 3h) [Reset = 0h]

DACVALS is shown in Figure 21-5 and described in Table 21-7.

Return to the Summary Table.

DAC Value Register - Shadow

Figure 21-5. DACVALS Register

15	14	13	12	11	10	9	8
RESERVED				DACVALS			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
DACVALS							
R/W-0h							

Table 21-7. DACVALS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DACVALS	R/W	0h	Shadow output code to be loaded into DACVALA Reset type: SYSRSn

21.5.2.5 DACOUTEN Register (Offset = 4h) [Reset = 0h]

DACOUTEN is shown in Figure 21-6 and described in Table 21-8.

Return to the Summary Table.

DAC Output Enable Register

Figure 21-6. DACOUTEN Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							DACOUTEN
R-0h							R/W-0h

Table 21-8. DACOUTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	DACOUTEN	R/W	0h	DAC output enable 0 DAC output is disabled 1 DAC output is enabled Reset type: SYSRSn

21.5.2.6 DACLOCK Register (Offset = 5h) [Reset = 0h]

DACLOCK is shown in Figure 21-7 and described in Table 21-9.

Return to the Summary Table.

DAC Lock Register

Figure 21-7. DACLOCK Register

15	14	13	12	11	10	9	8
KEY				RESERVED			
R-0/W-0h				R-0h			
7	6	5	4	3	2	1	0
RESERVED					DACOUTEN	DACVAL	DACCTL
R-0h					R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 21-9. DACLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	KEY	R-0/W	0h	Writes to this register succeed only if this field is written with a value of 0xA. Only 16-bit writes will succeed (provided the KEY matches). Read-modify-writes to individual bits in this register will be ignored. Reset type: SYSRSn
11-3	RESERVED	R	0h	Reserved
2	DACOUTEN	R/WSonce	0h	Lock write-access to the DACOUTEN register. 0 DACOUTEN register is not locked. Write 0 to this bit has no effect. 1 DACOUTEN register is locked. Only a system reset can clear this bit. Reset type: SYSRSn
1	DACVAL	R/WSonce	0h	Lock write-access to the DACVALS register. 0 DACVALS register is not locked. Write 0 to this bit has no effect. 1 DACVALS register is locked. Only a system reset can clear this bit. Reset type: SYSRSn
0	DACCTL	R/WSonce	0h	Lock write-access to the DACCTL register. 0 DACCTL register is not locked. Write 0 to this bit has no effect. 1 DACCTL register is locked. Only a system reset can clear this bit. Reset type: SYSRSn

21.5.2.7 DACTRIM Register (Offset = 6h) [Reset = 0h]

DACTRIM is shown in Figure 21-8 and described in Table 21-10.

Return to the Summary Table.

DAC Trim Register

Figure 21-8. DACTRIM Register

15	14	13	12	11	10	9	8
RESERVED				RESERVED			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
OFFSET_TRIM							
R/W-0h							

Table 21-10. DACTRIM Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-8	RESERVED	R/W	0h	Reserved
7-0	OFFSET_TRIM	R/W	0h	DAC Offset Trim. This register should not be modified unless specifically indicated by TI Errata or other documentation. Modifying the contents of this register could cause this module to operate outside of datasheet specifications. Reset type: SYSRSn

21.5.3 DAC Registers to Driverlib Functions

Table 21-11. DAC Registers to Driverlib Functions

File	Driverlib Function
DACREV	
dac.h	DAC_getRevision
DACCTL	
dac.h	DAC_setReferenceVoltage
dac.h	DAC_setLoadMode
dac.h	DAC_setPWMSyncSignal
DACVALA	
dac.h	DAC_getActiveValue
DACVALS	
dac.h	DAC_setShadowValue
dac.h	DAC_getShadowValue
DACOUTEN	
dac.h	DAC_enableOutput
dac.h	DAC_disableOutput
DACLOCK	
dac.h	DAC_lockRegister
dac.h	DAC_isRegisterLocked
DACTRIM	
dac.c	DAC_tuneOffsetTrim
dac.h	DAC_setOffsetTrim
dac.h	DAC_getOffsetTrim

Chapter 22

Comparator Subsystem (CMPSS)

The Comparator Subsystem (CMPSS) consists of analog comparators and supporting circuits that are useful for power applications such as peak current mode control, switched-mode power, power factor correction, voltage trip monitoring, and so forth.

See the *C2000 Real-Time Control Peripheral Reference Guide* for a list of all devices with modules of the same type, to determine the differences between the types, and for a list of device-specific differences within a type.

22.1 Introduction

The comparator subsystem is built around a number of modules. Each subsystem contains two comparators, two reference 12-bit DACs, and two digital filters. The subsystem also includes one ramp generator. The ramp generator ramps down only.Comparators are denoted "H" or "L" within each module where “H” and “L” represent high and low, respectively. Each comparator generates a digital output which indicates whether the voltage on the positive input is greater than the voltage on the negative input. The positive input of the comparator is driven from an external pin (see the *Analog Subsystem* chapter for mux options available to the CMPSS). The negative input can be driven by an external pin or by the programmable reference 12-bit DAC . Each comparator output passes through a programmable digital filter that can remove spurious trip signals. An unfiltered output is also available if filtering is not required. A ramp generator circuit is optionally available to control the reference 12-bit DAC value for the high comparator in the subsystem.

22.1.1 CMPSS Related Collateral

Foundational Materials

- C2000 Academy - CMPSS
- *Real-Time Control Reference Guide*
  - Refer to the Comparator section

Expert Materials

- *Peak Current Control Realization for Boost Circuit Based On C2000™ MCU Application Report*
- *Peak Current Mode Controlled PSFB Converter Reference Design Using C2000™ Real-time MCU*
- *Understanding and Applying Current-Mode Control Theory Application Report*

22.1.2 Features

Each CMPSS includes:

- Two analog comparators
- Two programmable reference 12-bit DACs
- One decrementing ramp generator
- Two digital filters, max filter clock prescale = 2^{10}
- Ability to synchronize submodules with EPWMSYNCPER
- Ability to extend clear signal with EPWMBLANK
- Ability to synchronize output with SYSCLK
- Ability to latch output
- Ability to invert output
- Option to use hysteresis on the input
- Option for negative input of comparator to be driven by an external signal or by the reference DAC
- The DAC reference voltage can be set to VDDA or VDAC

22.1.3 Block Diagram

The block diagram for the CMPSS is shown in Figure 22-1.

- CTRIPx(x= "H" or "L") signals are connected to the ePWM X-BAR for ePWM trip response. See the *Enhanced Pulse Width Modulator (ePWM)* chapter for more details on the ePWM X-BAR mux configuration.
- CTRIPxOUTx(x= "H" or "L") signals are connected to the Output X-BAR for external signaling. See the *General-Purpose Input/Output (GPIO)* chapter for more details on the Output X-BAR mux configuration.

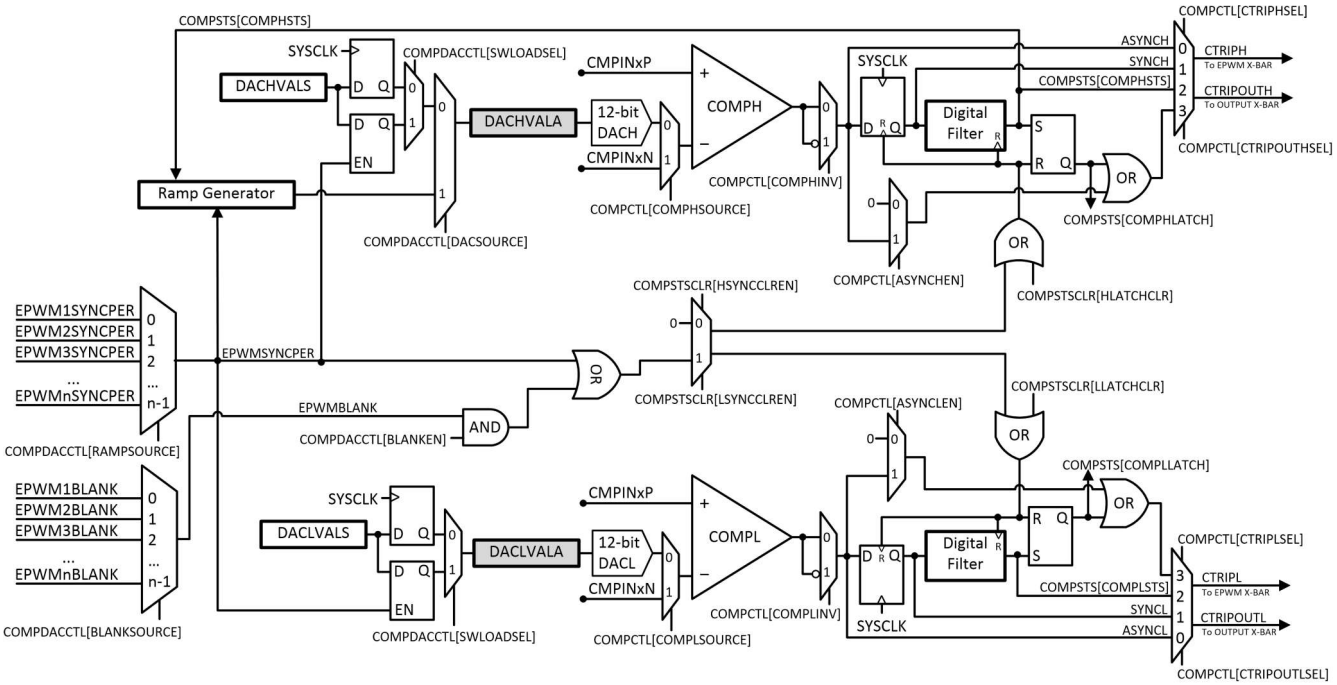


Figure 22-1. CMPSS Module Block Diagram

22.2 Comparator

The comparator generates a high digital output when the voltage on the positive input is greater than the voltage on the negative input, and a low digital output when the voltage on the positive input is less than the voltage on the negative input. The comparator is illustrated in Figure 22-2.

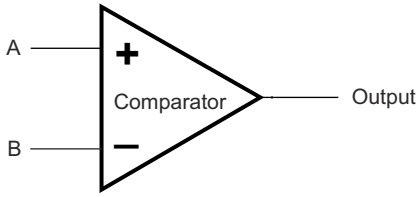


Figure 22-2. Comparator Block Diagram

Voltages	Output
Voltage A > Voltage B	1
Voltage A < Voltage B	0

22.3 Reference DAC

Each reference 12-bit DAC can be configured to drive a reference voltage into the negative input of the respective comparator. The reference 12-bit DAC output is internal only and cannot be observed externally.

Two sets of DACxVAL registers, DACxVALA and DACxVALS, are present for each reference 12-bit DAC. DACxVALA is a read-only register that actively controls the reference 12-bit DAC value. DACxVALS is a writable shadow register that loads into DACxVALA either immediately or synchronized with the next EPWMSYNCPER event. The high and low reference 12-bit DAC (DACx) can optionally source the register DACxVALA value from the ramp generator instead of the register DACxVALS.

The operating range of the reference 12-bit DAC is bounded by DACREF and VSSA. The high-voltage reference is VDDA by default, but the high voltage reference can be configured to be VDAC. The reference 12-bit DAC is illustrated in Figure 22-3.

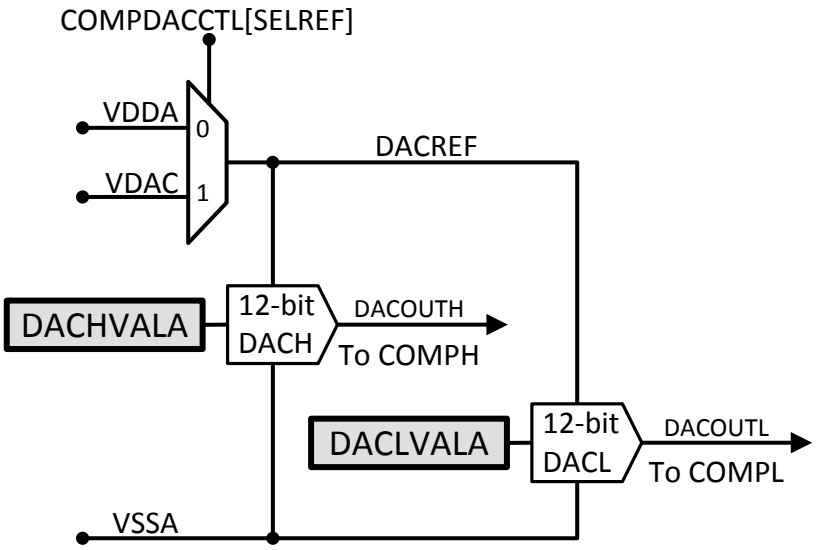


Figure 22-3. Reference DAC Block Diagram

The ideal output of the reference 12-bit DAC can be calculated as:

$$DACOUT = \frac{(1 + DACVALA)*DACREF}{4096} \quad (8)$$

Note

- In the situations where both the DACH and DACL are driving the high and low comparators, a trip on one comparator can temporarily disturb the DAC output of the other comparator. The amount and length of time of this disturbance is specified in the device data sheet as “CMPSS DAC output disturbance” and “CMPSS DAC disturbance time”, respectively.

  Users must design their system carefully so that if the input signal crosses either DACH or DACL and trips the associated comparator, the input signal stays more than a “CMPSS DAC output disturbance” away from the other comparator trip point for “CMPSS DAC disturbance time”.
- The DACH setting must always be higher than the DACL setting. If the user is not using the DACL, the DACLVALS register must be set to 0 to avoid the comparator COMPL from tripping and affecting the DACH. In this case, there is no limitation on the DACHVALS setting. Accordingly, when not using the DACH, the user must set the DACHVALS register to the maximum.
- The CMPSS instance can be enabled before programming the reference DAC values.

22.4 Ramp Generator

This section discusses the characteristics and behavior of the ramp generator.

22.4.1 Ramp Generator Overview

The ramp generator produces a falling ramp input for the high-reference 12-bit DAC when selected. In this mode, the reference 12-bit DAC uses the most-significant 12 bits of the RAMPSTS countdown register as the input. The low 4 bits of the RAMPSTS countdown register effectively act as a prescale for the falling ramp rate configurable with RAMPDECVALA.

The ramp generator is enabled by setting DACSOURCE = 1. When DACSOURCE = 1 is selected, the value of RAMPSTS is loaded from RAMPMAXREFS and the register remains static until the selected EPWMSYNCPER signal is received. After receiving the selected EPWMSYNCPER signal, the value of RAMPDECVALA is subtracted from RAMPSTS on every subsequent SYSCLK cycle.

To prevent the subtraction from commencing a SYSCLK cycle after a EPWMSYNCPER event, the RAMPDLYA register that serves as a delay counter can be used to hold off the RAMPSTS subtraction. On receiving a EPWMSYNCPER event, the value of RAMPDLYA is decremented by one on every SYSCLK cycle until the register reaches zero. So, the RAMPSTS subtraction only begins when RAMPDLYA is zero.

22.4.2 Ramp Generator Behavior

The ramp generator makes state changes on every rising edge of DACSOURCE, EPWMSYNCPER, and COMPHSTS.

On the rising edge of DACSOURCE: RAMPMAXREFA, RAMPDECVALA, and RAMPDLYA are loaded with their shadow registers. RAMPSTS is loaded with RAMPMAXREFS.

On the rising edge of the selected EPWMSYNCPER: RAMPMAXREFA, RAMPDECVALA, and RAMPDLYA are loaded with their shadow registers. RAMPSTS is loaded with RAMPMAXREFS and starts decrementing when RAMPDLYA counter reaches zero.

On the rising edge of COMPHSTS with RAMPLOADSEL = 1: RAMPMAXREFA, RAMPDECVALA, and RAMPDLYA are loaded with their shadow registers. RAMPSTS is loaded with RAMPMAXREFS and stops decrementing.

On the rising edge of COMPHSTS with RAMPLOADSEL = 0: RAMPSTS is loaded with RAMPMAXREFA and stops decrementing.

Additionally, if the value of RAMPSTS reaches zero, the RAMPSTS register remains static at zero until the next EPWMSYNCPER is received. These state changes are illustrated in the ramp generator block diagram in Figure 22-4.

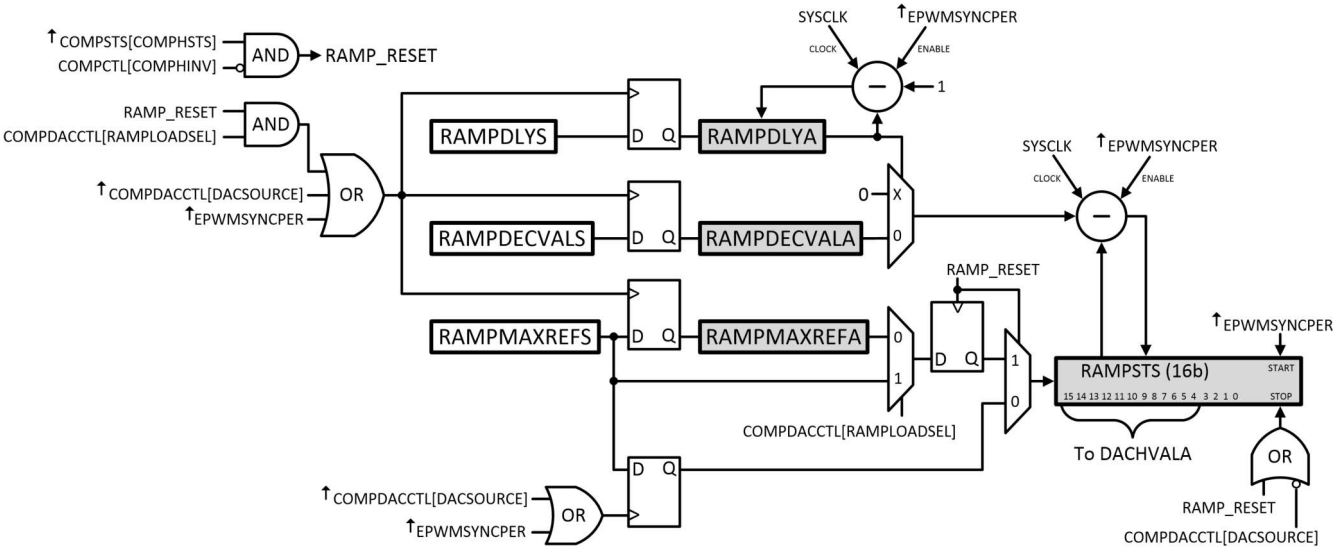


Figure 22-4. Ramp Generator Block Diagram

22.4.3 Ramp Generator Behavior at Corner Cases

Since the ramp generator makes state changes on every rising edge of EPWMSYNCPER and COMPHSTS, the following behavior can be expected on instances when these two events occur simultaneously or very close together.

Case 1: COMPHSTS rising edge occurs one or more cycles before EPWMSYNCPER rising edge.
RAMPSTS stops decrementing on COMPHSTS rising edge event. RAMPSTS starts decrementing on EPWMSYNCPER rising edge event when RAMPDLYA reaches 0.

Case 2: COMPHSTS rising edge occurs simultaneously as EPWMSYNCPER rising edge.
EPWMSYNCPER rising edge event takes precedence and RAMPSTS starts decrementing when RAMPDLYA reaches 0. COMPHSTS rising edge event is ignored and does not halt RAMPSTS.

Case 3: COMPHSTS rising edge occurs one or more cycles after EPWMSYNCPER rising edge but before RAMPDLYA reaches 0.
RAMPSTS does not decrement when RAMPDLYA reaches 0.

Case 4: COMPHSTS rising edge occurs simultaneously as RAMPDLYA reaches 0 from EPWMSYNCPER rising edge.
RAMPSTS does not decrement.

This behavior is also illustrated in Figure 22-5.

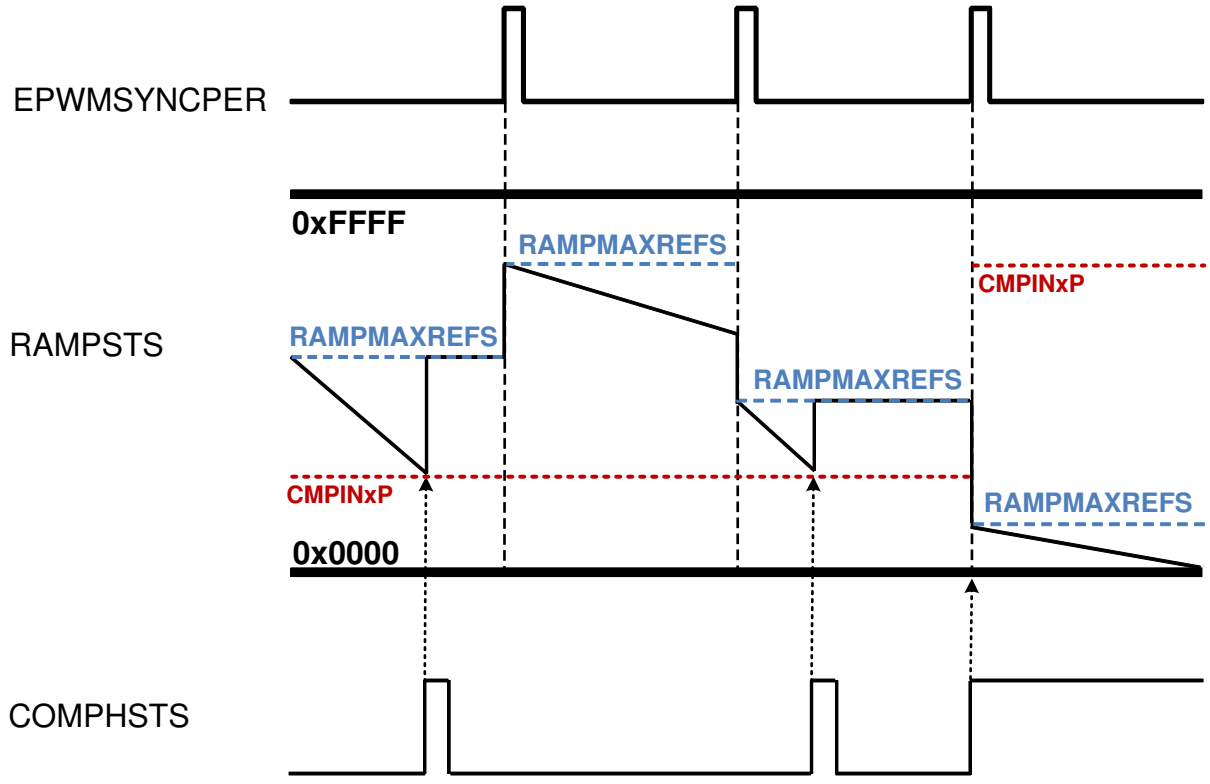


Figure 22-5. Ramp Generator Behavior

22.5 Digital Filter

The digital filter works on a window of FIFO samples (SAMPWIN) taken from the input. The filter output resolves to the majority value of the sample window, where majority is defined by the threshold (THRESH) value. If the majority threshold is not satisfied, the filter output remains unchanged.

For proper operation, the value of THRESH must be greater than SAMPWIN / 2 and less than or equal to SAMPWIN.

A prescale function (CLKPRESCALE) determines the filter sampling rate, where the filter FIFO captures one sample every prescale system clocks. Old data from the FIFO is discarded.

Note that for SAMPWIN, THRESH and CLKPRESCALE, the internal number used by the digital filter is + 1 in all cases. In essence, samples = SAMPWIN + 1, threshold = THRESH + 1 and prescale = CLKPRESCALE + 1.

A conceptual model of the digital filter is shown in Figure 22-6.

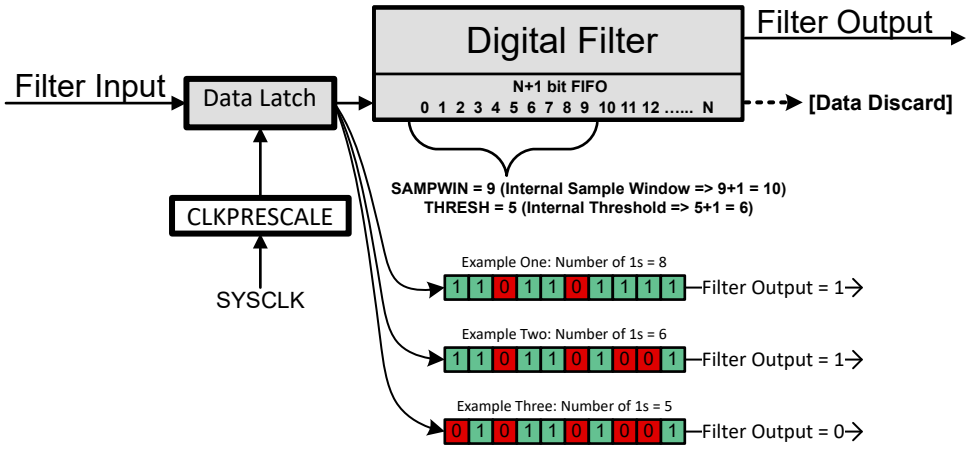


Figure 22-6. Digital Filter Behavior

Equivalent C code of the filter implementation is:

```
if (FILTER_OUTPUT == 0) {
    if (Num_1s_in_SAMPWIN >= THRESH) {
        FILTER_OUTPUT = 1;
    }
}
else {
    if (Num_0s_in_SAMPWIN >= THRESH) {
        FILTER_OUTPUT = 0;
    }
}
```

22.5.1 Filter Initialization Sequence

For proper operation of the digital filter, the following initialization sequence is recommended:

1. Configure and enable the comparator for operation.
2. Configure the digital filter parameters for operation:
   - Set SAMPWIN for the number of samples to monitor in the FIFO window.
   - Set THRESH for the threshold required for majority qualification.
   - Set CLKPRESCALE for the digital filter clock prescale value.
3. Initialize the sample values in the digital FIFO window by setting FILINIT.
4. Clear COMPSTS latch using COMPSTSCLR, if the latched path is desired.
5. Configure the CTRIP and CTRIPOUT signal paths.
6. If desired, configure the destination module, for example, ePWM, GPIO, and so on to accept the filtered signals.

22.6 Using the CMPSS

22.6.1 LATCHCLR, EPWMSYNCPER and EPWMBLANK Signals

The LATCHCLR signal holds the digital filter, synchronization block, and the latch output in reset (0) after the required delays. The LATCHCLR signal is activated in software using xLATCHCLR (x = H or L). The LATCHCLR signal can also be activated by EPWMSYNCPER when xSYNCCLREN (x = H or L) is set. If a longer LATCHCLR signal is required, the EPWMBLANK signal can be used to extend the LATCHCLR signal by setting BLANKEN.

EPWMSYNCPER and EPWMBLANK (BLANKWDW) come from the Time-Base and Digital Compare submodules of the EPWM, respectively. For a detailed description of how these two signals are generated, refer to the respective submodule section in the *Enhanced Pulse Width Modulator (ePWM)* chapter.

The EPWMSYNCPER signal that loads DACxVALA when COMPDACCTL [SWLOADSEL] = 1 is a level trigger load. If TBCTR and TBPRD of the EPWM are both 0, EPWMSYNCPER is held at level high and DACxVALA is loaded immediately from DACxVALS irrespective of the value of COMPDACCTL [SWLOADSEL]. Due to this, configure the EPWM first before setting COMPDACCTL [SWLOADSEL] to 1.

Note

The name of the sync signal that the CMPSS receives from the EPWM has been updated from PWMSYNC to EPWMSYNCPER (SYNCPER/PWMSYNCPER/EPWMxSYNCPER) to avoid confusion with the other EPWM sync signals EPWMSYNCI and EPWMSYNCO. For a description of what are these signals, see the *Enhanced Pulse Width Modulator (ePWM)* chapter.

22.6.2 Synchronizer, Digital Filter, and Latch Delays

The synchronization block adds a delay of 1-2 sysclks. If the digital filter is bypassed (all filter settings are 0), the digital filter adds a delay of 2 sysclks. The latch adds 1 sysclk delay.

22.6.3 Calibrating the CMPSS

The CMPSS has two sources of offset errors: comparator offset error and compdac offset error. In the data manual, the comparator offset error is referred to as ***Input referred offset error*** and compdac offset error is referred to as ***Static offset error***. See the device specific data manual for their values.

If both inputs of the comparator are driven from a pin, only the comparator offset error applies. However if the inverting input of the comparator is driven from the compdac, then only the compdac offset error applies. This is because the compdac offset error includes comparator offset error.

Due to the offset errors, it is recommended that the CMPSS be calibrated to ensure trips happen at the expected levels. The following flow outlines how the calibration can be performed if the inverting input of the comparator is driven from the compdac.

Notes before calibration:

1. A static DC signal is required on the non-inverting input of the comparator.
2. Hysteresis can be disabled for calibration and can be re-enabled after calibration is complete.
3. A noisy input can make calibration difficult so it is recommended to use the latch with non-zero filter settings depending on how noisy is the signal on the non-inverting input.

This approach sweeps down the compdac:

1. Set the starting compdac value to max, 0xFFF.
   - Optional: Instead of setting the starting compdac value to maximum, set to ***Vtarget*** + ***Static offset error*** + ***Margin***. Where ***Vtarget*** is the approximate DC voltage on the non-inverting input, ***Static offset error*** is the compdac offset error specification and ***Margin*** is some amount of guard band. This can lead to a faster calibration but only works if ***Vtarget*** is known. Alternatively, if ***Vtarget*** is unknown, the ADC can be used to convert ***Vtarget***.
2. Decrement compdac value by 1.
3. Wait for compdac to settle.
4. Clear latch.
5. Wait for possible latch set.
6. If latch is set, trip code is found exit.
   - Optional: The trip code can be double checked by:
     a. Increasing compdac value by 1.
     b. Clear latch.
     c. Wait for possible latch set.
     d. Latch can be unset.
7. If latch is unset, go back to step 2 and repeat.

It is also possible to calibrate the CMPSS, if both inputs of the comparator are driven from a pin. For this case, the flow stays the same but the voltage on the inverting pin of the comparator is swept externally.

22.6.4 Enabling and Disabling the CMPSS Clock

If the clock to the CMPSS module is disabled while the comparator is active, the following behavior can be expected:

- The comparator remains unaffected and continues to trip from voltages on the inputs.
- If the reference 12-bit DAC is driving the negative input of the comparator, the voltage on the negative input remains static and unaffected but DACVALA can no longer be updated from the ramp generator or DACVALS.
- The ramp generator, synchronize block and digital filter freeze on their current states.

Enabling the clock to the CMPSS restores the clock to the state before the clock was disabled.

22.7 Software

22.7.1 CMPSS Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/cmpss

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

22.7.1.1 CMPSS Asynchronous Trip

FILE: cmpss_ex1_asynch.c

This example enables the CMPSS1 COMPH comparator and feeds the asynchronous CTRIPOUTH signal to the GPIO14/OUTPUTXBAR3 pin and CTRIPH to GPIO15/EPWM8B.

CMPSS is configured to generate trip signals to trip the EPWM signals. CMPIN1P is used to give positive input and internal DAC is configured to provide the negative input. Internal DAC is configured to provide a signal at VDD/2. An EPWM signal is generated at GPIO15 and is configured to be tripped by CTRIPOUTH.

When a low input(VSS) is provided to CMPIN1P,
- Trip signal(GPIO14) output is low
- PWM8B(GPIO15) gives a PWM signal

When a high input(higher than VDD/2) is provided to CMPIN1P,
- Trip signal(GPIO14) output turns high
- PWM8B(GPIO15) gets tripped and outputs as high

External Connections
- Give input on CMPIN1P (HSEC Pin 15)
- Outputs can be observed on GPIO14 and GPIO15 using an oscilloscope

Watch Variables
- None

22.7.1.2 CMPSS Digital Filter Configuration

FILE: cmpss_ex2_digital_filter.c

This example enables the CMPSS1 COMPH comparator and feeds the output through the digital filter to the GPIO14/OUTPUTXBAR3 pin.

CMPIN1P is used to give positive input and internal DAC is configured to provide the negative input. Internal DAC is configured to provide a signal at VDD/2.

When a low input(VSS) is provided to CMPIN1P,
- GPIO14 output is low

When a high input(higher than VDD/2) is provided to CMPIN1P,
- GPIO14 output turns high

22.8 CMPSS Registers

This section describes the CMPSS Registers.

22.8.1 CMPSS Base Address Table (C28)

Table 22-1. CMPSS Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
Cmpss1Regs	CMPSS_REGS	CMPSS1_BASE	0x0000_5C80	YES	YES	YES	YES	YES
Cmpss2Regs	CMPSS_REGS	CMPSS2_BASE	0x0000_5CA0	YES	YES	YES	YES	YES
Cmpss3Regs	CMPSS_REGS	CMPSS3_BASE	0x0000_5CC0	YES	YES	YES	YES	YES
Cmpss4Regs	CMPSS_REGS	CMPSS4_BASE	0x0000_5CE0	YES	YES	YES	YES	YES
Cmpss5Regs	CMPSS_REGS	CMPSS5_BASE	0x0000_5D00	YES	YES	YES	YES	YES
Cmpss6Regs	CMPSS_REGS	CMPSS6_BASE	0x0000_5D20	YES	YES	YES	YES	YES
Cmpss7Regs	CMPSS_REGS	CMPSS7_BASE	0x0000_5D40	YES	YES	YES	YES	YES
Cmpss8Regs	CMPSS_REGS	CMPSS8_BASE	0x0000_5D60	YES	YES	YES	YES	YES

22.8.2 CMPSS_REGS Registers

Table 22-2 lists the memory-mapped registers for the CMPSS_REGS registers. All register offset addresses not listed in Table 22-2 should be considered as reserved locations and the register contents should not be modified.

Table 22-2. CMPSS_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	COMPCTL	CMPSS Comparator Control Register	EALLOW	Go
1h	COMPHYSCTL	CMPSS Comparator Hysteresis Control Register	EALLOW	Go
2h	COMPSTS	CMPSS Comparator Status Register		Go
3h	COMPSTSCLR	CMPSS Comparator Status Clear Register	EALLOW	Go
4h	COMPDACCTL	CMPSS DAC Control Register	EALLOW	Go
6h	DACHVALS	CMPSS High DAC Value Shadow Register		Go
7h	DACHVALA	CMPSS High DAC Value Active Register		Go
8h	RAMPMAXREFA	CMPSS Ramp Max Reference Active Register		Go
Ah	RAMPMAXREFS	CMPSS Ramp Max Reference Shadow Register		Go
Ch	RAMPDECVALA	CMPSS Ramp Decrement Value Active Register		Go
Eh	RAMPDECVALS	CMPSS Ramp Decrement Value Shadow Register		Go
10h	RAMPSTS	CMPSS Ramp Status Register		Go
12h	DACLVALS	CMPSS Low DAC Value Shadow Register		Go
13h	DACLVALA	CMPSS Low DAC Value Active Register		Go
14h	RAMPDLYA	CMPSS Ramp Delay Active Register		Go
15h	RAMPDLYS	CMPSS Ramp Delay Shadow Register		Go
16h	CTRIPLFILCTL	CTRIPL Filter Control Register	EALLOW	Go
17h	CTRIPLFILCLKCTL	CTRIPL Filter Clock Control Register	EALLOW	Go
18h	CTRIPHFILCTL	CTRIPH Filter Control Register	EALLOW	Go
19h	CTRIPHFILCLKCTL	CTRIPH Filter Clock Control Register	EALLOW	Go
1Ah	COMPLOCK	CMPSS Lock Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 22-3 shows the codes that are used for access types in this section.

Table 22-3. CMPSS_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 22-3. CMPSS_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

22.8.2.1 COMPCTL Register (Offset = 0h) [Reset = 0h]

COMPCTL is shown in Figure 22-7 and described in Table 22-4.

Return to the Summary Table.

CMPSS Comparator Control Register

Figure 22-7. COMPCTL Register

15	14	13	12	11	10	9	8
COMPDACE	ASYNCLEN	CTRIPOUTLSEL		CTRIPLSEL		COMPLINV	COMPLSOURCE
R/W-0h	R/W-0h	R/W-0h		R/W-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	ASYNCHEN	CTRIPOUTHSEL		CTRIPHSEL		COMPHINV	COMPHSOURCE
R-0h	R/W-0h	R/W-0h		R/W-0h		R/W-0h	R/W-0h

Table 22-4. COMPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	COMPDACE	R/W	0h	Comparator/DAC enable. 0 Comparator/DAC disabled 1 Comparator/DAC enabled Reset type: SYSRSn
14	ASYNCLEN	R/W	0h	Low comparator asynchronous path enable. Allows asynchronous comparator output to feed into OR gate with latched digital filter signal when CTRIPLSEL=3 or CTRIPOUTLSEL=3. 0 Asynchronous comparator output does not feed into OR gate with latched digital filter output 1 Asynchronous comparator output feeds into OR gate with latched digital filter output Reset type: SYSRSn
13-12	CTRIPOUTLSEL	R/W	0h	Low comparator CTRIPOUTL source select. 0 Asynchronous comparator output drives CTRIPOUTL 1 Synchronous comparator output drives CTRIPOUTL 2 Output of digital filter drives CTRIPOUTL 3 Latched output of digital filter drives CTRIPOUTL Reset type: SYSRSn
11-10	CTRIPLSEL	R/W	0h	Low comparator CTRIPL source select. 0 Asynchronous comparator output drives CTRIPL 1 Synchronous comparator output drives CTRIPL 2 Output of digital filter drives CTRIPL 3 Latched output of digital filter drives CTRIPL Reset type: SYSRSn
9	COMPLINV	R/W	0h	Low comparator output invert. 0 Output of comparator is not inverted 1 Output of comparator is inverted Reset type: SYSRSn
8	COMPLSOURCE	R/W	0h	Low comparator input source. 0 Inverting input of comparator driven by internal DAC 1 Inverting input of comparator driven through external pin Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6	ASYNCHEN	R/W	0h	High comparator asynchronous path enable. Allows asynchronous comparator output to feed into OR gate with latched digital filter signal when CTRIPHSEL=3 or CTRIPOUTHSEL=3. 0 Asynchronous comparator output does not feed into OR gate with latched digital filter output 1 Asynchronous comparator output feeds into OR gate with latched digital filter output Reset type: SYSRSn

Table 22-4. COMPCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-4	CTRIPOUTHSEL	R/W	0h	High comparator CTRIPOUTH source select. 0 Asynchronous comparator output drives CTRIPOUTH 1 Synchronous comparator output drives CTRIPOUTH 2 Output of digital filter drives CTRIPOUTH 3 Latched output of digital filter drives CTRIPOUTH Reset type: SYSRSn
3-2	CTRIPHSEL	R/W	0h	High comparator CTRIPH source select. 0 Asynchronous comparator output drives CTRIPH 1 Synchronous comparator output drives CTRIPH 2 Output of digital filter drives CTRIPH 3 Latched output of digital filter drives CTRIPH Reset type: SYSRSn
1	COMPHINV	R/W	0h	High comparator output invert. 0 Output of comparator is not inverted 1 Output of comparator is inverted Reset type: SYSRSn
0	COMPHSOURCE	R/W	0h	High comparator input source. 0 Inverting input of comparator driven by internal DAC 1 Inverting input of comparator driven through external pin Reset type: SYSRSn

22.8.2.2 COMPHYSCTL Register (Offset = 1h) [Reset = 0h]

COMPHYSCTL is shown in Figure 22-8 and described in Table 22-5.

Return to the Summary Table.

CMPSS Comparator Hysteresis Control Register

Figure 22-8. COMPHYSCTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				COMPHYS			
R-0h				R/W-0h			

Table 22-5. COMPHYSCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3-0	COMPHYS	R/W	0h	Comparator hysteresis. Sets the amount of hysteresis on the comparator inputs. 0 None 1 Set to typical hysteresis 2 Set to 2x of typical hysteresis 3 Set to 3x of typical hysteresis 4 Set to 4x of typical hysteresis others : undefined Reset type: SYSRSn

22.8.2.3 COMPSTS Register (Offset = 2h) [Reset = 0h]

COMPSTS is shown in Figure 22-9 and described in Table 22-6.

Return to the Summary Table.

CMPSS Comparator Status Register

Figure 22-9. COMPSTS Register

15	14	13	12	11	10	9	8
RESERVED						COMPLLATCH	COMPLSTS
R-0h						R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED						COMPHLATCH	COMPHSTS
R-0h						R-0h	R-0h

Table 22-6. COMPSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9	COMPLLATCH	R	0h	Latched value of low comparator digital filter output Reset type: SYSRSn
8	COMPLSTS	R	0h	Low comparator digital filter output Reset type: SYSRSn
7-2	RESERVED	R	0h	Reserved
1	COMPHLATCH	R	0h	Latched value of high comparator digital filter output Reset type: SYSRSn
0	COMPHSTS	R	0h	High comparator digital filter output Reset type: SYSRSn

22.8.2.4 COMPSTSCLR Register (Offset = 3h) [Reset = 0h]

COMPSTSCLR is shown in Figure 22-10 and described in Table 22-7.

Return to the Summary Table.

CMPSS Comparator Status Clear Register

Figure 22-10. COMPSTSCLR Register

15	14	13	12	11	10	9	8
RESERVED					LSYNCCLREN	LLATCHCLR	RESERVED
R-0h					R/W-0h	R-0/W1S-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED					HSYNCCLREN	HLATCHCLR	RESERVED
R-0h					R/W-0h	R-0/W1S-0h	R-0h

Table 22-7. COMPSTSCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	Reserved
10	LSYNCCLREN	R/W	0h	Low comparator latch EPWMSYNCPER clear. Enable EPWMSYNCPER reset of low comparator digital filter output latch COMPSTS[COMPLLATCH]. 0 EPWMSYNCPER will not reset latch 1 EPWMSYNCPER will reset latch Reset type: SYSRSn
9	LLATCHCLR	R-0/W1S	0h	Low comparator latch software clear. Perform software reset of low comparator digital filter output latch COMPSTS[COMPLLATCH]. Reads always return 0. 0 No effect 1 Generate a single pulse of latch reset signal for COMPSTS[COMPLLATCH] Reset type: SYSRSn
8-3	RESERVED	R	0h	Reserved
2	HSYNCCLREN	R/W	0h	High comparator latch EPWMSYNCPER clear. Enable EPWMSYNCPER reset of high comparator digital filter output latch COMPSTS[COMPHLATCH]. 0 EPWMSYNCPER will not reset latch 1 EPWMSYNCPER will reset latch Reset type: SYSRSn
1	HLATCHCLR	R-0/W1S	0h	High comparator latch software clear. Perform software reset of high comparator digital filter output latch COMPSTS[COMPHLATCH]. Reads always return 0. 0 No effect 1 Generate a single pulse of latch reset signal for COMPSTS[COMPHLATCH] Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

22.8.2.5 COMPDACCTL Register (Offset = 4h) [Reset = 0h]

COMPDACCTL is shown in Figure 22-11 and described in Table 22-8.

Return to the Summary Table.

CMPSS DAC Control Register

Figure 22-11. COMPDACCTL Register

15	14	13	12	11	10	9	8
FREESOFT		RESERVED	BLANKEN	BLANKSOURCE			
R/W-0h		R-0h	R/W-0h	R/W-0h			
7	6	5	4	3	2	1	0
SWLOADSEL	RAMPLOADSEL	SELREF	RAMPSOURCE				DACSOURCE
R/W-0h	R/W-0h	R/W-0h	R/W-0h				R/W-0h

Table 22-8. COMPDACCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	FREESOFT	R/W	0h	Free-run or software-run emulation behavior. Behavior of the ramp generator during emulation suspend. 00b Ramp generator stops immediately during emulation suspend 01b Ramp generator completes current ramp and stops at next EPWMSYNCPER during emulation suspend 1Xb Ramp generator runs freely Reset type: SYSRSn
13	RESERVED	R	0h	Reserved
12	BLANKEN	R/W	0h	EPWMBLANK enable. This bit enables the EPWMBLANK signal. 0 EPWMBLANK signal is disabled. 1 EPWMBLANK signal is enabled. Reset type: SYSRSn
11-8	BLANKSOURCE	R/W	0h	EPWMBLANK source select. This bit field determines which EPWMnBLANK is passed on as the EPWMBLANK signal. Where n represents the maximum number of EPWMBLANK signals available on the device: 0 EPWM1BLANK 1 EPWM2BLANK 2 EPWM3BLANK ... n-1 EPWMnBLANK Reset type: SYSRSn
7	SWLOADSEL	R/W	0h	Software load select. Determines whether DACxVALA is updated from DACxVALS on SYSCLK or EPWMSYNCPER. 0 DACxVALA is updated from DACxVALS on SYSCLK 1 DACxVALA is updated from DACxVALS on EPWMSYNCPER Reset type: SYSRSn
6	RAMPLOADSEL	R/W	0h	Ramp load select. Determines whether RAMPSTS is updated from RAMPMAXREFA or RAMPMAXREFS when COMPSTS[COMPHSTS] is triggered. 0 RAMPSTS is loaded from RAMPMAXREFA 1 RAMPSTS is loaded from RAMPMAXREFS Reset type: SYSRSn
5	SELREF	R/W	0h	DAC reference select. Determines which voltage supply is used as the reference for the internal comparator DACs. 0 VDDA is the voltage reference for the DAC 1 VDAC is the voltage reference for the DAC Reset type: SYSRSn

Table 22-8. COMPDACCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-1	RAMPSOURCE	R/W	0h	EPWMSYNCPER source select. Determines which EPWMnSYNCPER signal is used within the CMPSS module. Where n represents the maximum number of EPWMSYNCPER signals available on the device: 0 EPWM1SYNCPER 1 EPWM2SYNCPER 2 EPWM3SYNCPER ... n-1 EPWMnSYNCPER Reset type: SYSRSn
0	DACSOURCE	R/W	0h	DAC source select. Determines whether DACHVALA is updated from DACHVALS or from the ramp generator. 0 DACHVALA is updated from DACHVALS 1 DACHVALA is updated from the ramp generator Reset type: SYSRSn

22.8.2.6 DACHVALS Register (Offset = 6h) [Reset = 0h]

DACHVALS is shown in Figure 22-12 and described in Table 22-9.

Return to the Summary Table.

CMPSS High DAC Value Shadow Register

Figure 22-12. DACHVALS Register

15	14	13	12	11	10	9	8
RESERVED				DACVAL			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
DACVAL							
R/W-0h							

Table 22-9. DACHVALS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DACVAL	R/W	0h	High DAC shadow value. When COMPDACCTL[DACSOURCE]=0, the value of DACHVALS is loaded into DACHVALA on the trigger signal selected by COMPDACCTL[SWLOADSEL]. Reset type: SYSRSn

22.8.2.7 DACHVALA Register (Offset = 7h) [Reset = 0h]

DACHVALA is shown in Figure 22-13 and described in Table 22-10.

Return to the Summary Table.

CMPSS High DAC Value Active Register

Figure 22-13. DACHVALA Register

15	14	13	12	11	10	9	8
RESERVED				DACVAL			
R-0h				R-0h			
7	6	5	4	3	2	1	0
DACVAL							
R-0h							

Table 22-10. DACHVALA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DACVAL	R	0h	High DAC active value. Value that is actively driven by the high DAC. Reset type: SYSRSn

22.8.2.8 RAMPMAXREFA Register (Offset = 8h) [Reset = 0h]

RAMPMAXREFA is shown in Figure 22-14 and described in Table 22-11.

Return to the Summary Table.

CMPSS Ramp Max Reference Active Register

Figure 22-14. RAMPMAXREFA Register

15	14	13	12	11	10	9	8
RAMPMAXREF							
R-0h							
7	6	5	4	3	2	1	0
RAMPMAXREF							
R-0h							

Table 22-11. RAMPMAXREFA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RAMPMAXREF	R	0h	Ramp maximum reference active value. Latched value to be loaded into ramp generator RAMPSTS. Reset type: SYSRSn

22.8.2.9 RAMPMAXREFS Register (Offset = Ah) [Reset = 0h]

RAMPMAXREFS is shown in Figure 22-15 and described in Table 22-12.

Return to the Summary Table.

CMPSS Ramp Max Reference Shadow Register

Figure 22-15. RAMPMAXREFS Register

15	14	13	12	11	10	9	8
RAMPMAXREF							
R/W-0h							
7	6	5	4	3	2	1	0
RAMPMAXREF							
R/W-0h							

Table 22-12. RAMPMAXREFS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RAMPMAXREF	R/W	0h	Ramp maximum reference shadow. Unlatched value to be loaded into ramp generator RAMPSTS. Reset type: SYSRSn

22.8.2.10 RAMPDECVALA Register (Offset = Ch) [Reset = 0h]

RAMPDECVALA is shown in Figure 22-16 and described in Table 22-13.

Return to the Summary Table.

CMPSS Ramp Decrement Value Active Register

Figure 22-16. RAMPDECVALA Register

15	14	13	12	11	10	9	8
RAMPDECVAL							
R-0h							
7	6	5	4	3	2	1	0
RAMPDECVAL							
R-0h							

Table 22-13. RAMPDECVALA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RAMPDECVAL	R	0h	Ramp decrement value active. Latched value that will be subtracted from RAMPSTS. Reset type: SYSRSn

22.8.2.11 RAMPDECVALS Register (Offset = Eh) [Reset = 0h]

RAMPDECVALS is shown in Figure 22-17 and described in Table 22-14.

Return to the Summary Table.

CMPSS Ramp Decrement Value Shadow Register

Figure 22-17. RAMPDECVALS Register

15	14	13	12	11	10	9	8
RAMPDECVAL							
R/W-0h							
7	6	5	4	3	2	1	0
RAMPDECVAL							
R/W-0h							

Table 22-14. RAMPDECVALS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RAMPDECVAL	R/W	0h	Ramp decrement value shadow. Unlatched value to be loaded into RAMPDECVALA. Reset type: SYSRSn

22.8.2.12 RAMPSTS Register (Offset = 10h) [Reset = 0h]

RAMPSTS is shown in Figure 22-18 and described in Table 22-15.

Return to the Summary Table.

CMPSS Ramp Status Register

Figure 22-18. RAMPSTS Register

15	14	13	12	11	10	9	8
RAMPVALUE							
R-0h							
7	6	5	4	3	2	1	0
RAMPVALUE							
R-0h							

Table 22-15. RAMPSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RAMPVALUE	R	0h	Ramp value. Present value of ramp generator. Reset type: SYSRSn

22.8.2.13 DACLVALS Register (Offset = 12h) [Reset = 0h]

DACLVALS is shown in Figure 22-19 and described in Table 22-16.

Return to the Summary Table.

CMPSS Low DAC Value Shadow Register

Figure 22-19. DACLVALS Register

15	14	13	12	11	10	9	8
RESERVED				DACVAL			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
DACVAL							
R/W-0h							

Table 22-16. DACLVALS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DACVAL	R/W	0h	Low DAC shadow value. value to be loaded into DACLVALA on the trigger signal selected by COMPDACCTL[SWLOADSEL]. Reset type: SYSRSn

22.8.2.14 DACLVALA Register (Offset = 13h) [Reset = 0h]

DACLVALA is shown in Figure 22-20 and described in Table 22-17.

Return to the Summary Table.

CMPSS Low DAC Value Active Register

Figure 22-20. DACLVALA Register

15	14	13	12	11	10	9	8
RESERVED				DACVAL			
R-0h				R-0h			
7	6	5	4	3	2	1	0
DACVAL							
R-0h							

Table 22-17. DACLVALA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DACVAL	R	0h	Low DAC active value. Value that is actively driven by the low DAC. Reset type: SYSRSn

22.8.2.15 RAMPDLYA Register (Offset = 14h) [Reset = 0h]

RAMPDLYA is shown in Figure 22-21 and described in Table 22-18.

Return to the Summary Table.

CMPSS Ramp Delay Active Register

Figure 22-21. RAMPDLYA Register

15	14	13	12	11	10	9	8
RESERVED			DELAY				
R-0h			R-0h				
7	6	5	4	3	2	1	0
DELAY							
R-0h							

Table 22-18. RAMPDLYA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-0	DELAY	R	0h	Ramp delay active value. Latched value of the number of cycles to delay the start of the ramp generator decrementer after a EPWMSYNCPER is received. Reset type: SYSRSn

22.8.2.16 RAMPDLYS Register (Offset = 15h) [Reset = 0h]

RAMPDLYS is shown in Figure 22-22 and described in Table 22-19.

Return to the Summary Table.

CMPSS Ramp Delay Shadow Register

Figure 22-22. RAMPDLYS Register

15	14	13	12	11	10	9	8
RESERVED			DELAY				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
DELAY							
R/W-0h							

Table 22-19. RAMPDLYS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-0	DELAY	R/W	0h	Ramp delay shadow value. Unlatched value to be loaded into RAMPDLYA. Reset type: SYSRSn

22.8.2.17 CTRIPLFILCTL Register (Offset = 16h) [Reset = 0h]

CTRIPLFILCTL is shown in Figure 22-23 and described in Table 22-20.

Return to the Summary Table.

CTRIPL Filter Control Register

Figure 22-23. CTRIPLFILCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 22-20. CTRIPLFILCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	Low filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	Low filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Threshold used is THRESH+1. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	Low filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

22.8.2.18 CTRIPLFILCLKCTL Register (Offset = 17h) [Reset = 0h]

CTRIPLFILCLKCTL is shown in Figure 22-24 and described in Table 22-21.

Return to the Summary Table.

CTRIPL Filter Clock Control Register

Figure 22-24. CTRIPLFILCLKCTL Register

15	14	13	12	11	10	9	8
RESERVED						CLKPRESCALE	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 22-21. CTRIPLFILCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	CLKPRESCALE	R/W	0h	Low filter sample clock prescale. Number of system clocks between samples is CLKPRESCALE+1. Reset type: SYSRSn

22.8.2.19 CTRIPHFILCTL Register (Offset = 18h) [Reset = 0h]

CTRIPHFILCTL is shown in Figure 22-25 and described in Table 22-22.

Return to the Summary Table.

CTRIPH Filter Control Register

Figure 22-25. CTRIPHFILCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 22-22. CTRIPHFILCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	High filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	High filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Threshold used is THRESH+1. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	High filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

22.8.2.20 CTRIPHFILCLKCTL Register (Offset = 19h) [Reset = 0h]

CTRIPHFILCLKCTL is shown in Figure 22-26 and described in Table 22-23.

Return to the Summary Table.

CTRIPH Filter Clock Control Register

Figure 22-26. CTRIPHFILCLKCTL Register

15	14	13	12	11	10	9	8
RESERVED						CLKPRESCALE	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 22-23. CTRIPHFILCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	CLKPRESCALE	R/W	0h	High filter sample clock prescale. Number of system clocks between samples is CLKPRESCALE+1. Reset type: SYSRSn

22.8.2.21 COMPLOCK Register (Offset = 1Ah) [Reset = 0h]

COMPLOCK is shown in Figure 22-27 and described in Table 22-24.

Return to the Summary Table.

CMPSS Lock Register

Figure 22-27. COMPLOCK Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	CTRIP	DACCTL	COMPHYSCTL	COMPCTL
R-0h			R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 22-24. COMPLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4	RESERVED	R/WSonce	0h	Reserved
3	CTRIP	R/WSonce	0h	Lock write-access to the CTRIPxFILTCTL and CTRIPxFILCLKCTL* registers. 0 CTRIPxFILCTL and CTRIPxFILCLKCTL* registers are not locked. Write 0 to this bit has no effect. 1 CTRIPxFILCTL and CTRIPxFILCLKCTL* registers are locked. Only a system reset can clear this bit. Reset type: SYSRSn
2	DACCTL	R/WSonce	0h	Lock write-access to the COMPDAC*CTL* register(s). 0 COMPDAC*CTL* register(s) not locked. Write 0 to this bit has no effect. 1 COMPDAC*CTL* register(s) locked. Only a system reset can clear this bit. Reset type: SYSRSn
1	COMPHYSCTL	R/WSonce	0h	Lock write-access to the COMPHYSCTL register. 0 COMPHYSCTL register is not locked. Write 0 to this bit has no effect. 1 COMPHYSCTL register is locked. Only a system reset can clear this bit. Reset type: SYSRSn
0	COMPCTL	R/WSonce	0h	Lock write-access to the COMPCTL register. 0 COMPCTL register is not locked. Write 0 to this bit has no effect. 1 COMPCTL register is locked. Only a system reset can clear this bit. Reset type: SYSRSn

22.8.3 CMPSS Registers to Driverlib Functions

Table 22-25. CMPSS Registers to Driverlib Functions

File	Driverlib Function
COMPCTL	
cmpss.h	CMPSS_enableModule
cmpss.h	CMPSS_disableModule
cmpss.h	CMPSS_configHighComparator
cmpss.h	CMPSS_configLowComparator
cmpss.h	CMPSS_configOutputsHigh
cmpss.h	CMPSS_configOutputsLow
COMPHYSCTL	

Table 22-25. CMPSS Registers to Driverlib Functions (continued)

File	Driverlib Function
cmpss.h	CMPSS_setHysteresis
COMPSTS	
cmpss.c	CMPSS_configLatchOnPWMSYNC
cmpss.h	CMPSS_getStatus
cmpss.h	CMPSS_clearFilterLatchHigh
cmpss.h	CMPSS_clearFilterLatchLow
cmpss.h	CMPSS_enableLatchResetOnPWMSYNCHigh
cmpss.h	CMPSS_disableLatchResetOnPWMSYNCHigh
cmpss.h	CMPSS_enableLatchResetOnPWMSYNCLow
cmpss.h	CMPSS_disableLatchResetOnPWMSYNCLow
COMPSTSCLR	
cmpss.c	CMPSS_configLatchOnPWMSYNC
cmpss.h	CMPSS_clearFilterLatchHigh
cmpss.h	CMPSS_clearFilterLatchLow
cmpss.h	CMPSS_enableLatchResetOnPWMSYNCHigh
cmpss.h	CMPSS_disableLatchResetOnPWMSYNCHigh
cmpss.h	CMPSS_enableLatchResetOnPWMSYNCLow
cmpss.h	CMPSS_disableLatchResetOnPWMSYNCLow
COMPDACCTL	
cmpss.c	CMPSS_configRamp
cmpss.h	CMPSS_configDAC
cmpss.h	CMPSS_configureSyncSource
cmpss.h	CMPSS_configBlanking
cmpss.h	CMPSS_enableBlanking
cmpss.h	CMPSS_disableBlanking
DACHVALS	
cmpss.h	CMPSS_setDACValueHigh
DACHVALA	
cmpss.h	CMPSS_getDACValueHigh
RAMPMAXREFA	
cmpss.h	CMPSS_getMaxRampValue
RAMPMAXREFS	
cmpss.c	CMPSS_configRamp
cmpss.h	CMPSS_setMaxRampValue
RAMPDECVALA	
cmpss.h	CMPSS_getRampDecValue
RAMPDECVALS	
cmpss.c	CMPSS_configRamp
cmpss.h	CMPSS_setRampDecValue
RAMPSTS	
-	
DACLVALS	
cmpss.h	CMPSS_setDACValueLow
DACLVALA	
cmpss.h	CMPSS_getDACValueLow

Table 22-25. CMPSS Registers to Driverlib Functions (continued)

File	Driverlib Function
RAMPDLYA	
cmpss.h	CMPSS_getRampDelayValue
RAMPDLYS	
cmpss.c	CMPSS_configRamp
cmpss.h	CMPSS_setRampDelayValue
CTRIPLFILCTL	
cmpss.c	CMPSS_configFilterLow
cmpss.h	CMPSS_initFilterLow
CTRIPLFILCLKCTL	
cmpss.c	CMPSS_configFilterLow
CTRIPHFILCTL	
cmpss.c	CMPSS_configFilterHigh
cmpss.h	CMPSS_initFilterHigh
CTRIPHFILCLKCTL	
cmpss.c	CMPSS_configFilterHigh
COMPLOCK	
-	

This page intentionally left blank.

Chapter 23

▶ CONTROL PERIPHERALS

The following chapters describe the control peripherals.

23.1 Technical Reference Manual Overview

The block diagram for the F2838x device is shown in Figure 23-1. This Technical Reference Manual is organized into five major sections:

- C28x SYSTEM RESOURCES

  These chapters describe the C28x CPU subsystem, C28x Boot ROM, device configuration, and other system peripherals.

- ANALOG PERIPHERALS

  These chapters describe the Analog-to-Digital Converter (ADC), Buffered Digital-to-Analog Converter (DAC), Comparator Subsystem (CMPSS), and general analog subsystem configuration.

- CONTROL PERIPHERALS

  These chapters describe the Enhanced Capture (eCAP), High Resolution Capture (HRCAP), Enhanced Pulse Width Modulator (ePWM), Enhanced Quadrature Encoder Pulse (eQEP), and Sigma Delta Filter Module (SDFM) peripherals.

- COMMUNICATION PERIPHERALS

  These chapters describe the communication peripherals available to the C28x subsystem such as the I2C, SCI, FSI, McBSP, PMBUS, and SPI. The CAN, EtherCAT, and USB peripherals are also described in this section and can be assigned to the CM subsystem.

- CONNECTIVITY MANAGER (CM)

  These chapters describe the Connectivity Manager (CM) subsystem as well as the AES, GCRC, CM-I2C, CM-UART, SSI, and Ethernet peripherals.

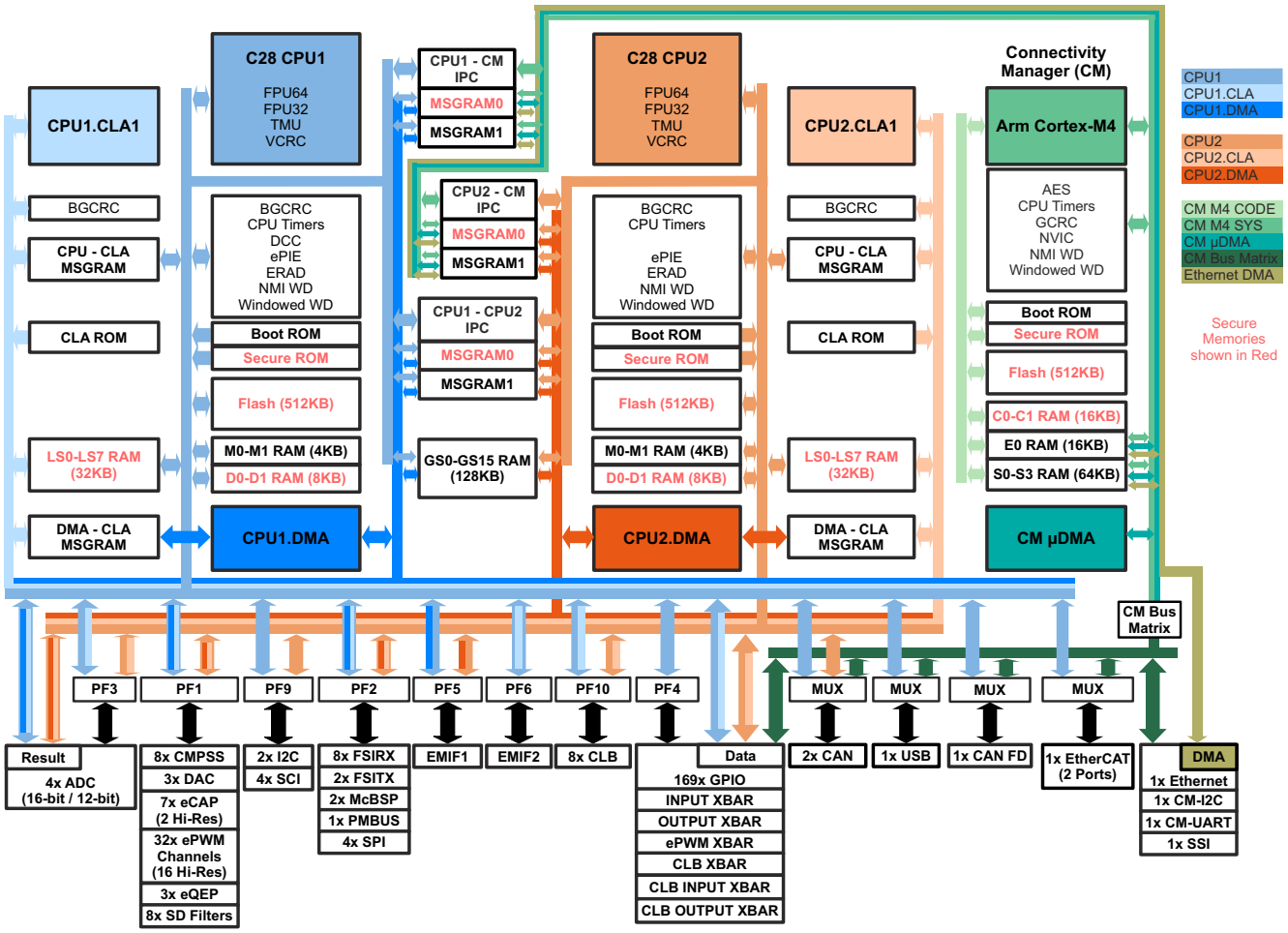


Figure 23-1. F2838x Block Diagram

Chapter 24

Enhanced Capture (eCAP)

This chapter describes the enhanced capture (eCAP) module, which is used in systems where accurate timing of external events is important.

The enhanced capture (eCAP) module is a Type 2 eCAP. See the *C2000 Real-Time Control Peripheral Reference Guide* for a list of all devices with an eCAP module of the same type, to determine the differences between the types, and for a list of device-specific differences within a type.

24.1 Introduction

24.1.1 Features

The features of the eCAP module include:

- Speed measurements of rotating machinery (for example, toothed sprockets sensed by way of Hall sensors)
- Elapsed time measurements between position sensor pulses
- Period and duty cycle measurements of pulse train signals
- Decoding current or voltage amplitude derived from duty cycle encoded current/voltage sensors

The eCAP module features described in this chapter include:

- 4-event time-stamp registers (each 32 bits)
- Edge polarity selection for up to four sequenced time-stamp capture events
- Interrupt on either of the four events
- Single-shot capture of up to four event time-stamps
- Continuous mode capture of time stamps in a four-deep circular buffer
- Absolute time-stamp capture
- Difference (Delta) mode time-stamp capture
- When not used in capture mode, the eCAP module can be configured as a single-channel PWM output

The capture functionality of the Type 1 eCAP is enhanced from the Type 0 eCAP with the following added features:

- Event filter reset bit
  - Writing a 1 to ECCTL2[CTRFILTRESET] clears the event filter, the modulo counter, and any pending interrupts flags. Resetting the bit is useful for initialization and debug.
- Modulo counter status bits
  - The modulo counter (ECCTL2 [MODCNTRSTS]) indicates which capture register is loaded next. In the Type 0 eCAP, to know the current state of the modulo counter was not possible
- DMA trigger source
  - eCAPxDMA was added as a DMA trigger. CEVT[1-4] can be configured as the source for eCAPxDMA.
- Input multiplexer
  - ECCTL0 [INPUTSEL] selects one of 128 input signals, which are detailed in Section 24.3.
- EALLOW protection
  - EALLOW protection was added to critical registers. To maintain software compatibility with Type-0, configure DEV_CFG_REGS.ECAPTYPE to make these registers unprotected.

The capture functionality of the Type 2 eCAP is enhanced from the Type 1 eCAP with the following added features:

- Added ECAPxSYNCINSEL register
  - ECAPxSYNCINSEL register is added for each eCAP to select an external SYNCIN. Every eCAP can have a separate SYNCIN signal.

24.1.2 ECAP Related Collateral

Foundational Materials

- C2000 Academy - ECAP

Getting Started Materials

- *Leveraging High Resolution Capture (HRCAP) for Single Wire Data Transfer Application Report*

24.2 Description

The eCAP module represents one complete capture channel that can be instantiated multiple times, depending on the target device. In the context of this guide, one eCAP channel has the following independent key resources:

- Capture inputs can be connected using the Input X-BAR
- 128:1 input multiplexer
- Output X-BAR is used to configure output in APWM mode
- 32-bit time base (counter)
- 4 x 32-bit time-stamp capture registers (CAP1-CAP4)
- Four-stage sequencer (modulo4 counter) that is synchronized to external events, eCAP pin rising/falling edges.
- Modulo counter status register (MODCNTRSTS) to indicate sequencer state
- Independent edge polarity (rising/falling edge) selection for all four events
- Input capture signal prescaling (from 2-62 or bypass)
- One-shot compare register (two bits) to freeze captures after 1-4 time-stamp events
- Control for continuous time-stamp captures using a four-deep circular buffer (CAP1-CAP4) scheme
- Ability to reset event filter, modulo counter, and interrupt flags
- Interrupt capabilities on any of the four capture events
- Separate DMA trigger
- EALLOW protection to control registers

24.3 Configuring Device Pins for the eCAP

The Input X-BAR connects the device pins to the module as input. Any GPIO on the device can be configured as an input. The GPIO input qualification can be set to synchronous or asynchronous mode by setting the GPxQSELn register bits. Using synchronized inputs can help with noise immunity but affects the eCAP accuracy by ±2 cycles. The internal pull-ups can be configured in the GPyPUD register. Since the GPIO mode is used, the GPyINV register can invert the signals.

New to the Type 1 eCAP module, a 128:1 input multiplexer must also be configured (see Figure 24-3). This multiplexer can select a variety of inputs detailed in Table 24-1 by configuring ECCTL0.INPUTSEL.

Table 24-1. eCAP Input Selection

Selection of ECAP Input	ECAP1 INDEX	ECAP2 INDEX	ECAP3 INDEX	ECAP4 INDEX	ECAP5 INDEX	ECAP6 INDEX	ECAP7 INDEX
INPUTXBAR1	0	0	0	0	0	0	0
INPUTXBAR2	1	1	1	1	1	1	1
INPUTXBAR3	2	2	2	2	2	2	2
INPUTXBAR4	3	3	3	3	3	3	3
INPUTXBAR5	4	4	4	4	4	4	4
INPUTXBAR6	5	5	5	5	5	5	5
INPUTXBAR7	6	6	6	6	6	6	6
INPUTXBAR8	7	7	7	7	7	7	7
INPUTXBAR9	8	8	8	8	8	8	8
INPUTXBAR10	9	9	9	9	9	9	9
INPUTXBAR11	10	10	10	10	10	10	10
INPUTXBAR12	11	11	11	11	11	11	11
INPUTXBAR13	12	12	12	12	12	12	12
INPUTXBAR14	13	13	13	13	13	13	13

Table 24-1. eCAP Input Selection (continued)

Selection of ECAP Input	ECAP1 INDEX	ECAP2 INDEX	ECAP3 INDEX	ECAP4 INDEX	ECAP5 INDEX	ECAP6 INDEX	ECAP7 INDEX
INPUTXBAR15	14	14	14	14	14	14	14
INPUTXBAR16	15	15	15	15	15	15	15
CLB1_OUT14	16	16	Reserved	Reserved	Reserved	Reserved	Reserved
CLB1_OUT15	17	17	Reserved	Reserved	Reserved	Reserved	Reserved
CLB2_OUT14	Reserved	Reserved	16	16	16	Reserved	Reserved
CLB2_OUT15	Reserved	Reserved	17	17	17	Reserved	Reserved
CLB3_OUT14	Reserved	Reserved	Reserved	Reserved	Reserved	16	16
CLB3_OUT15	Reserved	Reserved	Reserved	Reserved	Reserved	17	17
CLB4_OUT14	Reserved	Reserved	Reserved	Reserved	Reserved	18	18
CLB4_OUT15	Reserved	Reserved	Reserved	Reserved	Reserved	19	19
CLB5_OUT14	18	18	Reserved	Reserved	Reserved	Reserved	Reserved
CLB5_OUT15	19	19	Reserved	Reserved	Reserved	Reserved	Reserved
CLB6_OUT14	Reserved	Reserved	18	Reserved	Reserved	Reserved	Reserved
CLB6_OUT15	Reserved	Reserved	19	Reserved	Reserved	Reserved	Reserved
CLB7_OUT14	Reserved	Reserved	Reserved	18	Reserved	Reserved	Reserved
CLB7_OUT15	Reserved	Reserved	Reserved	19	Reserved	Reserved	Reserved
CLB8_OUT14	Reserved	Reserved	Reserved	Reserved	18	Reserved	Reserved
CLB8_OUT15	Reserved	Reserved	Reserved	Reserved	19	Reserved	Reserved
CANA_INT0	20	20	20	20	20	20	20
CANB_INT0	21	21	21	21	21	21	21
Reserved	22	22	22	22	22	22	22
ECAP6_DELAY_CLK	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	23
ECAP7_DELAY_CLK	Reserved	Reserved	Reserved	Reserved	Reserved	23	Reserved
Reserved	23	23	23	23	23	Reserved	Reserved
OUTPUTXBAR1	24	24	24	24	24	24	24
OUTPUTXBAR2	25	25	25	25	25	25	25
OUTPUTXBAR3	26	26	26	26	26	26	26
OUTPUTXBAR4	27	27	27	27	27	27	27
OUTPUTXBAR5	28	28	28	28	28	28	28
OUTPUTXBAR6	29	29	29	29	29	29	29
OUTPUTXBAR7	30	30	30	30	30	30	30
OUTPUTXBAR8	31	31	31	31	31	31	31
ADCDEVT1	32	32	32	32	32	32	32
ADCDEVT2	33	33	33	33	33	33	33
ADCDEVT3	34	34	34	34	34	34	34
ADCDEVT4	35	35	35	35	35	35	35
ADCCEVT1	36	36	36	36	36	36	36
ADCCEVT2	37	37	37	37	37	37	37

Table 24-1. eCAP Input Selection (continued)

Selection of ECAP Input	ECAP1 INDEX	ECAP2 INDEX	ECAP3 INDEX	ECAP4 INDEX	ECAP5 INDEX	ECAP6 INDEX	ECAP7 INDEX
ADCCEVT3	38	38	38	38	38	38	38
ADCCEVT4	39	39	39	39	39	39	39
ADCBEVT1	40	40	40	40	40	40	40
ADCBEVT2	41	41	41	41	41	41	41
ADCBEVT3	42	42	42	42	42	42	42
ADCBEVT4	43	43	43	43	43	43	43
ADCAEVT1	44	44	44	44	44	44	44
ADCAEVT2	45	45	45	45	45	45	45
ADCAEVT3	46	46	46	46	46	46	46
ADCAEVT4	47	47	47	47	47	47	47
FSIRXA_MEASURE	48	48	48	48	48	48	48
FSIRXA_MEASURE_RISE	49	49	49	49	49	49	49
FSIRXA_MEASURE_FALL	50	50	50	50	50	50	50
FSIRXB_MEASURE	51	51	51	51	51	51	51
FSIRXB_MEASURE_RISE	52	52	52	52	52	52	52
FSIRXB_MEASURE_FALL	53	53	53	53	53	53	53
FSIRXC_MEASURE	54	54	54	54	54	54	54
FSIRXC_MEASURE_RISE	55	55	55	55	55	55	55
FSIRXC_MEASURE_FALL	56	56	56	56	56	56	56
FSIRXD_MEASURE	57	57	57	57	57	57	57
FSIRXD_MEASURE_RISE	58	58	58	58	58	58	58
FSIRXD_MEASURE_FALL	59	59	59	59	59	59	59
SD2FLT1_COMPL	60	60	60	60	60	60	60
SD2FLT2_COMPL	61	61	61	61	61	61	61
SD2FLT3_COMPL	62	62	62	62	62	62	62
SD2FLT4_COMPL	63	63	63	63	63	63	63
SD1FLT1_COMPL	64	64	64	64	64	64	64
SD1FLT2_COMPL	65	65	65	65	65	65	65
SD1FLT3_COMPL	66	66	66	66	66	66	66
SD1FLT4_COMPL	67	67	67	67	67	67	67
FSIRXE_MEASURE	68	68	68	68	68	68	68
FSIRXE_MEASURE_RISE	69	69	69	69	69	69	69
FSIRXE_MEASURE_FALL	70	70	70	70	70	70	70
FSIRXF_MEASURE	71	71	71	71	71	71	71
FSIRXF_MEASURE_RISE	72	72	72	72	72	72	72
FSIRXF_MEASURE_FALL	73	73	73	73	73	73	73
ECAT_SYNC0	74	74	74	74	74	74	74
ECAT_SYNC1	75	75	75	75	75	75	75

Table 24-1. eCAP Input Selection (continued)

Selection of ECAP Input	ECAP1 INDEX	ECAP2 INDEX	ECAP3 INDEX	ECAP4 INDEX	ECAP5 INDEX	ECAP6 INDEX	ECAP7 INDEX
SD2FLT1_COMPH	76	76	76	76	76	76	76
SD2FLT2_COMPH	77	77	77	77	77	77	77
SD2FLT3_COMPH	78	78	78	78	78	78	78
SD2FLT4_COMPH	79	79	79	79	79	79	79
SD1FLT1_COMPH	80	80	80	80	80	80	80
SD1FLT2_COMPH	81	81	81	81	81	81	81
SD1FLT3_COMPH	82	82	82	82	82	82	82
SD1FLT4_COMPH	83	83	83	83	83	83	83
SD2FLT1_COMPH_OR_COMPL	84	84	84	84	84	84	84
SD2FLT2_COMPH_OR_COMPL	85	85	85	85	85	85	85
SD2FLT3_COMPH_OR_COMPL	86	86	86	86	86	86	86
SD2FLT4_COMPH_OR_COMPL	87	87	87	87	87	87	87
SD1FLT1_COMPH_OR_COMPL	88	88	88	88	88	88	88
SD1FLT2_COMPH_OR_COMPL	89	89	89	89	89	89	89
SD1FLT3_COMPH_OR_COMPL	90	90	90	90	90	90	90
SD1FLT4_COMPH_OR_COMPL	91	91	91	91	91	91	91
FSIRXG_MEASURE	92	92	92	92	92	92	92
FSIRXG_MEASURE_RISE	93	93	93	93	93	93	93
FSIRXG_MEASURE_FALL	94	94	94	94	94	94	94
Reserved	95	95	95	95	95	95	95
CMPSS1_CTRIPL	96	96	96	96	96	96	96
CMPSS2_CTRIPL	97	97	97	97	97	97	97
CMPSS3_CTRIPL	98	98	98	98	98	98	98
CMPSS4_CTRIPL	99	99	99	99	99	99	99
CMPSS5_CTRIPL	100	100	100	100	100	100	100
CMPSS6_CTRIPL	101	101	101	101	101	101	101
CMPSS7_CTRIPL	102	102	102	102	102	102	102
FSIRXH_MEASURE	103	103	103	103	103	103	103
FSIRXH_MEASURE_RISE	104	104	104	104	104	104	104
FSIRXH_MEASURE_FALL	105	105	105	105	105	105	105
Reserved	106-107	106-107	106-107	106-107	106-107	106-107	106-107
CMPSS1_CTRIPH	108	108	108	108	108	108	108

Table 24-1. eCAP Input Selection (continued)

Selection of ECAP Input	ECAP1 INDEX	ECAP2 INDEX	ECAP3 INDEX	ECAP4 INDEX	ECAP5 INDEX	ECAP6 INDEX	ECAP7 INDEX
CMPSS2_CTRIPH	109	109	109	109	109	109	109
CMPSS3_CTRIPH	110	110	110	110	110	110	110
CMPSS4_CTRIPH	111	111	111	111	111	111	111
CMPSS5_CTRIPH	112	112	112	112	112	112	112
CMPSS6_CTRIPH	113	113	113	113	113	113	113
CMPSS7_CTRIPH	114	114	114	114	114	114	114
GPIO8	115	115	115	115	115	115	115
GPIO9	116	116	116	116	116	116	116
GPIO22	117	117	117	117	117	117	117
GPIO23	118	118	118	118	118	118	118
Reserved	119	119	119	119	119	119	119
CMPSS1_CTRIPH_OR_CTRIPL	120	120	120	120	120	120	120
CMPSS2_CTRIPH_OR_CTRIPL	121	121	121	121	121	121	121
CMPSS3_CTRIPH_OR_CTRIPL	122	122	122	122	122	122	122
CMPSS4_CTRIPH_OR_CTRIPL	123	123	123	123	123	123	123
CMPSS5_CTRIPH_OR_CTRIPL	124	124	124	124	124	124	124
CMPSS6_CTRIPH_OR_CTRIPL	125	125	125	125	125	125	125
CMPSS7_CTRIPH_OR_CTRIPL	126	126	126	126	126	126	126
INPUTXBAR7	127	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
INPUTXBAR8	Reserved	127	Reserved	Reserved	Reserved	Reserved	Reserved
INPUTXBAR9	Reserved	Reserved	127	Reserved	Reserved	Reserved	Reserved
INPUTXBAR10	Reserved	Reserved	Reserved	127	Reserved	Reserved	Reserved
INPUTXBAR11	Reserved	Reserved	Reserved	Reserved	127	Reserved	Reserved
INPUTXBAR12	Reserved	Reserved	Reserved	Reserved	Reserved	127	Reserved
INPUTXBAR13	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	127

The Output X-BAR must be used to connect output signals to the OUTPUTXBARx output locations. The GPIO mux must then be configured to connect the OUTPUTXBARx lines to any of several IO pins with the GPIO mux. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

See the *General-Purpose Input/Output (GPIO)* chapter for more details on GPIO mux, GPIO settings, and XBAR configuration.

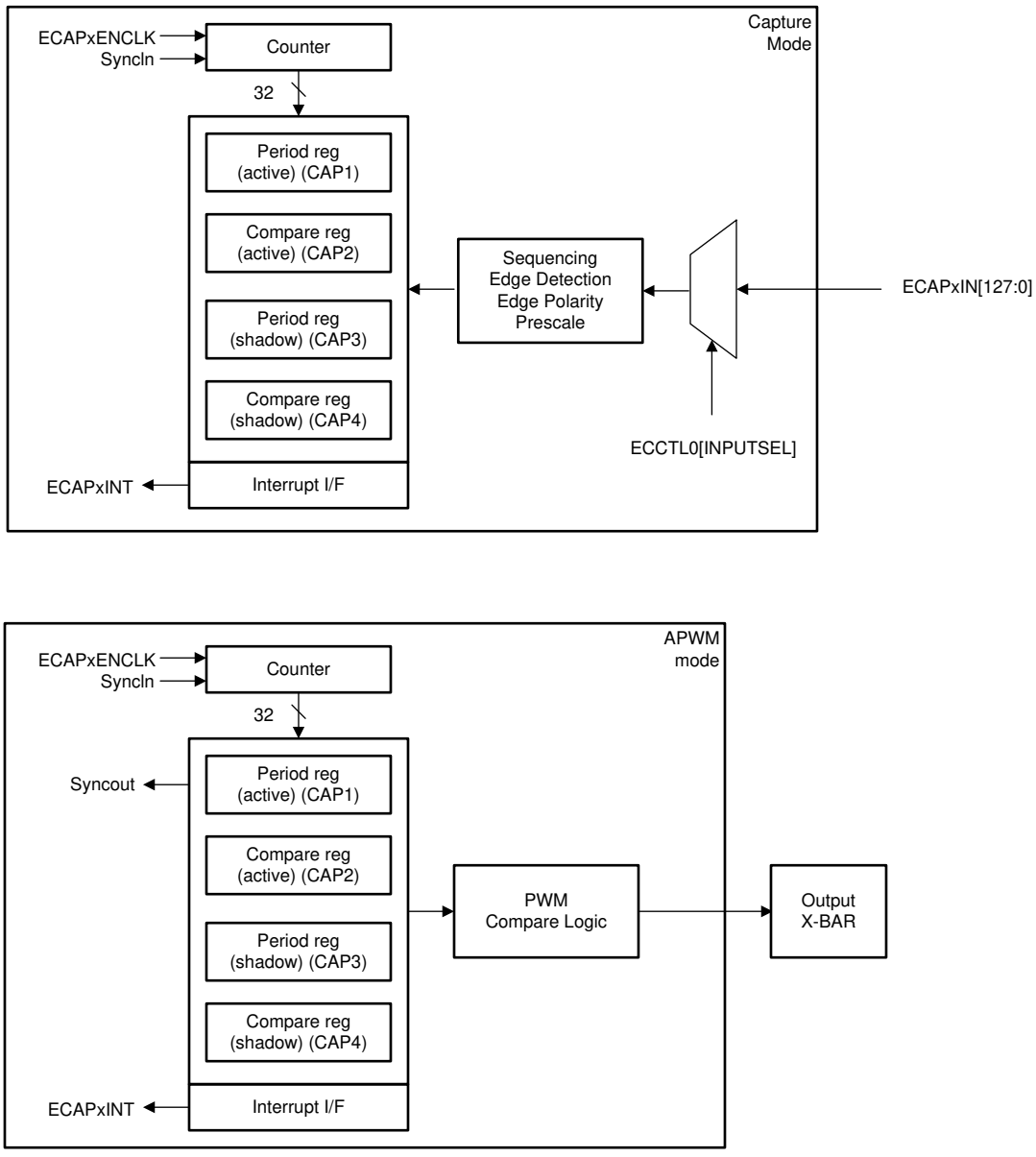
Note

ECAPxIN has to be at least 2 × SYSCLK-cycles wide to be properly captured by the eCAP module; otherwise, the input pulse can get missed from sampling by the SYSCLK.

24.4 Capture and APWM Operating Mode

Use the eCAP module resources to implement a single-channel PWM generator (with 32-bit capabilities) when the eCAP module is not being used for input captures. The counter operates in count-up mode, providing a time-base for asymmetrical pulse width modulation (PWM) waveforms. The CAP1 and CAP2 registers become the active period and compare registers, respectively, while CAP3 and CAP4 registers become the period and compare shadow registers, respectively. Figure 24-1 is a high-level view of both the capture and auxiliary pulse-width modulator (APWM) modes of operation.

Figure 24-2 further descries the output of the eCAP in APWM mode based on the CMP and PRD values.



A. A single pin is shared between CAP and APWM functions. In capture mode, the pin is an input; in APWM mode, the pin is an output.

B. In APWM mode, writing any value to CAP1/CAP2 active registers also writes the same value to the corresponding shadow registers CAP3/CAP4. This emulates immediate mode. Writing to the shadow registers CAP3/CAP4 invokes the shadow mode.

Figure 24-1. Capture and APWM Modes of Operation

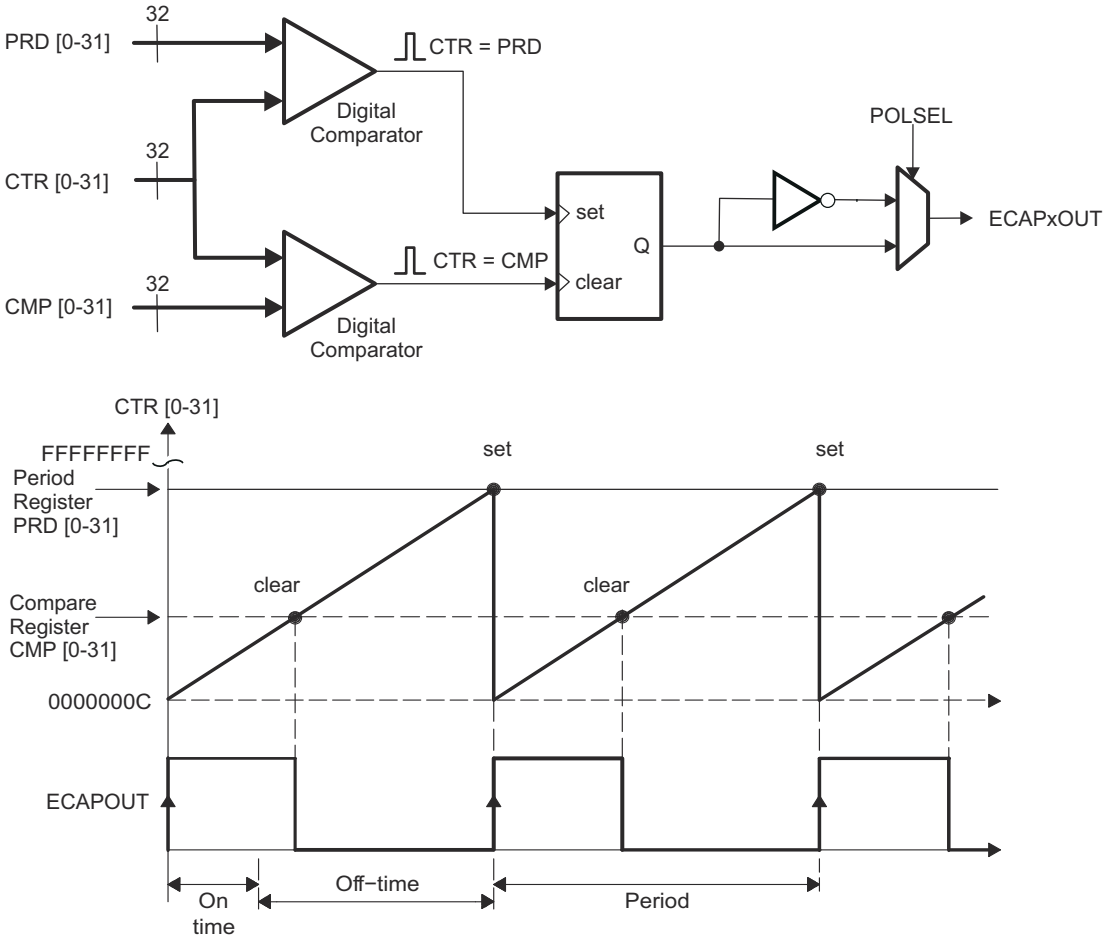
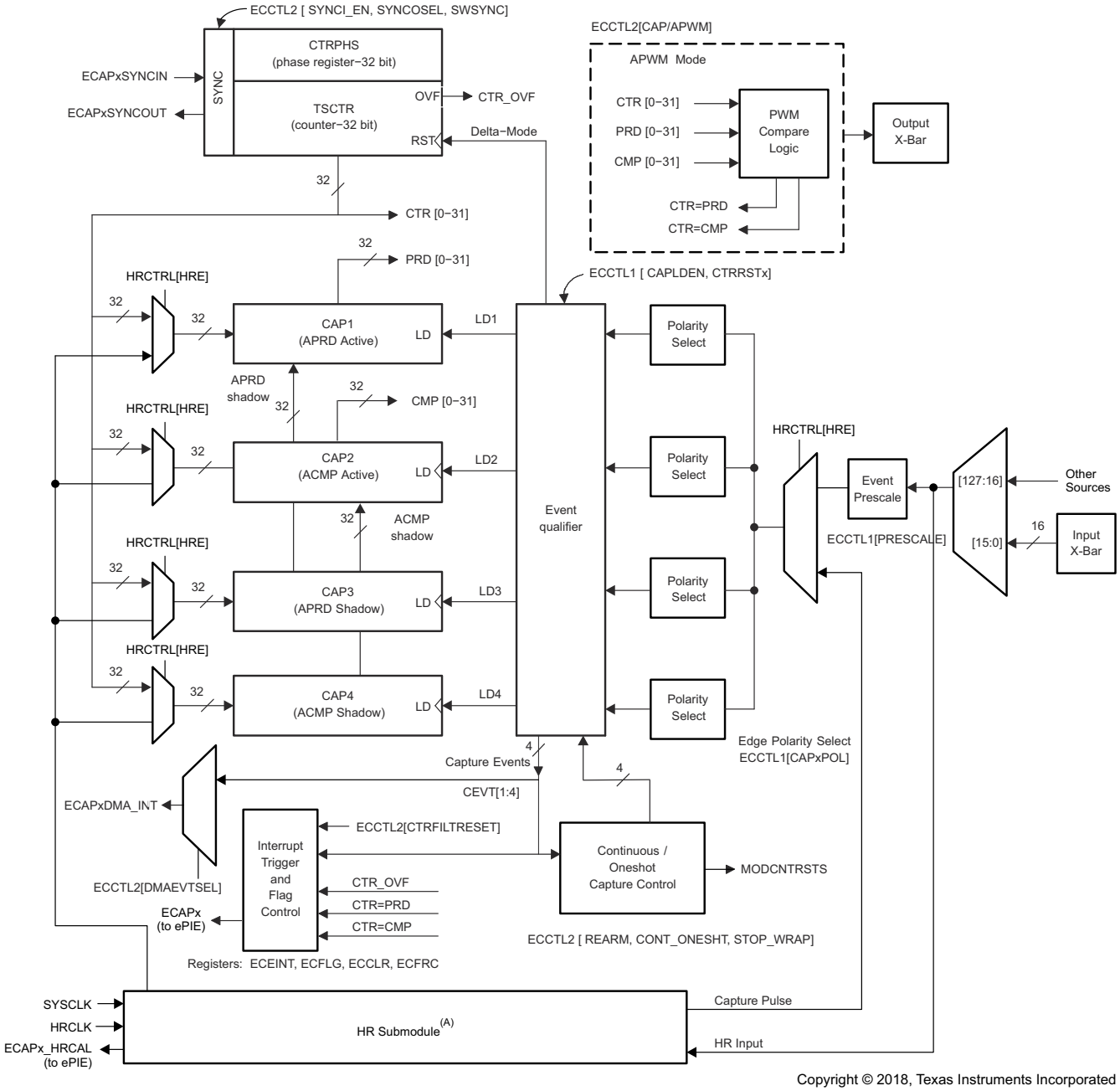


Figure 24-2. Counter Compare and PRD Effects on the eCAP Output in APWM Mode

24.5 Capture Mode Description

Figure 24-3 shows the various components that implement the capture function.

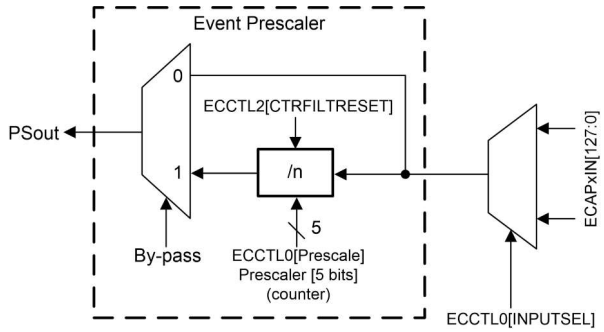


A. The HRCAP submodule is not available on all eCAP modules; in this case, the high-resolution muxes and hardware are not implemented.

Figure 24-3. eCAP Block Diagram

24.5.1 Event Prescaler

An input capture signal (pulse train) can be prescaled by N = 2-62 (in multiples of 2) or can bypass the prescaler. This is useful when very high frequency signals are used as inputs. Figure 24-4 shows a functional diagram and Figure 24-5 shows the operation of the prescale function. The event prescaler can be reset by setting the ECCTL2.CTRFILTRESET register bit.



A. When a prescale value of 1 is chosen (ECCTL1[13:9] = 0,0,0,0,0), the input capture signal bypasses the prescale logic completely.
B. The first Rise edge after Prescale configuration change is not passed to Capture logic, prescaler value takes into effect on the second rising edge after the configuration.

Figure 24-4. Event Prescale Control

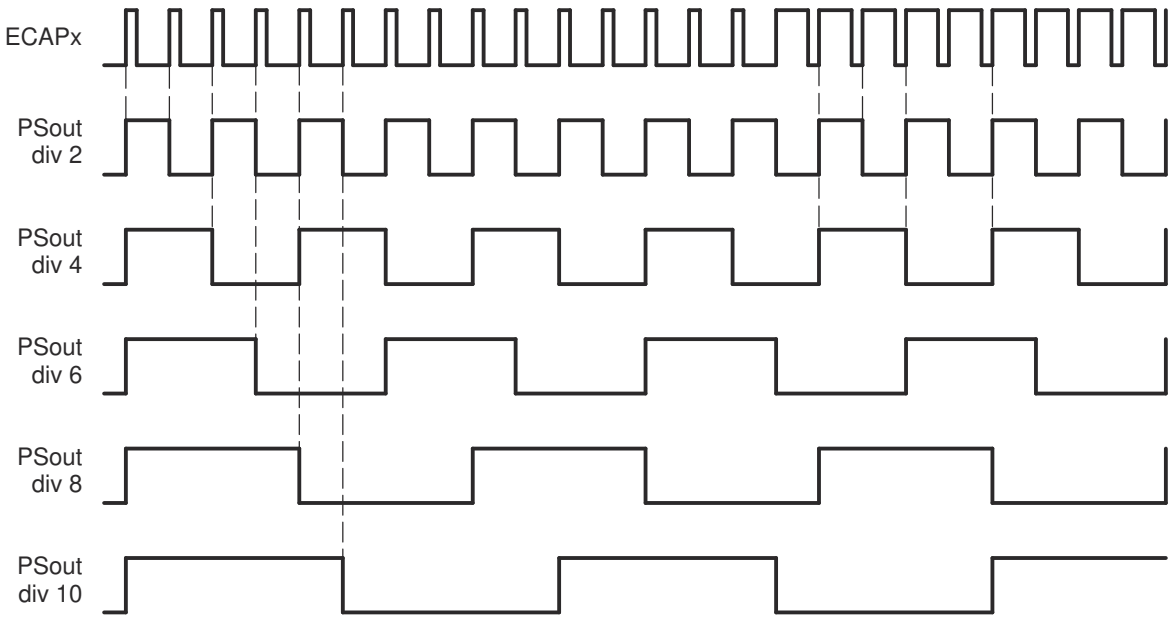


Figure 24-5. Prescale Function Waveforms

24.5.2 Edge Polarity Select and Qualifier

Functionality and features include:

- Four independent edge polarity (rising edge/falling edge) selection muxes are used, one for each capture event.
- Each edge (up to 4) is event qualified by the Modulo4 sequencer.
- The edge event is gated to the respective CAPx register by the Mod4 counter. The CAPx register is loaded on the falling edge.

24.5.3 Continuous/One-Shot Control

Operation of eCAP in Continuous/One-Shot mode:

- The Mod4 (2-bit) counter is incremented using edge qualified events (CEVT1-CEVT4).
- The Mod4 counter continues counting (0->1->2->3->0) and wraps around unless stopped.
- During one-shot operation, a 2-bit stop register (STOP_WRAP) is used to compare the Mod4 counter output, and when equal, stops the Mod4 counter and inhibits further loads of the CAP1-CAP4 registers. In this mode, if TSCCTR counter is configured to reset on capture event (CEVTx) by configuring ECCTL1.CTRRSTx bit, the operation still keeps resetting the TSCCTR counter on capture event (CEVTx) after the STOP_WRAP value is reached and re-arm (REARM) has not occurred.

The continuous/one-shot block controls the start, stop and reset (zero) functions of the Mod4 counter, using a mono-shot type of action that can be triggered by the stop-value comparator and re-armed using software control.

Once armed, the eCAP module waits for 1-4 (defined by stop-value) capture events before freezing both the Mod4 counter and contents of CAP1-4 registers (time stamps).

Re-arming prepares the eCAP module for another capture sequence. Also, re-arming clears (to zero) the Mod4 counter and permits loading of CAP1-4 registers again, providing the CAPLDEN bit is set.

In continuous mode, the Mod4 counter continues to run (0->1->2->3->0, the one-shot action is ignored, and capture values continue to be written to CAP1-4 in a circular buffer sequence.

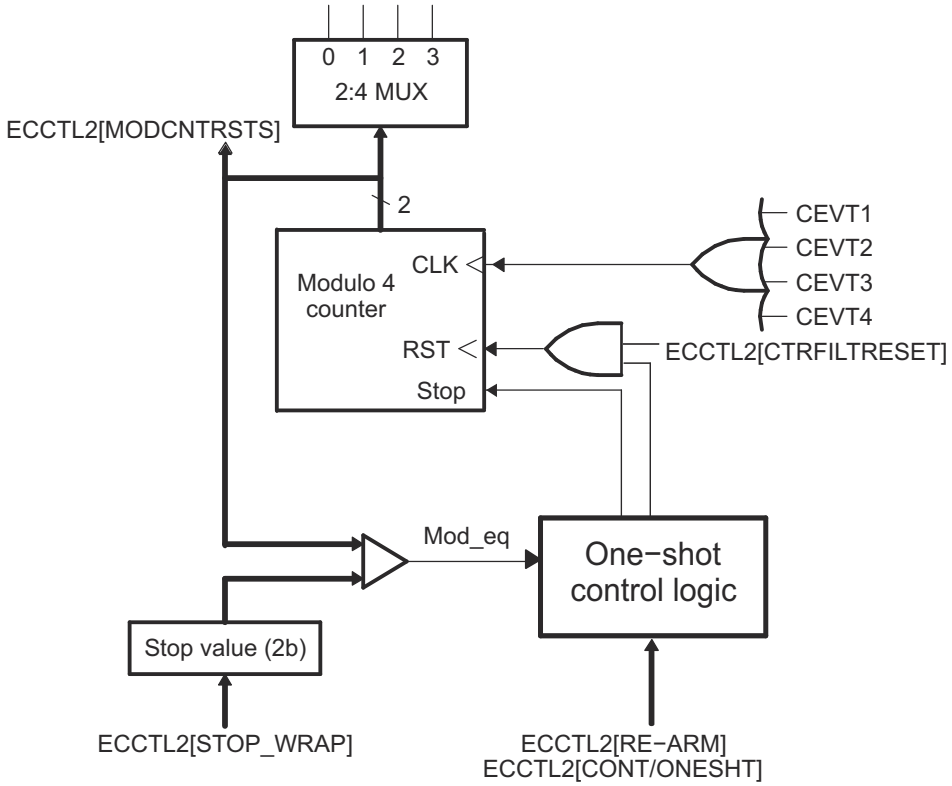


Figure 24-6. Details of the Continuous/One-shot Block

24.5.4 32-Bit Counter and Phase Control

This counter provides the time-base for event captures, and is clocked using the system clock.

A phase register is provided to achieve synchronization with other counters using a hardware and software forced sync. This is useful in APWM mode when a phase offset between modules is needed.

On any of the four event loads, an option to reset the 32-bit counter is given. This is useful for time difference capture. The 32-bit counter value is captured first, then the counter value is reset to 0 by any of the LD1-LD4 signals.

24.5.5 CAP1-CAP4 Registers

These 32-bit registers are supplied by the 32-bit counter timer bus, CTR[0-31], and are loaded (capture a time-stamp) when their respective LD inputs are strobed.

Control bit CAPLDEN can inhibit loading of the capture registers. During one-shot operation, this bit is cleared (loading is inhibited) automatically when a stop condition occurs, StopValue = Mod4.

CAP1 and CAP2 registers become the active period and compare registers, respectively, in APWM mode.

CAP3 and CAP4 registers become the respective shadow registers (APRD and ACMP) for CAP1 and CAP2 during APWM operation.

24.5.6 eCAP Synchronization

eCAP modules can be synchronized with each other by selecting a common SYNCIN source. SYNCIN source for eCAP can be either software sync-in or external sync-in. The external sync-in signal can come from EtherCat, eCAP, X-Bar or EPWM. The SWSYNC of the eCAP module is logical ORed with the SYNC signal as shown in Figure 24-7. The SYNC signal is defined by the selection of ECAPxSYNCINSEL[SEL] as shown in Figure 24-8.

Note

If SWSYNC is forced from ECAPx, then the Sync pulse is delayed by one clock cycle for sync pulse receiving ECAPs . This cycle delay/lag for the sync pulse receiving ECAPs can be corrected with the CTRPHS register. If there is an external SYNC signal, then all ECAPs see the signal at the same time.

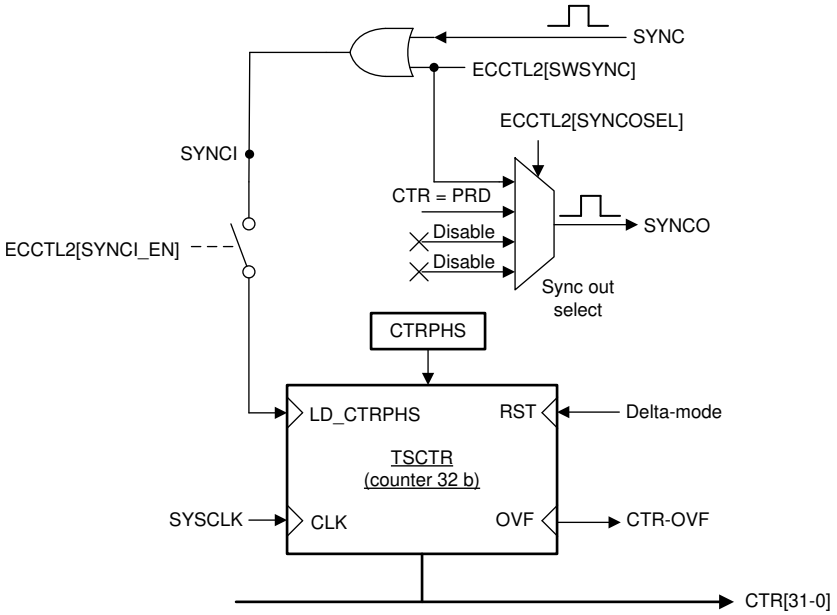


Figure 24-7. Details of the Counter and Synchronization Block

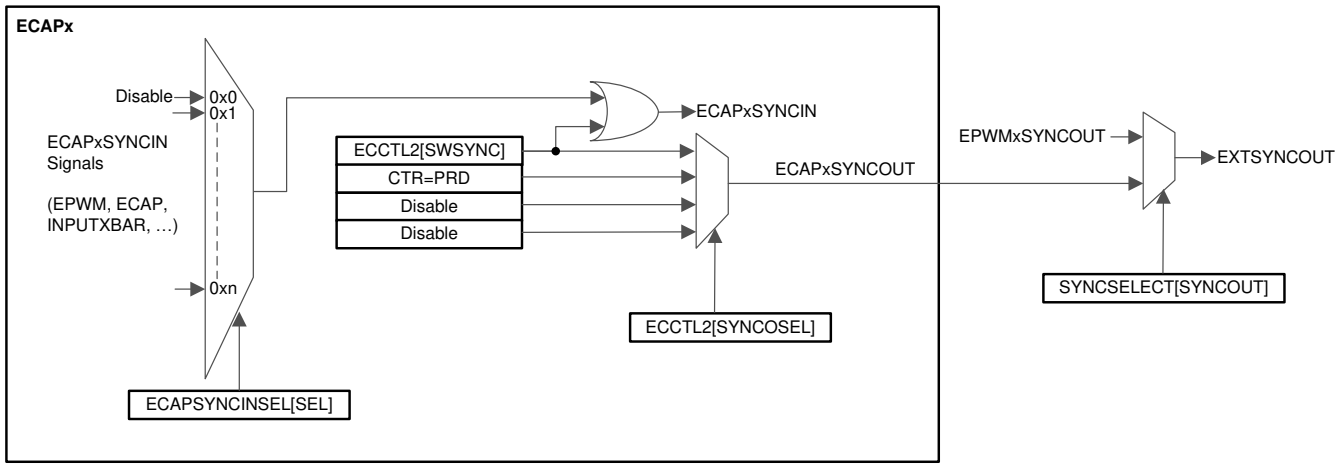


Figure 24-8. eCAP Synchronization Scheme

24.5.6.1 Example 1 - Using SWSYNC with ECAP Module

Implement the following steps to use SWSYNC with ECAP1 and ECAP2.

- Configure ECAP[1..2].ECAPSYNCINSEL.SEL = 0x0 to disable external SYNCIN coming to eCAP1.
- Configure ECAP[1..2].ECCTL2.SWSYNC = 0x1, to force Software Synchronization of the TSCTR counter.

To use SWSYNC with other eCAP modules, make sure that the previous eCAP chain is not generating a SYNCOUT signal that interferes with the software synchronization.

24.5.7 Interrupt Control

Operation and features of the eCAP interrupt control include (see Figure 24-9):

- An interrupt can be generated on capture events (CEVT1-CEVT4, CTROVF) or APWM events (CTR = PRD, CTR = CMP).
- A counter overflow event (FFFFFFFF->00000000) is also provided as an interrupt source (CTROVF).
- The capture events are edge and sequencer-qualified (ordered in time) by the polarity select and Mod4 gating, respectively.
- One of these events can be selected as the interrupt source (from the eCAPx module) going to the PIE and CLA.
- Seven interrupt events (CEVT1, CEVT2, CEVT3, CEVT4, CNTOVF, CTR=PRD, CTR=CMP) can be generated.
- The interrupt enable register (ECEINT) is used to enable/disable individual interrupt event sources. The interrupt flag register (ECFLG) indicates if any interrupt event has been latched and contains the global interrupt flag bit (INT). An interrupt pulse is generated to the PIE only if any of the interrupt events are enabled, the flag bit is 1, and the INT flag bit is 0. The interrupt service routine must clear the global interrupt flag bit and the serviced event using the interrupt clear register (ECCLR) before any other interrupt pulses are generated. All interrupt flags are cleared upon an event filter reset by writing a 1 to ECCTL2[CLRFILTRESET]. To force an interrupt event, use the interrupt force register (ECFRC). This is useful for test purposes.

Note

The CEVT1, CEVT2, CEVT3, CEVT4 flags are only active in capture mode (ECCTL2[CAP/APWM == 0]). The CTR=PRD, CTR=CMP flags are only valid in APWM mode (ECCTL2[CAP/APWM == 1]). CNTOVF flag is valid in both modes.

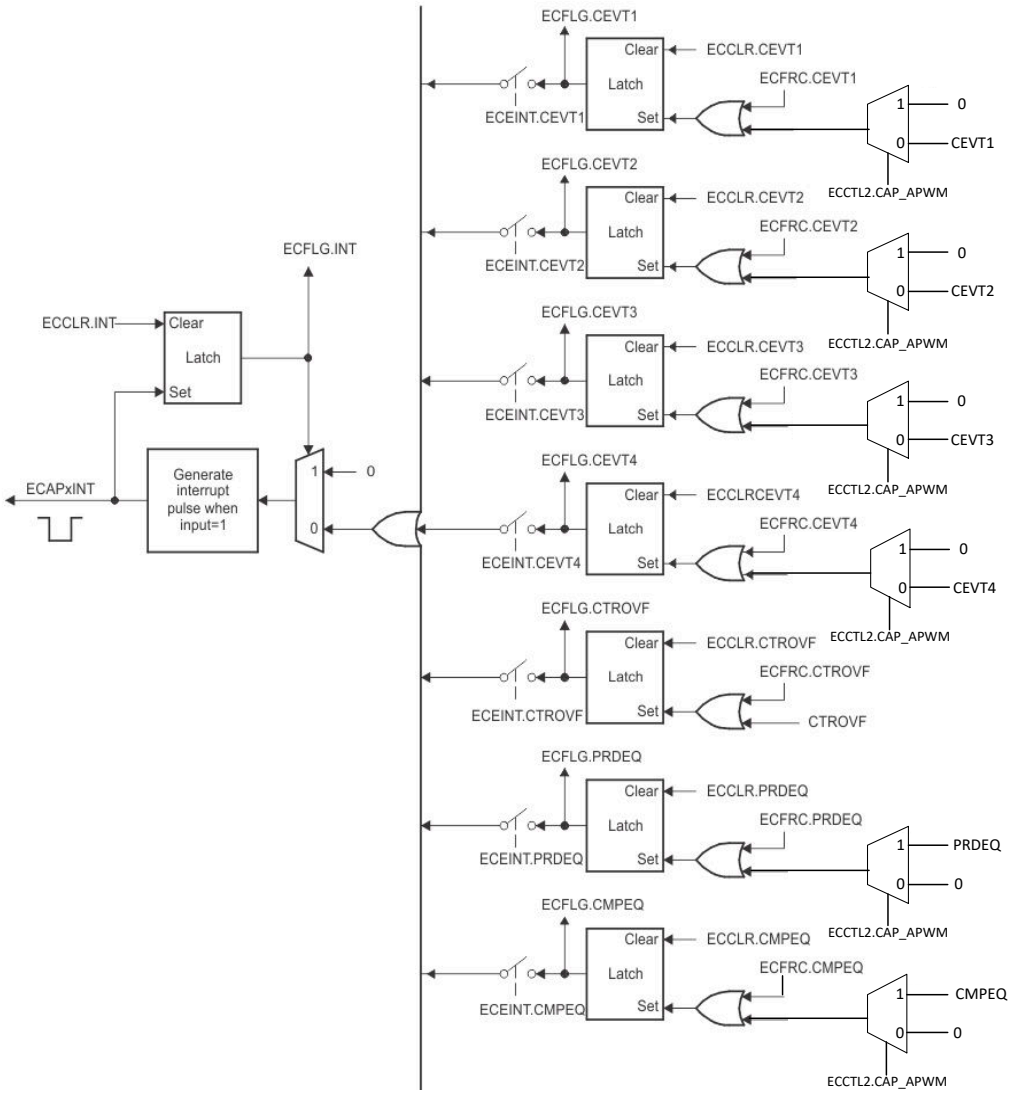


Figure 24-9. Interrupts in eCAP Module

24.5.8 DMA Interrupt

On Type 0 eCAP modules, the CPU was required to begin data transfers using DMA. New to the Type 1 eCAP, a separate DMA Trigger (ECAP_DMA_INT) enables continuous transfer of capture data from eCAP registers to on-chip memory using DMA. Any one of the four available interrupt events (CEVT1, CEVT2, CEVT3, and CEVT4) can be selected as the trigger source for ECAP_DMA_INT using ECCTL2 [DMAEVTSEL].

24.5.9 Shadow Load and Lockout Control

In capture mode, this logic inhibits (locks out) any shadow loading of CAP1 or CAP2 from APRD and ACMP registers, respectively.

In APWM mode, shadow loading is active and two choices are permitted:

- Immediate - APRD or ACMP are transferred to CAP1 or CAP2 immediately upon writing a new value.
- On period equal, CTR[31:0] = PRD[31:0].

24.5.10 APWM Mode Operation

Main operating highlights of the APWM section:

- The time-stamp counter bus is made available for comparison by way of 2 digital (32-bit) comparators.
- When CAP1/2 registers are not used in capture mode, their contents can be used as Period and Compare values in APWM mode.
- Double buffering is achieved using shadow registers APRD and ACMP (CAP3/4). The shadow register contents are transferred over to CAP1/2 registers, either immediately upon a write, or on a CTR = PRD trigger.
- In APWM mode, writing to CAP1/CAP2 active registers also writes the same value to the corresponding shadow registers CAP3/CAP4. This emulates immediate mode. Writing to the shadow registers CAP3/CAP4 invokes the shadow mode.
- During initialization, write to the active registers for both period and compare. This automatically copies the initial values into the shadow values. For subsequent compare updates during run-time, use the shadow registers.

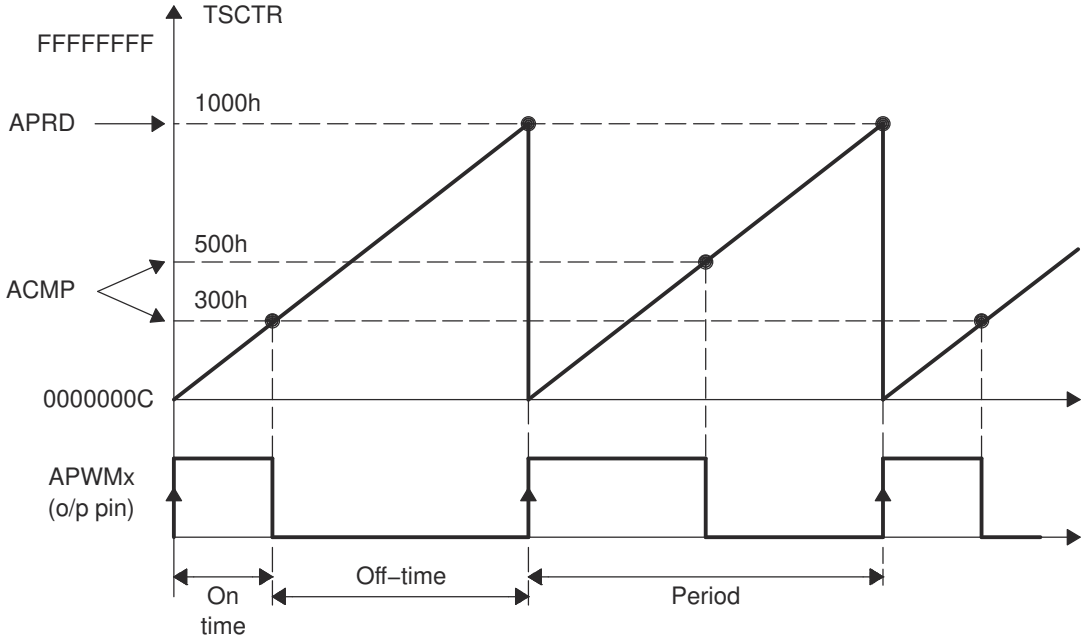


Figure 24-10. PWM Waveform Details Of APWM Mode Operation

The behavior of APWM active high mode (APWMPOL == 0) is as follows:

CMP = 0x00000000, output low for duration of period (0% duty)

CMP = 0x00000001, output high 1 cycle

CMP = 0x00000002, output high 2 cycles

CMP = PERIOD, output high except for 1 cycle (<100% duty)

CMP = PERIOD+1, output high for complete period (100% duty)

CMP > PERIOD+1, output high for complete period

The behavior of APWM active low mode (APWMPOL == 1) is as follows:

CMP = 0x00000000, output high for duration of period (0% duty)

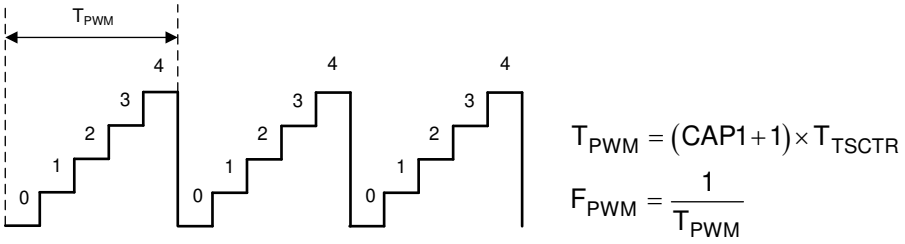
CMP = 0x00000001, output low 1 cycle

CMP = 0x00000002, output low 2 cycles

CMP = PERIOD, output low except for 1 cycle (<100% duty)

CMP = PERIOD+1, output low for complete period (100% duty)

CMP > PERIOD+1, output low for complete period



$$T_{PWM} = (CAP1 + 1) \times T_{TSCTR}$$

$$F_{PWM} = \frac{1}{T_{PWM}}$$

Figure 24-11. Time-Base Frequency and Period Calculation

24.6 Application of the eCAP Module

The following sections provide applications examples to show how to operate the eCAP module.

24.6.1 Example 1 - Absolute Time-Stamp Operation Rising-Edge Trigger

Figure 24-12 shows an example of continuous capture operation (Mod4 counter wraps around). In this figure, TSCTR counts-up without resetting and capture events are qualified on the rising edge only, this gives period (and frequency) information.

On an event, the TSCTR contents (time-stamp) is first captured, then Mod4 counter is incremented to the next state. When the TSCTR reaches FFFFFFFF (maximum value), the Mod4 counter wraps around to 00000000 (not shown in Figure 24-12), if this occurs, the CTROVF (counter overflow) flag is set, and an interrupt (if enabled) occurs, CTROVF (counter overflow) flag is set, and an interrupt (if enabled) occurs. Captured Time-stamps are valid at the point indicated by the diagram (after the fourth event); hence, event CEVT4 can conveniently be used to trigger an interrupt and the CPU can read data from the CAPx registers.

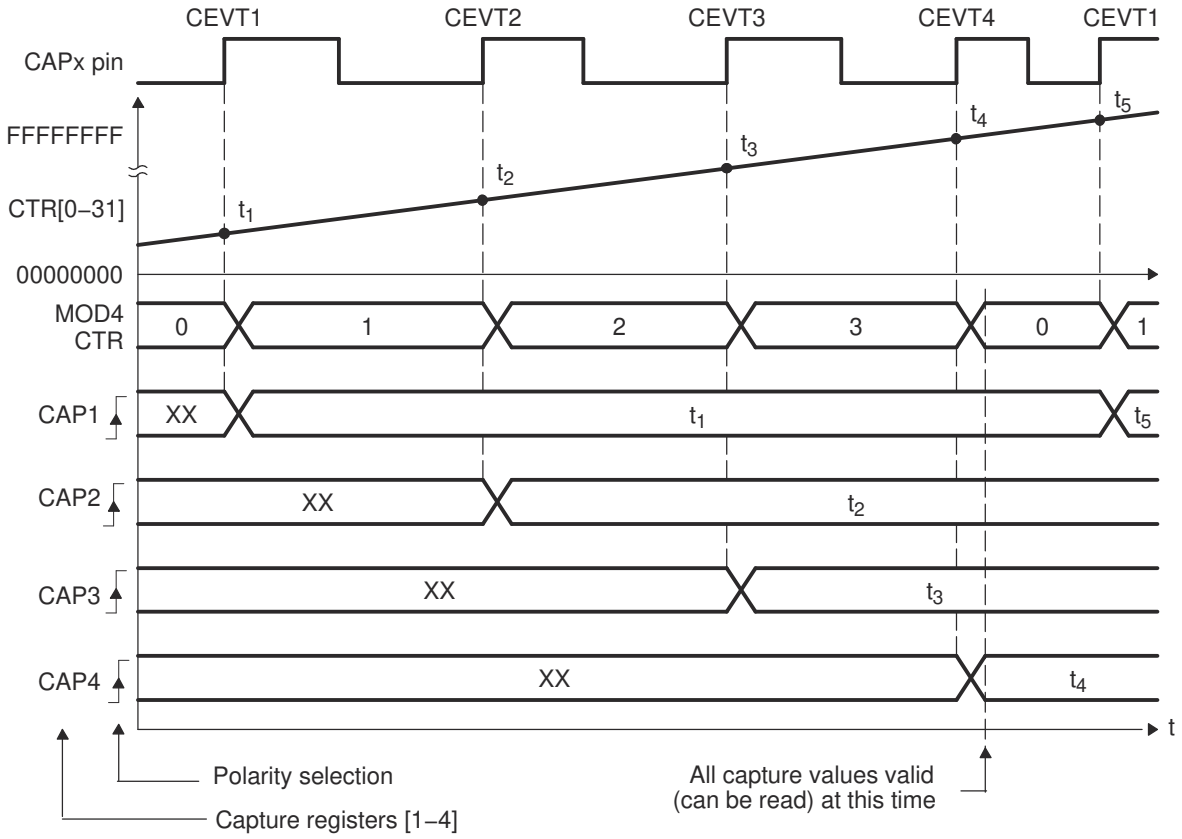


Figure 24-12. Capture Sequence for Absolute Time-stamp and Rising-Edge Detect

24.6.2 Example 2 - Absolute Time-Stamp Operation Rising- and Falling-Edge Trigger

In Figure 24-13, the eCAP operating mode is almost the same as in the previous section except capture events are qualified as either rising or falling edge, this now gives both period and duty cycle information, that is: Period1 = $t_3 - t_1$, Period2 = $t_5 - t_3$, …and so on. Duty Cycle1 (on-time %) = $(t_2 - t_1)$ / Period1 x 100%, and so on. Duty Cycle1 (off-time %) = $(t_3 - t_2)$ / Period1 x 100%, and so on.

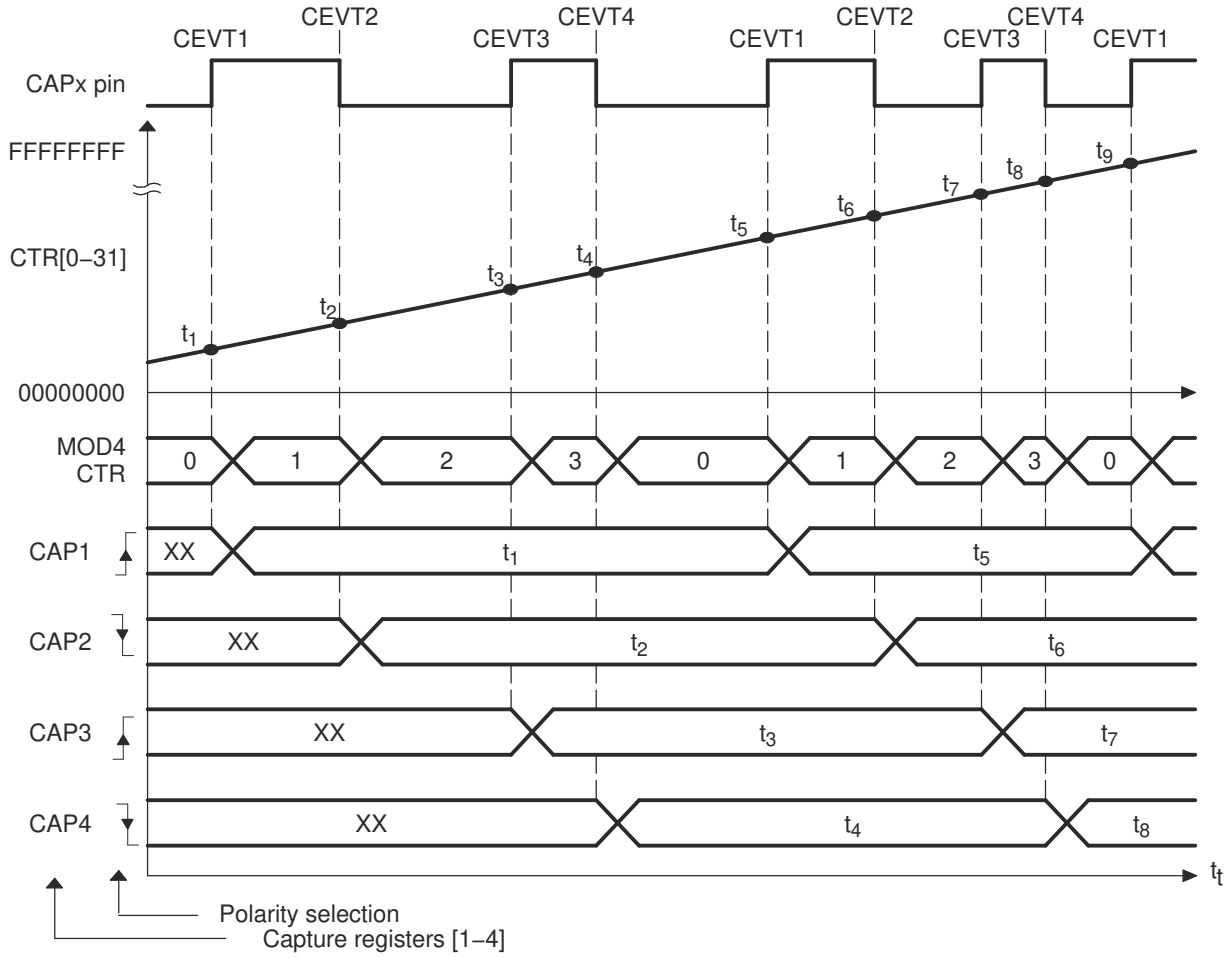


Figure 24-13. Capture Sequence for Absolute Time-stamp with Rising- and Falling-Edge Detect

24.6.3 Example 3 - Time Difference (Delta) Operation Rising-Edge Trigger

Figure 24-14 shows how the eCAP module can be used to collect delta timing data from pulse train waveforms. Here Continuous Capture mode (TSCTR counts-up without resetting, and Mod4 counter wraps around) is used. In Delta-time mode, TSCTR is reset back to zero on every valid event. Here capture events are qualified as rising edge only. On an event, TSCTR contents (Time-Stamp) is captured first, and then TSCTR is reset to zero. The Mod4 counter then increments to the next state. If TSCTR reaches FFFFFFFF (maximum value), before the next event, the Mod4 counter wraps around to 00000000 and continues, a CNTOVF (counter overflow) flag is set, and an interrupt (if enabled) occurs. The advantage of Delta-time mode is that the CAPx contents directly give timing data without the need for CPU calculations, that is, Period1 = T_1, Period2 = T_2, and so on. As shown in Figure 24-14, the CEVT1 event is a good trigger point to read the timing data, T_1, T_2, T_3, T_4 are all valid here.

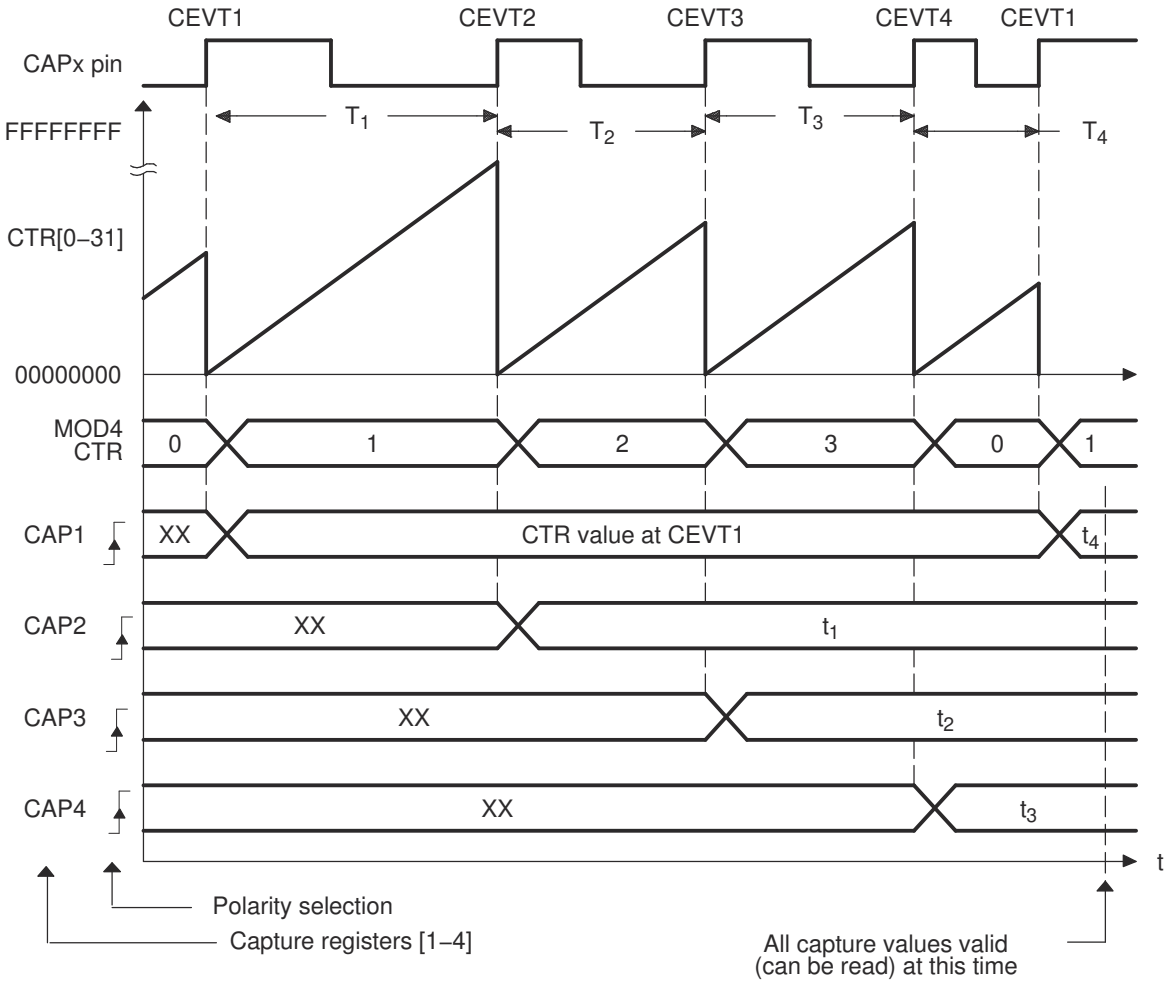


Figure 24-14. Capture Sequence for Delta Mode Time-stamp and Rising Edge Detect

24.6.4 Example 4 - Time Difference (Delta) Operation Rising- and Falling-Edge Trigger

In Figure 24-15, the eCAP operating mode is almost the same as in previous section except capture events are qualified as either rising or falling edge, this now gives both period and duty cycle information, that is: Period1 = T_1+T_2, Period2 = T_3+T_4, and so on. Duty Cycle1 (on-time %) = T_1 / Period1 x 100%, Duty Cycle1 (off-time %) = T_2 / Period1 x 100%, and so on.

During initialization, write to the active registers for both period and compare. This action automatically copies the init values into the shadow values. For subsequent compare updates during run-time, the shadow registers must be used.

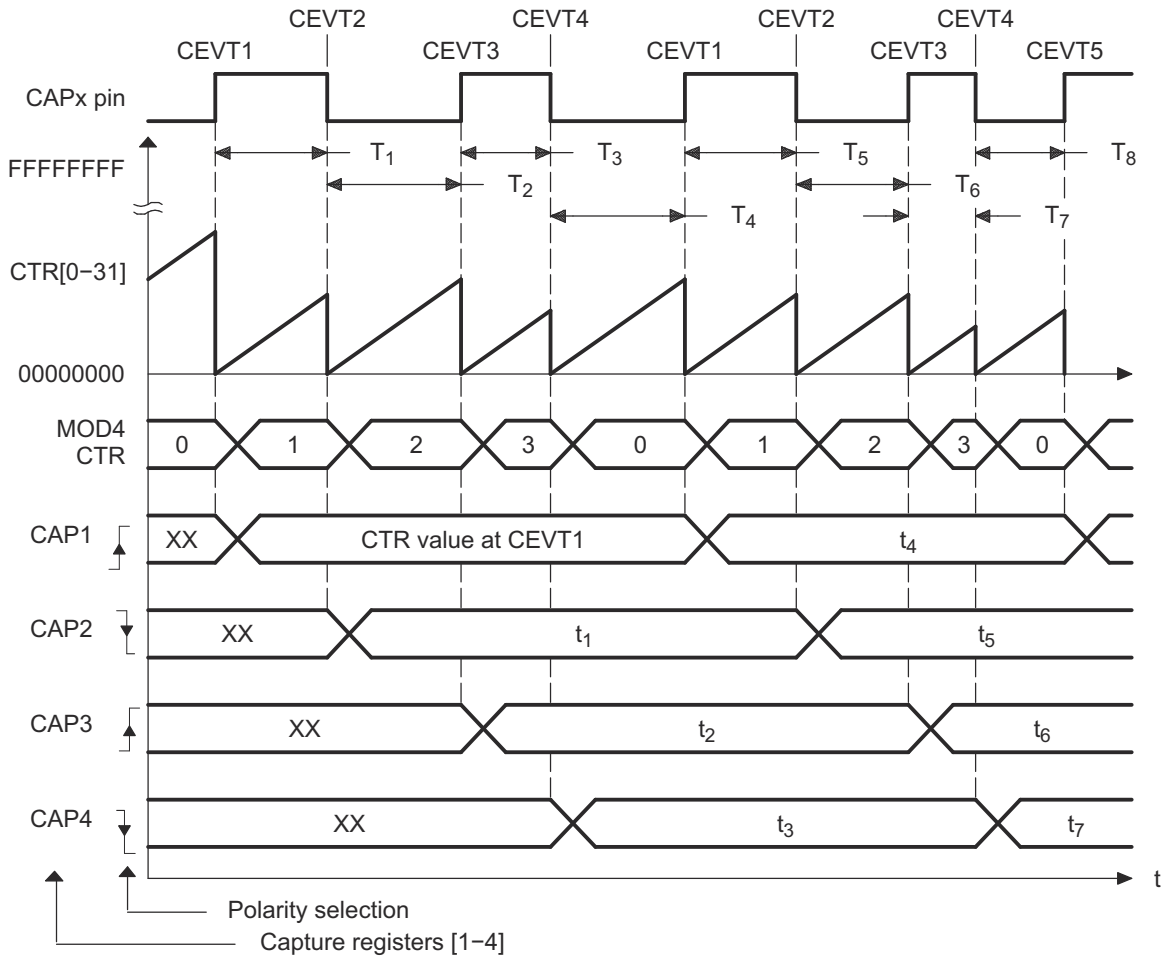


Figure 24-15. Capture Sequence for Delta Mode Time-stamp with Rising- and Falling-Edge Detect

24.7 Application of the APWM Mode

In this example, the eCAP module is configured to operate as a PWM generator. Here, a very simple single-channel PWM waveform is generated from the APWMx output pin. The PWM polarity is active high, which means that the compare value (CAP2 reg is now a compare register) represents the on-time (high level) of the period. Alternatively, if the APWMPOL bit is configured for active low, then the compare value represents the off-time.

24.7.1 Example 1 - Simple PWM Generation (Independent Channels)

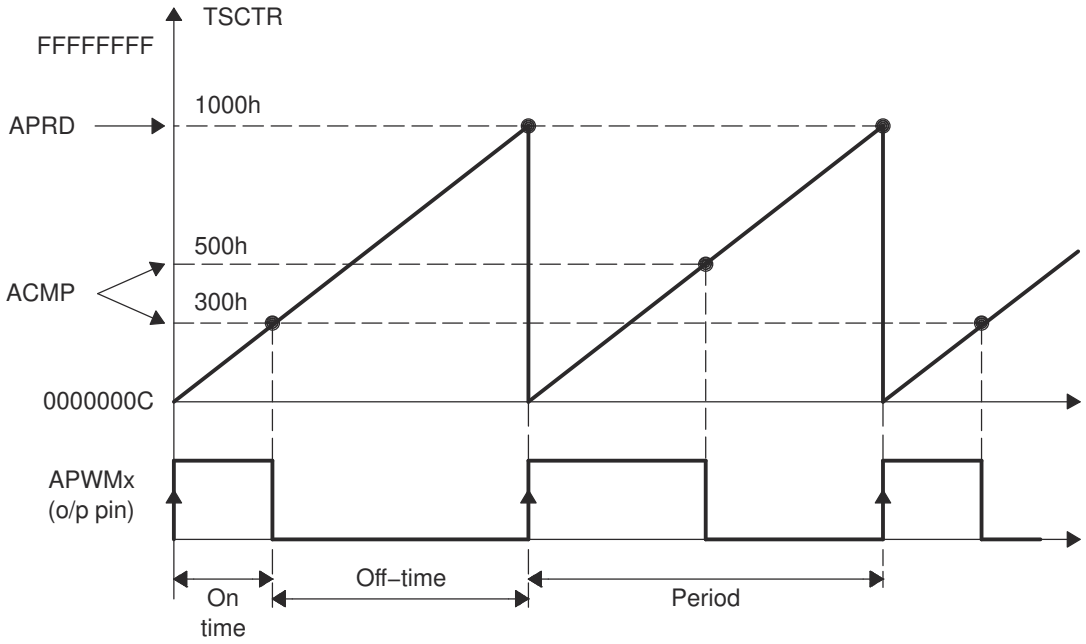


Figure 24-16. PWM Waveform Details of APWM Mode Operation

24.8 Software

24.8.1 ECAP Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/ecap

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

24.8.1.1 eCAP APWM Example

FILE: ecap_ex1_apwm.c

This program sets up the eCAP module in APWM mode. The PWM waveform will come out on GPIO5. The frequency of PWM is configured to vary between 10Hz and 20Hz using the shadow registers to load the next period/compare values.

24.8.1.2 eCAP Capture PWM Example

FILE: ecap_ex2_capture_pwm.c

This example configures ePWM3A for:
- Up count mode
- Period starts at 500 and goes up to 8000
- Toggle output on PRD

eCAP1 is configured to capture the time between rising and falling edge of the ePWM3A output.

External Connections
- eCAP1 is on GPIO16
- ePWM3A is on GPIO4
- Connect GPIO4 to GPIO16.

Watch Variables
- *ecap1PassCount* - Successful captures.
- *ecap1IntCount* - Interrupt counts.

24.8.1.3 eCAP APWM Phase-shift Example

FILE: ecap_ex3_apwm_phase_shift.c

This program sets up the eCAP1 and eCAP2 modules in APWM mode to generate the two phase-shifted PWM outputs of same duty and frequency value The frequency, duty and phase values can be programmed of choice by updating the defined macros. By default 10 Khz frequency, 50% duty and 30% phase shift values are used. eCAP2 output leads the eCAP1 output by 30% GPIO5 and GPIO6 are used as eCAP1/2 outputs and can be probed using analyzer/CRO to observe the waveforms.

24.8.1.4 eCAP Software Sync Example

FILE: ecap_ex4_sw_sync.c

This example configures ePWM3A for:
- Up count mode
- Period starts at 500 and goes up to 8000
- Toggle output on PRD

eCAP1, eCAP2 and eCAP3 are configured to capture the time between rising and falling edge of the ePWM3A output.

External Connections
- eCAP1, eCAP2, eCAP3 are on GPIO16
- ePWM3A is on GPIO4
- Connect GPIO4 to GPIO16.

Watch Variables

- *ecapPassCount* - Successful captures.
- *ecap3IntCount* - Interrupt counts.

24.9 eCAP Registers

This section describes the Enhanced Capture Registers.

24.9.1 ECAP Base Address Table (C28)

Table 24-2. ECAP Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
ECap1Regs	ECAP_REGS	ECAP1_BASE	0x0000_5200	YES	YES	YES	YES	YES
ECap2Regs	ECAP_REGS	ECAP2_BASE	0x0000_5240	YES	YES	YES	YES	YES
ECap3Regs	ECAP_REGS	ECAP3_BASE	0x0000_5280	YES	YES	YES	YES	YES
ECap4Regs	ECAP_REGS	ECAP4_BASE	0x0000_52C0	YES	YES	YES	YES	YES
ECap5Regs	ECAP_REGS	ECAP5_BASE	0x0000_5300	YES	YES	YES	YES	YES
ECap6Regs	ECAP_REGS	ECAP6_BASE	0x0000_5340	YES	YES	YES	YES	YES
ECap7Regs	ECAP_REGS	ECAP7_BASE	0x0000_5380	YES	YES	YES	YES	YES

24.9.2 ECAP_REGS Registers

Table 24-3 lists the memory-mapped registers for the ECAP_REGS registers. All register offset addresses not listed in Table 24-3 should be considered as reserved locations and the register contents should not be modified.

Table 24-3. ECAP_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	TSCTR	Time-Stamp Counter		Go
2h	CTRPHS	Counter Phase Offset Value Register		Go
4h	CAP1	Capture 1 Register		Go
6h	CAP2	Capture 2 Register		Go
8h	CAP3	Capture 3 Register		Go
Ah	CAP4	Capture 4 Register		Go
12h	ECCTL0	Capture Control Register 0	EALLOW	Go
14h	ECCTL1	Capture Control Register 1	EALLOW	Go
15h	ECCTL2	Capture Control Register 2	EALLOW	Go
16h	ECEINT	Capture Interrupt Enable Register	EALLOW	Go
17h	ECFLG	Capture Interrupt Flag Register		Go
18h	ECCLR	Capture Interrupt Clear Register		Go
19h	ECFRC	Capture Interrupt Force Register	EALLOW	Go
1Eh	ECAPSYNCINSEL	SYNC source select register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 24-4 shows the codes that are used for access types in this section.

Table 24-4. ECAP_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 24-4. ECAP_REGS Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

24.9.2.1 TSCTR Register (Offset = 0h) [Reset = 0h]

TSCTR is shown in Figure 24-17 and described in Table 24-5.

Return to the Summary Table.

Time-Stamp Counter

Figure 24-17. TSCTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSCTR																															
R/W-0h																															

Table 24-5. TSCTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TSCTR	R/W	0h	Active 32-bit counter register that is used as the capture time-base HR mode : 1) This register reads HRCOUNTER value and is not writable 2) can be reset using CTRFILTRESET 3) Its not synchronized to SYSCLK domain so reads may not be accurate Reset type: SYSRSn

24.9.2.2 CTRPHS Register (Offset = 2h) [Reset = 0h]

CTRPHS is shown in Figure 24-18 and described in Table 24-6.

Return to the Summary Table.

Counter Phase Offset Value Register

Figure 24-18. CTRPHS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTRPHS																															
R/W-0h																															

Table 24-6. CTRPHS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CTRPHS	R/W	0h	Counter phase value register that can be programmed for phase lag/lead. This register CTRPHS is loaded into TSCTR upon either a SYNCI event or S/W force via a control bit. Used to achieve phase control synchronization with respect to other eCAP and EPWM time-bases. This register is not applicable in HR mode. Reset type: SYSRSn

24.9.2.3 CAP1 Register (Offset = 4h) [Reset = 0h]

CAP1 is shown in Figure 24-19 and described in Table 24-7.

Return to the Summary Table.

Capture 1 Register

Figure 24-19. CAP1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP1																															
R/W-0h																															

Table 24-7. CAP1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP1	R/W	0h	This register can be loaded (written) by: - Time-Stamp counter value (TSCTR) during a capture event - Software - may be useful for test purposes or initialization - ARPD shadow register (CAP3) when used in APWM mode Reset type: SYSRSn

24.9.2.4 CAP2 Register (Offset = 6h) [Reset = 0h]

CAP2 is shown in Figure 24-20 and described in Table 24-8.

Return to the Summary Table.

Capture 2 Register

Figure 24-20. CAP2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP2																															
R/W-0h																															

Table 24-8. CAP2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP2	R/W	0h	This register can be loaded (written) by: - Time-Stamp (counter value) during a capture event - Software - may be useful for test purposes - ACMP shadow register (CAP4) when used in APWM mode Reset type: SYSRSn

24.9.2.5 CAP3 Register (Offset = 8h) [Reset = 0h]

CAP3 is shown in Figure 24-21 and described in Table 24-9.

Return to the Summary Table.

Capture 3 Register

Figure 24-21. CAP3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP3																															
R/W-0h																															

Table 24-9. CAP3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP3	R/W	0h	In CMP mode, this is a time-stamp capture register. In APWM mode, this is the period shadow (APRD) register. You can update the PWM period value through this register. CAP3 (APRD) shadows CAP1 in this mode. Reset type: SYSRSn

24.9.2.6 CAP4 Register (Offset = Ah) [Reset = 0h]

CAP4 is shown in Figure 24-22 and described in Table 24-10.

Return to the Summary Table.

Capture 4 Register

Figure 24-22. CAP4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP4																															
R/W-0h																															

Table 24-10. CAP4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP4	R/W	0h	In CMP mode, this is a time-stamp capture register. In APWM mode, this is the compare shadow (ACMP) register. You can update the PWM compare value via this register. CAP4 (ACMP) shadows CAP2 in this mode. Reset type: SYSRSn

24.9.2.7 ECCTL0 Register (Offset = 12h) [Reset = 7Fh]

ECCTL0 is shown in Figure 24-23 and described in Table 24-11.

Return to the Summary Table.

Capture Control Register 0

Figure 24-23. ECCTL0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									INPUTSEL						
R-0-0h									R/W-7Fh						

Table 24-11. ECCTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R-0	0h	Reserved
6-0	INPUTSEL	R/W	7Fh	Capture input source select bits 0000000 capture input is ECAPxINPUT[0] 0000001 capture input is ECAPxINPUT[1] 0000010 capture input is ECAPxINPUT[2] ... 1111111 capture input is ECAPxINPUT[127] Reset type: CPU1.SYSRSn

24.9.2.8 ECCTL1 Register (Offset = 14h) [Reset = 0h]

ECCTL1 is shown in Figure 24-24 and described in Table 24-12.

Return to the Summary Table.

Capture Control Register 1

Figure 24-24. ECCTL1 Register

15	14	13	12	11	10	9	8
FREE_SOFT		PRESCALE					CAPLDEN
R/W-0h		R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
CTRRST4	CAP4POL	CTRRST3	CAP3POL	CTRRST2	CAP2POL	CTRRST1	CAP1POL
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 24-12. ECCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	FREE_SOFT	R/W	0h	Emulation Control Reset type: SYSRSn 0h (R/W) = TSCTR counter stops immediately on emulation suspend 1h (R/W) = TSCTR counter runs until = 0 2h (R/W) = TSCTR counter is unaffected by emulation suspend (Run Free) 3h (R/W) = TSCTR counter is unaffected by emulation suspend (Run Free)
13-9	PRESCALE	R/W	0h	Event Filter prescale select Reset type: SYSRSn 0h (R/W) = Divide by 1 (i.e,. no prescale, by-pass the prescaler) 1h (R/W) = Divide by 2 2h (R/W) = Divide by 4 3h (R/W) = Divide by 6 4h (R/W) = Divide by 8 5h (R/W) = Divide by 10 1Eh (R/W) = Divide by 60 1Fh (R/W) = Divide by 62
8	CAPLDEN	R/W	0h	Enable Loading of CAP1-4 registers on a capture event. Note that this bit does not disable CEVTn events from being generated. Reset type: SYSRSn 0h (R/W) = Disable CAP1-4 register loads at capture event time. 1h (R/W) = Enable CAP1-4 register loads at capture event time.
7	CTRRST4	R/W	0h	Counter Reset on Capture Event 4 Reset type: SYSRSn 0h (R/W) = Do not reset counter on Capture Event 4 (absolute time stamp operation) 1h (R/W) = Reset counter after Capture Event 4 time-stamp has been captured (used in difference mode operation)
6	CAP4POL	R/W	0h	Capture Event 4 Polarity select Reset type: SYSRSn 0h (R/W) = Capture Event 4 triggered on a rising edge (RE) 1h (R/W) = Capture Event 4 triggered on a falling edge (FE)
5	CTRRST3	R/W	0h	Counter Reset on Capture Event 3 Reset type: SYSRSn 0h (R/W) = Do not reset counter on Capture Event 3 (absolute time stamp) 1h (R/W) = Reset counter after Event 3 time-stamp has been captured (used in difference mode operation)
4	CAP3POL	R/W	0h	Capture Event 3 Polarity select Reset type: SYSRSn 0h (R/W) = Capture Event 3 triggered on a rising edge (RE) 1h (R/W) = Capture Event 3 triggered on a falling edge (FE)

Table 24-12. ECCTL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	CTRRST2	R/W	0h	Counter Reset on Capture Event 2 Reset type: SYSRSn 0h (R/W) = Do not reset counter on Capture Event 2 (absolute time stamp) 1h (R/W) = Reset counter after Event 2 time-stamp has been captured (used in difference mode operation)
2	CAP2POL	R/W	0h	Capture Event 2 Polarity select Reset type: SYSRSn 0h (R/W) = Capture Event 2 triggered on a rising edge (RE) 1h (R/W) = Capture Event 2 triggered on a falling edge (FE)
1	CTRRST1	R/W	0h	Counter Reset on Capture Event 1 Reset type: SYSRSn 0h (R/W) = Do not reset counter on Capture Event 1 (absolute time stamp) 1h (R/W) = Reset counter after Event 1 time-stamp has been captured (used in difference mode operation)
0	CAP1POL	R/W	0h	Capture Event 1 Polarity select Reset type: SYSRSn 0h (R/W) = Capture Event 1 triggered on a rising edge (RE) 1h (R/W) = Capture Event 1 triggered on a falling edge (FE)

24.9.2.9 ECCTL2 Register (Offset = 15h) [Reset = 6h]

ECCTL2 is shown in Figure 24-25 and described in Table 24-13.

Return to the Summary Table.

Capture Control Register 2

Figure 24-25. ECCTL2 Register

15	14	13	12	11	10	9	8
MODCNTRSTS		DMAEVTSEL		CTRFILTRESET	APWMPOL	CAP_APWM	SWSYNC
R-0h		R/W-0h		R-0/W1C-0h	R/W-0h	R/W-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
SYNCO_SEL		SYNCI_EN	TSCTRSTOP	REARM	STOP_WRAP		CONT_ONESHT
R/W-0h		R/W-0h	R/W-0h	R-0/W1S-0h	R/W-3h		R/W-0h

Table 24-13. ECCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	MODCNTRSTS	R	0h	This bit field reads current status on modulo counter 00b (R) = CAP1 register gets loaded on next capture event. 01b (R) = CAP2 register gets loaded on next capture event. 10b (R) = CAP3 register gets loaded on next capture event. 11b (R) = CAP4 register gets loaded on next capture event. Reset type: CPU1.SYSRSn
13-12	DMAEVTSEL	R/W	0h	DMA event select 00b (R/W) = DMA interrupt source is CEVT1 01b (R/W) = DMA interrupt source is CEVT2 10b (R/W) = DMA interrupt source is CEVT3 11b (R/W) = DMA interrupt source is CEVT4 Note: ECCTL1.CAPLDEN also needs to be set to "1" for ECAPxDMA_INT to be generated Reset type: CPU1.SYSRSn
11	CTRFILTRESET	R-0/W1C	0h	Reset Bit 0h (R) = No effect 1h (W) = Resets event filter, counter, modulo counter and CEVT[1,2,3,4] and CNTOVF , HRERROR flags Note: This provides an ability start capture module from known state in case spurious inputs are captured while ECAP is configured. Reset type: CPU1.SYSRSn
10	APWMPOL	R/W	0h	APWM output polarity select. This is applicable only in APWM operating mode. Reset type: SYSRSn 0h (R/W) = Output is active high (Compare value defines high time) 1h (R/W) = Output is active low (Compare value defines low time)
9	CAP_APWM	R/W	0h	CAP/APWM operating mode select Reset type: SYSRSn 0h (R/W) = ECAP module operates in capture mode. This mode forces the following configuration: - Inhibits TSCTR resets via CTR = PRD event - Inhibits shadow loads on CAP1 and 2 registers - Permits user to enable CAP1-4 register load - CAPx/APWMx pin operates as a capture input 1h (R/W) = ECAP module operates in APWM mode. This mode forces the following configuration: - Resets TSCTR on CTR = PRD event (period boundary - Permits shadow loading on CAP1 and 2 registers - Disables loading of time-stamps into CAP1-4 registers - CAPx/APWMx pin operates as a APWM output

Table 24-13. ECCTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	SWSYNC	R-0/W1S	0h	Software-forced Counter (TSCTR) Synchronizer. This provides the user a method to generate a synchronization pulse through software. In APWM mode, the synchronization pulse can also be sourced from the CTR = PRD event. Reset type: SYSRSn 0h (R/W) = Writing a zero has no effect. Reading always returns a zero 1h (R/W) = Writing a one forces a TSCTR shadow load of current ECAP module and any ECAP modules down-stream providing the SYNCO_SEL bits are 0,0. After writing a 1, this bit returns to a zero.
7-6	SYNCO_SEL	R/W	0h	Sync-Out Select Reset type: SYSRSn 0h (R/W) = sync out signal is SWSYNC 1h (R/W) = Select CTR = PRD event to be the sync-out signal. Note: Selection CTR = PRD is meaningful only in APWM mode 2h (R/W) = Disable sync out signal 3h (R/W) = Disable sync out signal
5	SYNCI_EN	R/W	0h	Counter (TSCTR) Sync-In select mode Reset type: SYSRSn 0h (R/W) = Disable sync-in option 1h (R/W) = Enable counter (TSCTR) to be loaded from CTRPHS register upon either a SYNCI signal or a S/W force event.
4	TSCTRSTOP	R/W	0h	Time Stamp (TSCTR) Counter Stop (freeze) Control Reset type: SYSRSn 0h (R/W) = TSCTR stopped 1h (R/W) = TSCTR free-running
3	REARM	R-0/W1S	0h	Re-Arming Control. Note: The re-arm function is valid in one shot or continuous mode Reset type: SYSRSn 0h (R/W) = Has no effect (reading always returns a 0) 1h (R/W) = Arms the one-shot sequence as follows: 1) Resets the Mod4 counter to zero 2) Unfreezes the Mod4 counter 3) Enables capture register loads
2-1	STOP_WRAP	R/W	3h	Stop value for one-shot mode. This is the number (between 1-4) of captures allowed to occur before the CAP(1-4) registers are frozen, that is, capture sequence is stopped. Wrap value for continuous mode. This is the number (between 1-4) of the capture register in which the circular buffer wraps around and starts again. Notes: STOP_WRAP is compared to Mod4 counter and, when equal, 2 actions occur: - Mod4 counter is stopped (frozen) - Capture register loads are inhibited In one-shot mode, further interrupt events are blocked until re-armed. Reset type: SYSRSn 0h (R/W) = Stop after Capture Event 1 in one-shot mode Wrap after Capture Event 1 in continuous mode. 1h (R/W) = Stop after Capture Event 2 in one-shot mode Wrap after Capture Event 2 in continuous mode. 2h (R/W) = Stop after Capture Event 3 in one-shot mode Wrap after Capture Event 3 in continuous mode. 3h (R/W) = Stop after Capture Event 4 in one-shot mode Wrap after Capture Event 4 in continuous mode.
0	CONT_ONESHT	R/W	0h	Continuous or one-shot mode control (applicable only in capture mode) Reset type: SYSRSn 0h (R/W) = Operate in continuous mode 1h (R/W) = Operate in one-Shot mode

24.9.2.10 ECEINT Register (Offset = 16h) [Reset = 0h]

ECEINT is shown in Figure 24-26 and described in Table 24-14.

Return to the Summary Table.

The interrupt enable bits (CEVT1, ...) block any of the selected events from generating an interrupt. Events will still be latched into the flag bit (ECFLG register) and can be forced/cleared via the ECFRC/ECCLR registers.
The proper procedure for configuring peripheral modes and interrupts is as follows:
- Disable global interrupts
- Stop eCAP counter
- Disable eCAP interrupts
- Configure peripheral registers
- Clear spurious eCAP interrupt flags
- Enable eCAP interrupts
- Start eCAP counter
- Enable global interrupts

Figure 24-26. ECEINT Register

15	14	13	12	11	10	9	8
RESERVED							RESERVED
R-0h							R/W-0h
7	6	5	4	3	2	1	0
CTR_EQ_CMP	CTR_EQ_PRD	CTROVF	CEVT4	CEVT3	CEVT2	CEVT1	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

Table 24-14. ECEINT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	RESERVED	R	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7	CTR_EQ_CMP	R/W	0h	Counter Equal Compare Interrupt Enable Reset type: SYSRSn 0h (R/W) = Disable Compare Equal as an Interrupt source 1h (R/W) = Enable Compare Equal as an Interrupt source
6	CTR_EQ_PRD	R/W	0h	Counter Equal Period Interrupt Enable Reset type: SYSRSn 0h (R/W) = Disable Period Equal as an Interrupt source 1h (R/W) = Enable Period Equal as an Interrupt source
5	CTROVF	R/W	0h	Counter Overflow Interrupt Enable Reset type: SYSRSn 0h (R/W) = Disabled counter Overflow as an Interrupt source 1h (R/W) = Enable counter Overflow as an Interrupt source
4	CEVT4	R/W	0h	Capture Event 4 Interrupt Enable Reset type: SYSRSn 0h (R/W) = Disable Capture Event 4 as an Interrupt source 1h (R/W) = Capture Event 4 Interrupt Enable
3	CEVT3	R/W	0h	Capture Event 3 Interrupt Enable Reset type: SYSRSn 0h (R/W) = Disable Capture Event 3 as an Interrupt source 1h (R/W) = Enable Capture Event 3 as an Interrupt source
2	CEVT2	R/W	0h	Capture Event 2 Interrupt Enable Reset type: SYSRSn 0h (R/W) = Disable Capture Event 2 as an Interrupt source 1h (R/W) = Enable Capture Event 2 as an Interrupt source
1	CEVT1	R/W	0h	Capture Event 1 Interrupt Enable Reset type: SYSRSn 0h (R/W) = Disable Capture Event 1 as an Interrupt source 1h (R/W) = Enable Capture Event 1 as an Interrupt source

Table 24-14. ECEINT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	RESERVED	R	0h	Reserved

24.9.2.11 ECFLG Register (Offset = 17h) [Reset = 0h]

ECFLG is shown in Figure 24-27 and described in Table 24-15.

Return to the Summary Table.

Capture Interrupt Flag Register

Figure 24-27. ECFLG Register

15	14	13	12	11	10	9	8
RESERVED							RESERVED
R-0h							R-0h
7	6	5	4	3	2	1	0
CTR_CMP	CTR_PRD	CTROVF	CEVT4	CEVT3	CEVT2	CEVT1	INT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 24-15. ECFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	CTR_CMP	R	0h	Compare Equal Compare Status Flag. This flag is active only in APWM mode. Reset type: SYSRSn 0h (R/W) = Indicates no event occurred 1h (R/W) = Indicates the counter (TSCTR) reached the compare register value (ACMP)
6	CTR_PRD	R	0h	Counter Equal Period Status Flag. This flag is only active in APWM mode. Reset type: SYSRSn 0h (R/W) = Indicates no event occurred 1h (R/W) = Indicates the counter (TSCTR) reached the period register value (APRD) and was reset.
5	CTROVF	R	0h	Counter Overflow Status Flag. This flag is active in CAP and APWM mode. Reset type: SYSRSn 0h (R/W) = Indicates no event occurred 1h (R/W) = Indicates the counter (TSCTR) has made the transition from FFFFFFFF to 00000000
4	CEVT4	R	0h	Capture Event 4 Status Flag This flag is only active in CAP mode. Reset type: SYSRSn 0h (R/W) = Indicates no event occurred 1h (R/W) = Indicates the fourth event occurred at ECAPx pin
3	CEVT3	R	0h	Capture Event 3 Status Flag. This flag is active only in CAP mode. Reset type: SYSRSn 0h (R/W) = Indicates no event occurred 1h (R/W) = Indicates the third event occurred at ECAPx pin.
2	CEVT2	R	0h	Capture Event 2 Status Flag. This flag is only active in CAP mode. Reset type: SYSRSn 0h (R/W) = Indicates no event occurred 1h (R/W) = Indicates the second event occurred at ECAPx pin.
1	CEVT1	R	0h	Capture Event 1 Status Flag. This flag is only active in CAP mode. Reset type: SYSRSn 0h (R/W) = Indicates no event occurred 1h (R/W) = Indicates the first event occurred at ECAPx pin.
0	INT	R	0h	Global Interrupt Status Flag Reset type: SYSRSn 0h (R/W) = Indicates no event occurred 1h (R/W) = Indicates that an interrupt was generated.

24.9.2.12 ECCLR Register (Offset = 18h) [Reset = 0h]

ECCLR is shown in Figure 24-28 and described in Table 24-16.

Return to the Summary Table.

Capture Interrupt Clear Register

Figure 24-28. ECCLR Register

15	14	13	12	11	10	9	8
RESERVED							RESERVED
R-0h							R-0/W1C-0h
7	6	5	4	3	2	1	0
CTR_CMP	CTR_PRD	CTROVF	CEVT4	CEVT3	CEVT2	CEVT1	INT
R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h

Table 24-16. ECCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	RESERVED	R	0h	Reserved
8	RESERVED	R-0/W1C	0h	Reserved
7	CTR_CMP	R-0/W1C	0h	Counter Equal Compare Status Clear Reset type: SYSRSn 0h (R/W) = Writing a 0 has no effect. Always reads back a 0 1h (R/W) = Writing a 1 clears the CTR=CMP flag.
6	CTR_PRD	R-0/W1C	0h	Counter Equal Period Status Clear Reset type: SYSRSn 0h (R/W) = Writing a 0 has no effect. Always reads back a 0 1h (R/W) = Writing a 1 clears the CTR=PRD flag.
5	CTROVF	R-0/W1C	0h	Counter Overflow Status Clear Reset type: SYSRSn 0h (R/W) = Writing a 0 has no effect. Always reads back a 0 1h (R/W) = Writing a 1 clears the CTROVF flag.
4	CEVT4	R-0/W1C	0h	Capture Event 4 Status Clear Reset type: SYSRSn 0h (R/W) = Writing a 0 has no effect. Always reads back a 0 1h (R/W) = Writing a 1 clears the CEVT4 flag.
3	CEVT3	R-0/W1C	0h	Capture Event 3 Status Clear Reset type: SYSRSn 0h (R/W) = Writing a 0 has no effect. Always reads back a 0 1h (R/W) = Writing a 1 clears the CEVT3 flag.
2	CEVT2	R-0/W1C	0h	Capture Event 2 Status Clear Reset type: SYSRSn 0h (R/W) = Writing a 0 has no effect. Always reads back a 0 1h (R/W) = Writing a 1 clears the CEVT2 flag.
1	CEVT1	R-0/W1C	0h	Capture Event 1 Status Clear Reset type: SYSRSn 0h (R/W) = Writing a 0 has no effect. Always reads back a 0 1h (R/W) = Writing a 1 clears the CEVT1 flag.
0	INT	R-0/W1C	0h	ECAP Global Interrupt Status Clear Reset type: SYSRSn 0h (R/W) = Writing a 0 has no effect. Always reads back a 0 1h (R/W) = Writing a 1 clears the INT flag and enable further interrupts to be generated if any of the event flags are set to 1

24.9.2.13 ECFRC Register (Offset = 19h) [Reset = 0h]

ECFRC is shown in Figure 24-29 and described in Table 24-17.

Return to the Summary Table.

Capture Interrupt Force Register

Figure 24-29. ECFRC Register

15	14	13	12	11	10	9	8
RESERVED							RESERVED
R-0h							R-0/W1S-0h
7	6	5	4	3	2	1	0
CTR_CMP	CTR_PRD	CTROVF	CEVT4	CEVT3	CEVT2	CEVT1	RESERVED
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0h

Table 24-17. ECFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	RESERVED	R	0h	Reserved
8	RESERVED	R-0/W1S	0h	Reserved
7	CTR_CMP	R-0/W1S	0h	Force Counter Equal Compare Interrupt. This event is only active in APWM mode. Reset type: SYSRSn 0h (R/W) = No effect. Always reads back a 0. 1h (R/W) = Writing a 1 sets the CTR_CMP flag.
6	CTR_PRD	R-0/W1S	0h	Force Counter Equal Period Interrupt. This event is only active in APWM mode. Reset type: SYSRSn 0h (R/W) = No effect. Always reads back a 0. 1h (R/W) = Writing a 1 sets the CTR_PRD flag.
5	CTROVF	R-0/W1S	0h	Force Counter Overflow Reset type: SYSRSn 0h (R/W) = No effect. Always reads back a 0. 1h (R/W) = Writing a 1 to this bit sets the CTROVF flag.
4	CEVT4	R-0/W1S	0h	Force Capture Event 4. This event is only active in CAP mode. Reset type: SYSRSn 0h (R/W) = No effect. Always reads back a 0. 1h (R/W) = Writing a 1 sets the CEVT4 flag.
3	CEVT3	R-0/W1S	0h	Force Capture Event 3. This event is only active in CAP mode. Reset type: SYSRSn 0h (R/W) = No effect. Always reads back a 0. 1h (R/W) = Writing a 1 sets the CEVT3 flag.
2	CEVT2	R-0/W1S	0h	Force Capture Event 2. This event is only active in CAP mode. Reset type: SYSRSn 0h (R/W) = No effect. Always reads back a 0. 1h (R/W) = Writing a 1 sets the CEVT2 flag.
1	CEVT1	R-0/W1S	0h	Force Capture Event 1. This event is only active in CAP mode. Reset type: SYSRSn 0h (R/W) = No effect. Always reads back a 0. 1h (R/W) = Sets the CEVT1 flag.
0	RESERVED	R	0h	Reserved

24.9.2.14 ECAPSYNCINSEL Register (Offset = 1Eh) [Reset = 1h]

ECAPSYNCINSEL is shown in Figure 24-30 and described in Table 24-18.

Return to the Summary Table.

SYNC source select register

Figure 24-30. ECAPSYNCINSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											SEL				
R-0h																											R/W-1h				

Table 24-18. ECAPSYNCINSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4-0	SEL	R/W	1h	These bits determines the source of SYNCIN signal. Reset type: SYSRSn 0h (R/W) = Disable Syncin to eCAP 1h (R/W) = EPWM1SYNCOUT 2h (R/W) = EPWM2SYNCOUT 3h (R/W) = EPWM3SYNCOUT 4h (R/W) = EPWM4SYNCOUT 5h (R/W) = EPWM5SYNCOUT 6h (R/W) = EPWM6SYNCOUT 7h (R/W) = EPWM7SYNCOUT 8h (R/W) = EPWM8SYNCOUT 9h (R/W) = EPWM9SYNCOUT Ah (R/W) = EPWM10SYNCOUT Bh (R/W) = EPWM11SYNCOUT Ch (R/W) = EPWM12SYNCOUT Dh (R/W) = EPWM13SYNCOUT Eh (R/W) = EPWM14SYNCOUT Fh (R/W) = EPWM15SYNCOUT 10h (R/W) = EPWM16SYNCOUT 11h (R/W) = ECAP1SYNCOUT 12h (R/W) = ECAP2SYNCOUT 13h (R/W) = ECAP3SYNCOUT 14h (R/W) = ECAP4SYNCOUT 15h (R/W) = ECAP5SYNCOUT 16h (R/W) = ECAP6SYNCOUT 17h (R/W) = ECAP7SYNCOUT 18h (R/W) = INPUTXBAROUT5 19h (R/W) = INPUTXBAROUT6 1Ah (R/W) = EtherCATSYNC0 1Bh (R/W) = EtherCATSYNC1 1Ch (R/W) = RSVD 1Dh (R/W) = RSVD 1Eh (R/W) = RSVD 1Fh (R/W) = FSI_RXA_RX_TRIG1

24.9.3 ECAP Registers to Driverlib Functions

Table 24-19. ECAP Registers to Driverlib Functions

File	Driverlib Function
TSCTR	
ecap.h	ECAP_getTimeBaseCounter
CTRPHS	
ecap.h	ECAP_setPhaseShiftCount
CAP1	
ecap.h	ECAP_setAPWMPeriod

Table 24-19. ECAP Registers to Driverlib Functions (continued)

File	Driverlib Function
ecap.h	ECAP_getEventTimeStamp
CAP2	
ecap.h	ECAP_setAPWMCompare
ecap.h	ECAP_getEventTimeStamp
CAP3	
ecap.h	ECAP_setAPWMShadowPeriod
ecap.h	ECAP_getEventTimeStamp
CAP4	
ecap.h	ECAP_setAPWMShadowCompare
ecap.h	ECAP_getEventTimeStamp
ECCTL0	
ecap.h	ECAP_selectECAPInput
ECCTL1	
ecap.c	ECAP_setEmulationMode
ecap.h	ECAP_setEventPrescaler
ecap.h	ECAP_setEventPolarity
ecap.h	ECAP_enableCounterResetOnEvent
ecap.h	ECAP_disableCounterResetOnEvent
ecap.h	ECAP_enableTimeStampCapture
ecap.h	ECAP_disableTimeStampCapture
ECCTL2	
ecap.h	ECAP_setCaptureMode
ecap.h	ECAP_reArm
ecap.h	ECAP_enableCaptureMode
ecap.h	ECAP_enableAPWMMode
ecap.h	ECAP_enableLoadCounter
ecap.h	ECAP_disableLoadCounter
ecap.h	ECAP_loadCounter
ecap.h	ECAP_setSyncOutMode
ecap.h	ECAP_stopCounter
ecap.h	ECAP_startCounter
ecap.h	ECAP_setAPWMPolarity
ecap.h	ECAP_resetCounters
ecap.h	ECAP_setDMASource
ecap.h	ECAP_getModuloCounterStatus
ECEINT	
ecap.h	ECAP_enableInterrupt
ecap.h	ECAP_disableInterrupt
ECFLG	
ecap.h	ECAP_getInterruptSource
ecap.h	ECAP_getGlobalInterruptStatus
ECCLR	
ecap.h	ECAP_clearInterrupt
ecap.h	ECAP_clearGlobalInterrupt
ECFRC	

Table 24-19. ECAP Registers to Driverlib Functions (continued)

File	Driverlib Function
ecap.h	ECAP_forceInterrupt
SYNCINSEL	
ecap.h	ECAP_setSyncInPulseSource

This page intentionally left blank.

Chapter 25

High Resolution Capture (HRCAP)

This chapter describes the operation of the high resolution capture (HRCAP) module. The HRCAP submodule described here is part of the Type1 eCAP. HRCAP measures the width of external pulses to a higher degree of accuracy than the eCAP module. See the *C2000 Real-Time Control Peripheral Reference Guide* for a list of all devices with an HRCAP module of the same type, to determine the differences between types, and for a list of device-specific differences within a type. A detailed description of all referenced functions can be found in the C2000Ware documentation.

25.1 Introduction

Uses for the HRCAP module include:

- Capacitive touch applications
- High-resolution period and duty cycle measurements of pulse train cycles
- Instantaneous speed measurements
- Instantaneous frequency measurements
- Voltage measurements across an isolation boundary
- Distance/sonar measurement and scanning
- Measuring flow

25.1.1 HRCAP Related Collateral

Foundational Materials

- C2000 Academy - Control Peripherals

Getting Started Materials

- *Leveraging High Resolution Capture (HRCAP) for Single Wire Data Transfer Application Report*

25.1.2 Features

The HRCAP module includes the following features:

- Pulse-width capture in either non-high-resolution or high-resolution modes
- Absolute mode pulse-width capture
- Continuous or one-shot capture
- Interrupt on either falling or rising edge
- Continuous mode capture of pulse widths in 4-deep buffer
- Hardware calibration logic for precision high-resolution capture

All of the previous resources are available on any pin using the Input X-BAR.

25.1.3 Description

Improvements from the Type 0 HRCAP are:

- Simplified calibration scheme:
  - HRCAP is always functional; never offline to perform calibration
  - Calibration is always running in the background; drastically reduced software overhead to calibrate
- Reduced software overhead to compute fractional bits
- Fractional and integer portions are packed into 32 bits
- All eCAP hardware is accessible when using the HRCAP enhancements. See Section 25.3 for practical considerations.
- Usage of the HRCAP is now unified with the eCAP

The HRCAP enhancement has been added to eCAP 6 and eCAP 7 to allow signals to be captured asynchronously to SYSCLK. Each HRCAP submodule includes one capture channel in addition to a hardware calibration block. All eCAP hardware is accessible when using the HRCAP enhancements; however, using the Event Filter or the Input Qualifier is not valid, as these are synchronous to SYSCLK.

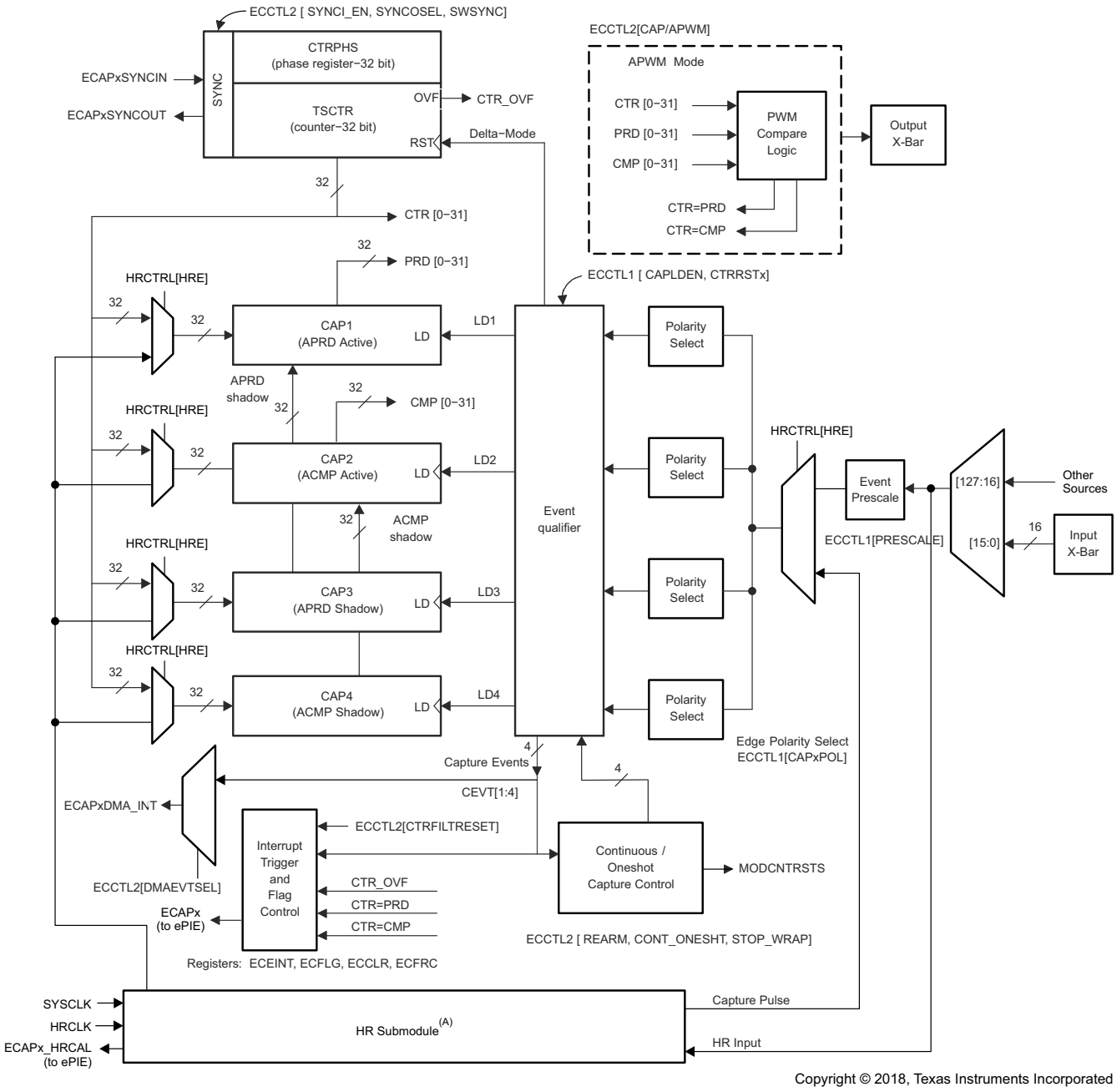
Each HRCAP-capable channel has the following independent key resources:

- All hardware of the respective eCAP
- High-resolution calibration logic
- Dedicated calibration interrupt

25.2 Operational Details

Figure 25-1 shows the various components that implement the high-resolution capture functionality of the eCAP module. Note that existing eCAP resources are reused, which requires that the eCAP module is set up before using the HRCAP enhancements. For simplicity, absolute timestamp measurements are recommended. See Section 25.3 for more details.

All HRCAP measurements are relative-time measures, in terms of minimum step size. Calibration hardware as well as software functions, have been provided to convert relative-time measurements to time-converted measurements in terms of seconds. The calibration hardware and software is only required if time-converted measurements are required.



A. The HRCAP submodule is not available on all eCAP modules; in this case, the high-resolution muxes and hardware are not implemented.

Figure 25-1. HRCAP Operations Block Diagram

25.2.1 HRCAP Clocking

Unlike previous Type-0 HRCAP modules, the Type-1 eCAP, with HRCAP functionality, does not require a second PLL. However, the module still requires both SYSCLK and a second asynchronous clock source called HRCLK. The HRCLK is sensitive to changes in both temperature and voltage. For this reason, when using time-converted measurements, it is required to make periodic continuous calibrations.

25.2.2 HRCAP Initialization Sequence

Following are the HRCAP initialization sequence steps. When using the HRCAP to take relative-time measurements, only steps 1-5 are required. When using the HRCAP to take time-converted measurements, all steps are required.

1. Enable HRCLK using HRCAP_enableHighResolutionClock()
2. Delay 1 µS
3. Enable HR mode using HRCAP_enableHighResolution()
4. Delay 1 µS
5. Configure the eCAP module as desired, including interrupts
6. Set calibration period using HRCAP_setCalibrationPeriod()
7. Enable continuous calibration using HRCAP_setCalibrationMode()
8. Enable interrupts using HRCAP_enableCalibrationInterrupt()
9. Start calibration using HRCAP_startCalibration()

25.2.3 HRCAP Interrupts

The HRCAP enhancements leverage the existing eCAP interrupts (see *Interrupt Control* in the ECAP chapter) in addition to HRCALINT, which is used exclusively by the hardware calibration block. HRCALINT can be triggered by the following conditions:

1. SYSCLKCTR = HRCALIBPERIOD
2. SYSCLKCTR or HRCLKCTR experience an overflow condition

Figure 25-2 shows this logic.

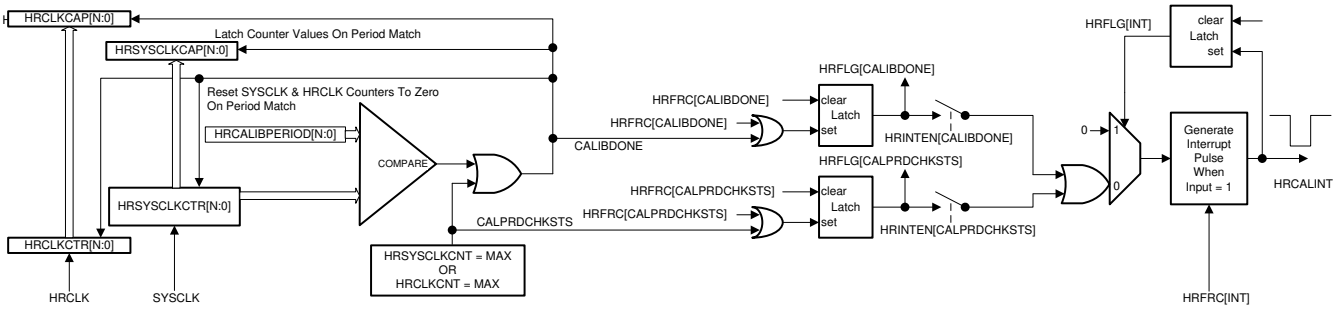


Figure 25-2. HRCAP Calibration

25.2.4 HRCAP Calibration

Note

For HRCAP calibration to work, the system needs to operate at SYSCLK frequency > 100 MHz.

The following section applies only to time-converted measurements; calibration for relative-time measurements is not required. All values captured by the HRCAP submodule are in number of HRCLK cycles. The HRCLK speed varies widely with temperature and voltage, thus a scale factor is required to convert the capture value to the SYSCLK domain. For the same reason, it is required to periodically recalculate the scale factor. The HRCAP submodule has a calibration block to reduce software overhead when calculating a scale factor between HRCLK and SYSCLK.

The calibration block contains the following key resources:

- **HRSYSCLKCNT:** A 32-bit counter connected to SYSCLK. The counter starts counting when CALIBSTART is set.
- **HRCLKCNT:** A 32-bit counter connected to HRCLK. The counter starts counting when CALIBSTART is set.
- **HRCALIBPERIOD:** Calibration period, calibration is stopped when HRSYSCLKCNT is equal to the value in this register.
- **HRSYSCLKCAP:** On a calibration period match, the value of HRSYSCLKCNT is captured into HRSYSCLKCAP.
- **HRCLKCAP:** On a calibration period match, the value of HRCLKCNT is captured into HRCLKCAP.
- **HRCALINT:** An interrupt that occurs on a calibration period match, or when one of the counter registers experiences an overflow condition.

The calibration logic consists of two free-running counters; one clocked by HRCLK(HRCLKCTR) and one clocked by SYSCLK(HRSYSCLKCTR). When HRSYSCLKCTR is equal to HRCALIBPERIOD, the calibration block captures and resets both counter values, then triggers an interrupt, indicating a new scale factor is ready to be calculated. The scale factor can be found by dividing HRSYSCLKCAP by HRCLKCAP (see Equation 9). A DriverLib function, HRCAP_getScaleFactor, has been provided to determine the scale factor. This function can be called inside of the calibration interrupt service routine. If one of the counters experiences overflow, the CALPRDCHKSTS flag is set. The full details of the calibration block are described in Figure 25-2.

$$ScaleFactor = \frac{HRSYSCLKCAP}{HRCLKCAP} \tag{9}$$

Note

- Even with calibration, noise on the 1.2-V VDD supply negatively affects the standard deviation of the HRCAP submodule. Care can be taken to make sure that the 1.2-V supply is clean, and that noisy internal events such as enabling and disabling clock trees have been minimized while using the HRCAP submodule.
- When HRCLK > SYSCLK, calibration stops immaturely to mitigate improper calibration, SYSCLK must be set to at least 100 MHz.

25.2.4.1 Applying the Scale Factor

A DriverLib function has been provided to apply the scale factor to a capture value, HRCAP_getEventTimeStampNanoseconds(). Equation 10 shows how to convert a raw count to seconds without using the DriverLib function.

$$Measurement(ns) = \frac{RawCount \times scaleFactor}{128} * SysClkPrd(ns) \quad (10)$$

Table 25-1. Scale Factor

Parameter	Explanation
RawCount	Capture value as read from ECAP_REGS_CAP1-4
ScaleFactor(1)	The Scale factor as calculated from Equation 10
128	Constant determined by the hardware of the HRCAP submodule
SysClkPrd(nS)	Period of the system clock
Measurement(nS)	Signal converted to nS

(1) The scale factor is not automatically applied to captured values. The user is required to apply the scale factor to all captured values as shown in Equation 10.

25.3 Known Exceptions

In HRCAP mode:

- Enabling and disabling core clocks negatively affects the standard deviation of the HRCAP submodule. Do not enable or disable core clocks while taking measurements.
- TSCTR is not writable; however, TSCTR can be reset using ECCTL2[CTRFILTRESET]
- Input synchronization is not applicable when using the HRCAP enhancements, because the HRCAP submodule is asynchronous to SYSCLK.
- The Event Filter functionality is not applicable for HRCAP, which defeats the purpose of HRCAP as the Event Filter's output is synchronous to SYSCLK.
- The best practice is to use absolute time mode for high-resolution mode. If time difference mode is used, it can lead to inaccurate results if the fractional value is not taken into consideration for capture events that have reset the time base counter.
  - Actual Capture Value = (Capture Value) – (fractional value of reference event that reset the counter)
- For high-frequency input signals, the CPU can not be able to cope with the speed of the captures. In such a case, one-shot mode is recommended. This mode allows the device to capture up to four edges before waiting to be serviced when the CPU is ready. This is applicable for the eCAP as well; however, in that case the event filter can be used to reduce the rate of captures.

25.4 Software

25.4.1 HRCAP Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/hrcap

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

25.4.1.1 HRCAP Capture and Calibration Example

FILE: hrcap_ex1_capture.c

This example configures an ECAP to use HRCAP functionality to capture time between edges on input GPIO2.

External Connections
The user must provide a signal to GPIO2. XCLKOUT has been configured to an output GPIO and can be externally jumped to serve this purpose. See Sysconfig file for XCLKOUT GPIO selected.

Watch Variables
- onTime1, onTime2
- offTime1, offTime2
- period1, period2

25.5 HRCAP Registers

This section describes the High-Resolution Capture Registers.

25.5.1 HRCAP Base Address Table (C28)

Table 25-2. HRCAP Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
HRCap6Regs	HRCAP_REGS	HRCAP6_BASE	0x0000_5360	YES	YES	YES	YES	YES
HRCap7Regs	HRCAP_REGS	HRCAP7_BASE	0x0000_53A0	YES	YES	YES	YES	YES

25.5.2 HRCAP_REGS Registers

Table 25-3 lists the memory-mapped registers for the HRCAP_REGS registers. All register offset addresses not listed in Table 25-3 should be considered as reserved locations and the register contents should not be modified.

Table 25-3. HRCAP_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	HRCTL	High-Res Control Register	EALLOW	Go
4h	HRINTEN	High-Res Calibration Interrupt Enable Register	EALLOW	Go
6h	HRFLG	High-Res Calibration Interrupt Flag Register		Go
8h	HRCLR	High-Res Calibration Interrupt Clear Register		Go
Ah	HRFRC	High-Res Calibration Interrupt Force Register	EALLOW	Go
Ch	HRCALPRD	High-Res Calibration Period Register	EALLOW	Go
Eh	HRSYSCLKCTR	High-Res Calibration SYSCLK Counter Register		Go
10h	HRSYSCLKCAP	High-Res Calibration SYSCLK Capture Register		Go
12h	HRCLKCTR	High-Res Calibration HRCLK Counter Register		Go
14h	HRCLKCAP	High-Res Calibration HRCLK Capture Register		Go

Complex bit access types are encoded to fit into small table cells. Table 25-4 shows the codes that are used for access types in this section.

Table 25-4. HRCAP_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

25.5.2.1 HRCTL Register (Offset = 0h) [Reset = 0h]

HRCTL is shown in Figure 25-3 and described in Table 25-5.

Return to the Summary Table.

High-Res Control Register

Figure 25-3. HRCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED		CALIBCONT	CALIBSTS	CALIBSTART	PRDSEL	HRCLKE	HRE
R/W-0h		R/W-0h	R-0h	R-0/W1S-0h	R/W-0h	R/W-0h	R/W-0h

Table 25-5. HRCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R/W	0h	Reserved
5	CALIBCONT	R/W	0h	Continuous mode Calibration Select Bit: 0 Continuous mode disabled. 1 Continuous mode enabled. Calibration automatically restarts at end of current calibration cycle. Reset type: CPU1.SYSRSn
4	CALIBSTS	R	0h	Calibration status Bit: 0 No active calibration cycle 1 Calibration cycle in progress Reset type: CPU1.SYSRSn
3	CALIBSTART	R-0/W1S	0h	Calibration start Bit: 0 No effect 1 Starts the calibration cycle Reset type: CPU1.SYSRSn
2	PRDSEL	R/W	0h	Calibration Period Match Select Bit: 0 Use SYSCLK Counter For Period Match (default at reset) 1 Reserved Reset type: CPU1.SYSRSn
1	HRCLKE	R/W	0h	High Resolution Clock Enable Bit: 0 High resolution clock disabled (default at reset) 1 High resolution clock enabled. The clock should be enabled before enabling the high res function via the HRE bit. Reset type: CPU1.SYSRSn
0	HRE	R/W	0h	High Resolution Enable Bit: 0 High resolution mode disabled (default at reset) 1 High resolution mode enabled. Enabling this mode will connect the capture registers and edge event modes of the ECAP to be accessed by the High Res function. Note: The High Res clock needs to be enabled (using the HRCLKE bit) first before enabling the module. Allow a certain start up stabilization period before enabling the module. Reset type: CPU1.SYSRSn

25.5.2.2 HRINTEN Register (Offset = 4h) [Reset = 0h]

HRINTEN is shown in Figure 25-4 and described in Table 25-6.

Return to the Summary Table.

High-Res Calibration Interrupt Enable Register

Figure 25-4. HRINTEN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED					CALPRDCHKSTS	CALIBDONE	RESERVED
R-0-0h					R/W-0h	R/W-0h	R-0-0h

Table 25-6. HRINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R-0	0h	Reserved
2	CALPRDCHKSTS	R/W	0h	Calibration Period Check status Interrupt Enable: 0 Disable Calibration Period Check interrupt status 1 Enable Calibration Period Check interrupt status Reset type: CPU1.SYSRSn
1	CALIBDONE	R/W	0h	Calibration done Interrupt Enable: 0 Disable Calibration done Interrupt 1 Enable Calibration done Interrupt Reset type: CPU1.SYSRSn
0	RESERVED	R-0	0h	Reserved

25.5.2.3 HRFLG Register (Offset = 6h) [Reset = 0h]

HRFLG is shown in Figure 25-5 and described in Table 25-7.

Return to the Summary Table.

High-Res Calibration Interrupt Flag Register

Figure 25-5. HRFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED					CALPRDCHKSTS	CALIBDONE	CALIBINT
R-0-0h					R-0h	R-0h	R-0h

Table 25-7. HRFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R-0	0h	Reserved
2	CALPRDCHKSTS	R	0h	Calibration period check status Flag Bit: 1 Indicates that calibration ended before PRDCHK due to overflow on one of the counters. 0 Indicates no event occurred. Note: This bit remains latched until cleared by the user using the HRCLR [CALPRDCHKSTS] bit. Reset type: CPU1.SYSRSn
1	CALIBDONE	R	0h	Calibration Done Interrupt Flag Bit: 1 Indicates calibration cycle is completed 0 Indicates calibration cycle has not completed. Note: This bit remains latched until cleared by the user using the HRCLR [CALIBDONE] bit. Reset type: CPU1.SYSRSn
0	CALIBINT	R	0h	Global calibration Interrupt Status Flag: 1 Indicates that an interrupt was generated from CALIBDONE or CALPRDCHKSTS. 0 Indicates no interrupt generated. Reset type: CPU1.SYSRSn

25.5.2.4 HRCLR Register (Offset = 8h) [Reset = 0h]

HRCLR is shown in Figure 25-6 and described in Table 25-8.

Return to the Summary Table.

High-Res Calibration Interrupt Clear Register

Figure 25-6. HRCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED					CALPRDCHKSTS	CALIBDONE	CALIBINT
R-0-0h					R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h

Table 25-8. HRCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R-0	0h	Reserved
2	CALPRDCHKSTS	R-0/W1C	0h	Clear Calibration period check status Flag Bit: 1 Clears the CALPRDCHKSTS flag register bit. 0 No effect. Note: H/W has priority over CPU writes if the user tries to clear a flag bit and an event occurs on the same cycle that tries to set the flag for the selected bit. Reset type: CPU1.SYSRSn
1	CALIBDONE	R-0/W1C	0h	Clear Calibration Done Interrupt Flag Bit: 1 Clears the CALIBDONE interrupt flag register bit. 0 No effect. Note: H/W has priority over CPU writes if the user tries to clear a flag bit and an event occurs on the same cycle that tries to set the flag for the selected bit. Reset type: CPU1.SYSRSn
0	CALIBINT	R-0/W1C	0h	Clear Global calibration Interrupt Flag 1 Clears the Global interrupt flag and enables further interrupts to be generated if any of the event flags are set. 0 No effect. Reset type: CPU1.SYSRSn

25.5.2.5 HRFRC Register (Offset = Ah) [Reset = 0h]

HRFRC is shown in Figure 25-7 and described in Table 25-9.

Return to the Summary Table.

High-Res Calibration Interrupt Force Register

Figure 25-7. HRFRC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED					CALPRDCHKSTS	CALIBDONE	RESERVED
R-0-0h					R-0/W1S-0h	R-0/W1S-0h	R-0-0h

Table 25-9. HRFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R-0	0h	Reserved
2	CALPRDCHKSTS	R-0/W1S	0h	Force CALPRDCHKSTS flag: 0 No effect 1 Sets the CALPRDCHKSTS flag. Reset type: CPU1.SYSRSn
1	CALIBDONE	R-0/W1S	0h	Force CALIBDONE flag: 0 No effect 1 Sets the CALIBDONE flag. Reset type: CPU1.SYSRSn
0	RESERVED	R-0	0h	Reserved

25.5.2.6 HRCALPRD Register (Offset = Ch) [Reset = 003FFFFFh]

HRCALPRD is shown in Figure 25-8 and described in Table 25-10.

Return to the Summary Table.

High-Res Calibration Period Register

Figure 25-8. HRCALPRD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRD																															
R/W-003FFFFFh																															

Table 25-10. HRCALPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PRD	R/W	003FFFFFh	Register to program calibration period. The period value is matched against HRSYSCLKCTR. On a match an interrupt is generated and the counter registers values are captured. Reset type: CPU1.SYSRSn

25.5.2.7 HRSYSCLKCTR Register (Offset = Eh) [Reset = 0h]

HRSYSCLKCTR is shown in Figure 25-9 and described in Table 25-11.

Return to the Summary Table.

High-Res Calibration SYSCLK Counter Register

Figure 25-9. HRSYSCLKCTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HRSYSCLKCTR																															
R-0h																															

Table 25-11. HRSYSCLKCTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HRSYSCLKCTR	R	0h	Current SYSCLK counter value Reset type: CPU1.SYSRSn

25.5.2.8 HRSYSCLKCAP Register (Offset = 10h) [Reset = 0h]

HRSYSCLKCAP is shown in Figure 25-10 and described in Table 25-12.

Return to the Summary Table.

High-Res Calibration SYSCLK Capture Register

Figure 25-10. HRSYSCLKCAP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HRSYSCLKCAP																															
R-0h																															

Table 25-12. HRSYSCLKCAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HRSYSCLKCAP	R	0h	HRSYSCLKCTR is captures into this register at end of calibration cycle. Reset type: CPU1.SYSRSn

25.5.2.9 HRCLKCTR Register (Offset = 12h) [Reset = 0h]

HRCLKCTR is shown in Figure 25-11 and described in Table 25-13.

Return to the Summary Table.

High-Res Calibration HRCLK Counter Register

Figure 25-11. HRCLKCTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HRCLKCTR																															
R-0h																															

Table 25-13. HRCLKCTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HRCLKCTR	R	0h	Current HRCLK counter value Note: HRCLK is not synchronized to SYSCLK domain so reads may not be accurate Reset type: CPU1.SYSRSn

25.5.2.10 HRCLKCAP Register (Offset = 14h) [Reset = 0h]

HRCLKCAP is shown in Figure 25-12 and described in Table 25-14.

Return to the Summary Table.

High-Res Calibration HRCLK Capture Register

Figure 25-12. HRCLKCAP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HRCLKCAP																															
R-0h																															

Table 25-14. HRCLKCAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HRCLKCAP	R	0h	HRCLKCTR is captures into this register at end of calibration cycle. Note: HRCLK is not synchronized to SYSCLK domain so reads may not be accurate Reset type: CPU1.SYSRSn

25.5.3 HRCAP Registers to Driverlib Functions

Table 25-15. HRCAP Registers to Driverlib Functions

File	Driverlib Function
TSCTR	
-	
CTRPHS	
-	
CAP1	
-	
CAP2	
-	
CAP3	
-	
CAP4	
-	
ECCTL0	
-	
ECCTL1	
-	
ECCTL2	
-	
ECEINT	
-	
ECFLG	
-	
ECCLR	
-	
ECFRC	
-	
ECAPSYNCINSEL	
-	

Table 25-15. HRCAP Registers to Driverlib Functions (continued)

File	Driverlib Function
HRCTL	
hrcap.h	HRCAP_enableHighResolution
hrcap.h	HRCAP_disableHighResolution
hrcap.h	HRCAP_enableHighResolutionClock
hrcap.h	HRCAP_disbleHighResolutionClock
hrcap.h	HRCAP_startCalibration
hrcap.h	HRCAP_setCalibrationMode
hrcap.h	HRCAP_isCalibrationBusy
HRINTEN	
hrcap.h	HRCAP_enableCalibrationInterrupt
hrcap.h	HRCAP_disableCalibrationInterrupt
HRFLG	
hrcap.h	HRCAP_getCalibrationFlags
HRCLR	
hrcap.h	HRCAP_clearCalibrationFlags
HRFRC	
hrcap.h	HRCAP_forceCalibrationFlags
HRCALPRD	
hrcap.h	HRCAP_setCalibrationPeriod
hrcap.h	HRCAP_configCalibrationPeriod
HRSYSCLKCTR	
-	
HRSYSCLKCAP	
hrcap.h	HRCAP_getCalibrationClockPeriod
HRCLKCTR	
-	
HRCLKCAP	
hrcap.h	HRCAP_getCalibrationClockPeriod

Chapter 26

Enhanced Pulse Width Modulator (ePWM)

The enhanced pulse width modulator (ePWM) peripheral is a key element in controlling many of the power electronic systems found in both commercial and industrial equipment. These systems include digital motor control, switch mode power supply control, uninterruptible power supplies (UPS), and other forms of power conversion. The ePWM peripheral can also perform a digital-to-analog (DAC) function, where the duty cycle is equivalent to a DAC analog value; it is sometimes referred to as a power DAC.

This chapter is applicable for ePWM type 4 with added register protection capability. See the *C2000 Real-Time Control Peripheral Reference Guide* for a list of all devices with an ePWM module of the same type, to determine the differences between the types, and for a list of device-specific differences within a type.

26.1 Introduction

This chapter includes an overview and information about each submodule:

- Time Base (TB) Submodule
- Counter Compare (CC) Submodule
- Action Qualifier (AQ) Submodule
- Dead-Band Generator (DB) Submodule
- PWM Chopper (PC) Submodule
- Trip Zone (TZ) Submodule
- Event Trigger (ET) Submodule
- Digital Compare (DC) Submodule

The ePWM Type 4 is functionally compatible to Type 2 (a Type 3 does not exist). Type 4 has the following enhancements in addition to the Type 2 features:

- **Register Address Map:** Additional registers are required for new features on ePWM Type 4. The ePWM register address space has been remapped for better alignment and easy usage.
- **Delayed Trip Functionality:** Changes have been added to achieve deadband insertion capabilities to support, for example, delayed trip functionality needed for peak current mode control type application scenarios. This has been accomplished by allowing comparator events to go into the Action Qualifier as a trigger event (Events T1 and T2). If comparator T1 / T2 events are used to edit the PWM, changes to the PWM waveform do not take place immediately. Instead, the waveform synchronizes to the next TBCLK.
- **Dead-Band Generator Submodule Enhancements:** Shadowing of the DBCTL register to allow dynamic configuration changes.
- **One Shot and Global Load of Registers:** The ePWM Type 4 allows one shot and global load capability from shadow to active registers to avoid partial loads in, for example, multiphase applications. ePWM Type 4 also allows a programmable prescale of shadow to active load events. ePWM Type 4 Global Load can simplify ePWM software by removing interrupts and ensuring that all registers are loaded at the same time.
- **Trip-Zone Submodule Enhancements:** Independent flags have been added to reflect the trip status for each of the TZ sources. Changes have been made to the trip-zone submodule to support certain power converter switching techniques like valley switching.
- **Digital Compare Submodule Enhancements:** Blanking window filter register width has been increased from 8 to 16 bits. DCCAP functionality has been enhanced to provide more programmability.
- **PWM SYNC Related Enhancements:** The ePWM Type 4 allows PWM SYNCOUT generation based on CMPC and CMPD events. These events can also be used for PWMSYNC pulse selection.

The ePWM Type 2 is fully compatible to Type 1. Type 2 has the following enhancements in addition to the Type 1 features:

- **High-Resolution Dead-Band Capability:** High-resolution capability is added to dead-band RED and FED in half-cycle clocking mode.
- **Dead-Band Generator Submodule Enhancements:** The ePWM Type 2 has features to enable both RED and FED on either PWM outputs. Provides increased dead band with 14-bit counters and dead-band / dead-band high-resolution registers are shadowed
- **High-Resolution Extension available on ePWMxB outputs:** Provides the ability to enable high-resolution period and duty cycle control on ePWMxB outputs. This is discussed in more detail in Section 26.15.
- **Counter Compare Submodule Enhancements:** The ePWM Type 2 allows interrupts and SOC events to be generated by additional counter compares CMPC and CMPD.
- **Event Trigger Submodule Enhancements:** Prescaling logic to issue interrupt requests and ADC start of conversion expanded up to every 15 events. It allows software initialization of event counters on SYNC event.
- **Digital Compare Submodule Enhancements:** Digital Compare Trip Select logic [DCTRIPSEL] has up to 12 external trip sources selected by the Input X-BAR logic in addition to an ability to OR all of them (up to 14 [external and internal sources]) to create the respective DCxEVTs.
- **Simultaneous Writes to TBPRD and CMPx Registers:** This feature allows writes to TBPRD, CMPA:CMPAHR, CMPB:CMPBHR, CMPC and CMPD of any ePWM module to be tied to any other ePWM module, and also allows all ePWM modules to be tied to a particular ePWM module if desired.
- **Shadow to Active Load on SYNC of TBPRD and CMP Registers:** This feature supports simultaneous writes of TBPRD and CMPA/B/C/D registers.

An effective PWM peripheral must be able to generate complex pulse width waveforms with minimal CPU overhead or intervention and must be highly programmable and very flexible while being easy to understand and use. The ePWM unit described here addresses these requirements by allocating all needed timing and control resources on a per PWM channel basis. Cross coupling or sharing of resources has been avoided; instead, the ePWM is built up from smaller single channel submodules with separate resources that can operate together as required to form a system. This modular approach results in an orthogonal architecture and provides a more transparent view of the peripheral structure, helping users to understand the operation quickly.

In this document, the letter x within a signal or submodule name is used to indicate a generic ePWM instance on a device. For example, output signals EPWMxA and EPWMxB refer to the output signals from the ePWMx instance. Thus, EPWM1A and EPWM1B belong to ePWM1 and likewise EPWM4A and EPWM4B belong to ePWM4.

Type0 to Type1 Enhancements

- **Increased Dead-Band Resolution:** Dead-band clocking has been enhanced to allow half-cycle clocking to double resolution.
- **Enhanced Interrupt and SOC Generation:** Interrupts and ADC start-of-conversion can now be generated on both the TBCTR == zero and TBCTR == period events. This feature enables dual edge PWM control. Additionally, the ADC start-of-conversion can be generated from an event defined in the digital compare submodule.
- **High-Resolution Period Capability:** Provides the ability to enable high-resolution period. This is discussed in more detail in Section 26.15.
- **Digital Compare Submodule:** The digital compare submodule enhances the event triggering and trip zone submodules by providing filtering, blanking and improved trip functionality to digital compare signals. Such features are essential for peak current mode control and for support of analog comparators.

Note

The name of the sync signal that goes to the CMPSS and GPDAC has been updated from PWMSYNC to EPWMSYNCPER (SYNCPER/PWMSYNCPER/EPWMxSYNCPER) to avoid confusion with the other EPWM sync signals EPWMSYNCI and EPWMSYNCO. For a description of these signals, see Table 26-2.

26.1.1 EPWM Related Collateral

Foundational Materials

- C2000 Academy - EPWM
- *Real-Time Control Reference Guide*
  - Refer to the EPWM section

Getting Started Materials

- *C2000 ePWM Developer's Guide Application Report*
- Enhanced Pulse Width Modulator (ePWM) Training for C2000 MCUs (Video)
- *Flexible PWMs Enable Multi-Axis Drives, Multi-Level Inverters Application Report*
- Getting Started with the C2000 ePWM Module (Video)
- *Using PWM Output as a Digital-to-Analog Converter on a TMS320F280x Digital Signal Control Application Report*
  - Chapters 1 to 6 are Fundamental material, derivations, and explanations that are useful for learning about how PWM can be used to implement a DAC. Subsequent chapters are Getting Started and Expert material for implementing in a system.
- *Using the Enhanced Pulse Width Modulator (ePWM) Module Application Report*

Expert Materials

- C2000 real-time microcontrollers - Reference designs

  - See TI designs related to specific end applications used.
- *CRM/ZVS PFC Implementation Based on C2000 Type-4 PWM Module Application Report*
- *Leverage New Type ePWM Features for Multiple Phase Control Application Report*

26.1.2 Submodule Overview

The ePWM module represents one complete PWM channel composed of two PWM outputs: EPWMxA and EPWMxB. Multiple ePWM modules are instanced within a device as shown in Figure 26-1. Each ePWM instance is identical with one exception. Some instances include a hardware extension that allows more precise control of the PWM outputs. This extension is the high-resolution pulse width modulator (HRPWM) and is described in Section 26.15. See the device data sheet to determine which ePWM instances include this feature. Each ePWM module is indicated by a numerical value starting with 1. For example, ePWM1 is the first instance and ePWM3 is the third instance in the system and ePWMx indicates any instance.

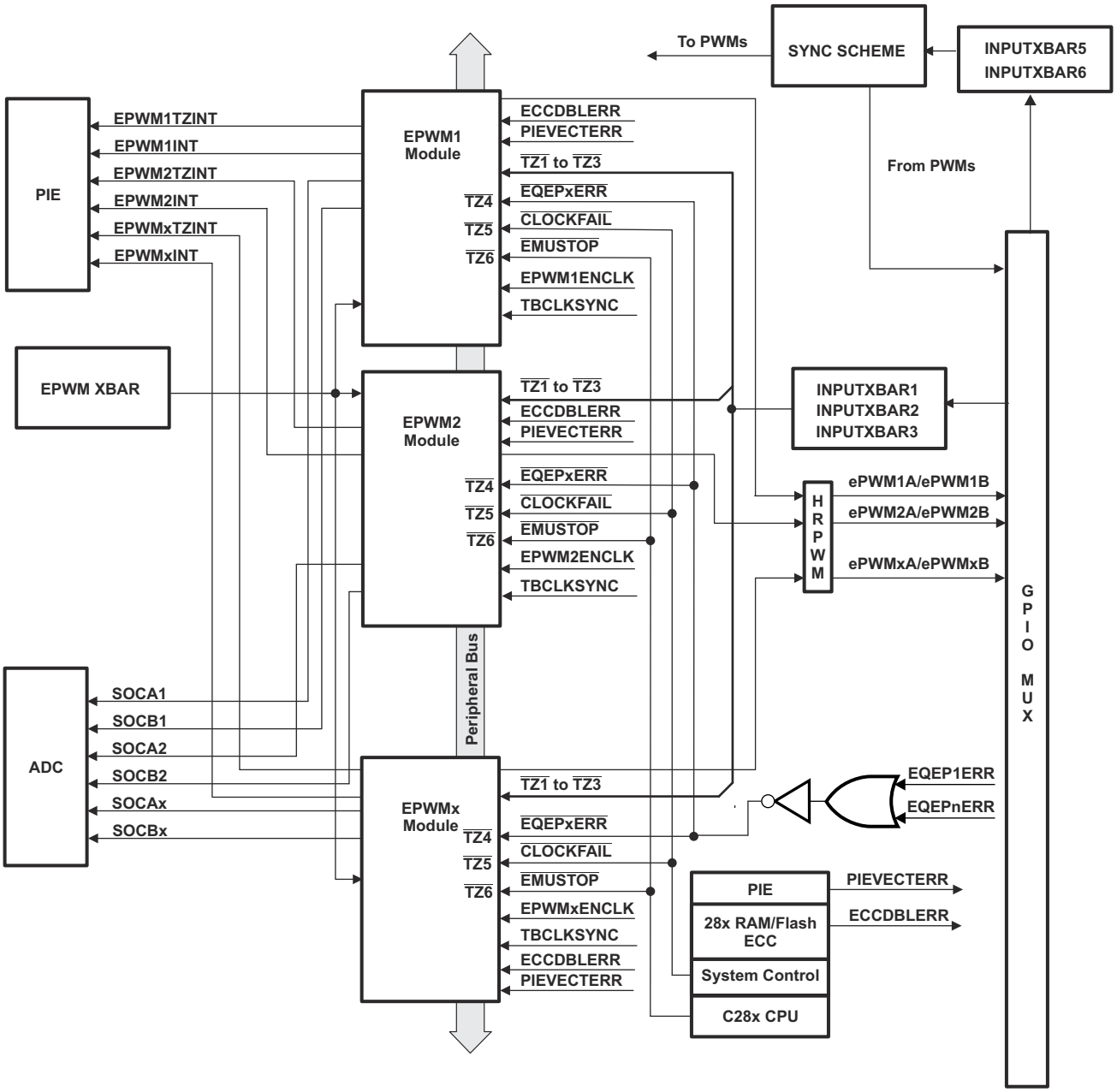
The ePWM modules are chained together by way of a clock synchronization scheme that allows them to operate as a single system when required. Additionally, this synchronization scheme can be extended to the capture peripheral submodules (eCAP). The number of submodules is device-dependent and based on target application needs. Submodules can also operate standalone.

Each ePWM module supports the following features:

- Dedicated 16-bit time-base counter with period and frequency control
- Two PWM outputs (EPWMxA and EPWMxB) that can be used in the following configurations:
  - Two independent PWM outputs with single-edge operation
  - Two independent PWM outputs with dual-edge symmetric operation
  - One independent PWM output with dual-edge asymmetric operation
- Asynchronous override control of PWM signals through software.
- Programmable phase-control support for lag or lead operation relative to other ePWM modules.
- Hardware-locked (synchronized) phase relationship on a cycle-by-cycle basis.
- Dead-band generation with independent rising and falling edge delay control.
- Programmable trip zone allocation of both cycle-by-cycle trip and one-shot trip on fault conditions.
- A trip condition can force either high, low, or high-impedance state logic levels at PWM outputs.
- All events can trigger both CPU interrupts and ADC start of conversion (SOC)
- Programmable event prescaling minimizes CPU overhead on interrupts.
- PWM chopping by high-frequency carrier signal, useful for pulse transformer gate drives.

Each ePWM module is connected to the input/output signals shown in Figure 26-1. The signals are described in detail in subsequent sections.

The order in which the ePWM modules are connected can differ from what is shown in Figure 26-1. See Section 26.4.3.3 for the synchronization scheme for a particular device. Each ePWM module consists of eight submodules and is connected within a system by way of the signals shown in Figure 26-2.



A. This signal exists only on devices with an eQEP submodule.

Figure 26-1. Multiple ePWM Modules

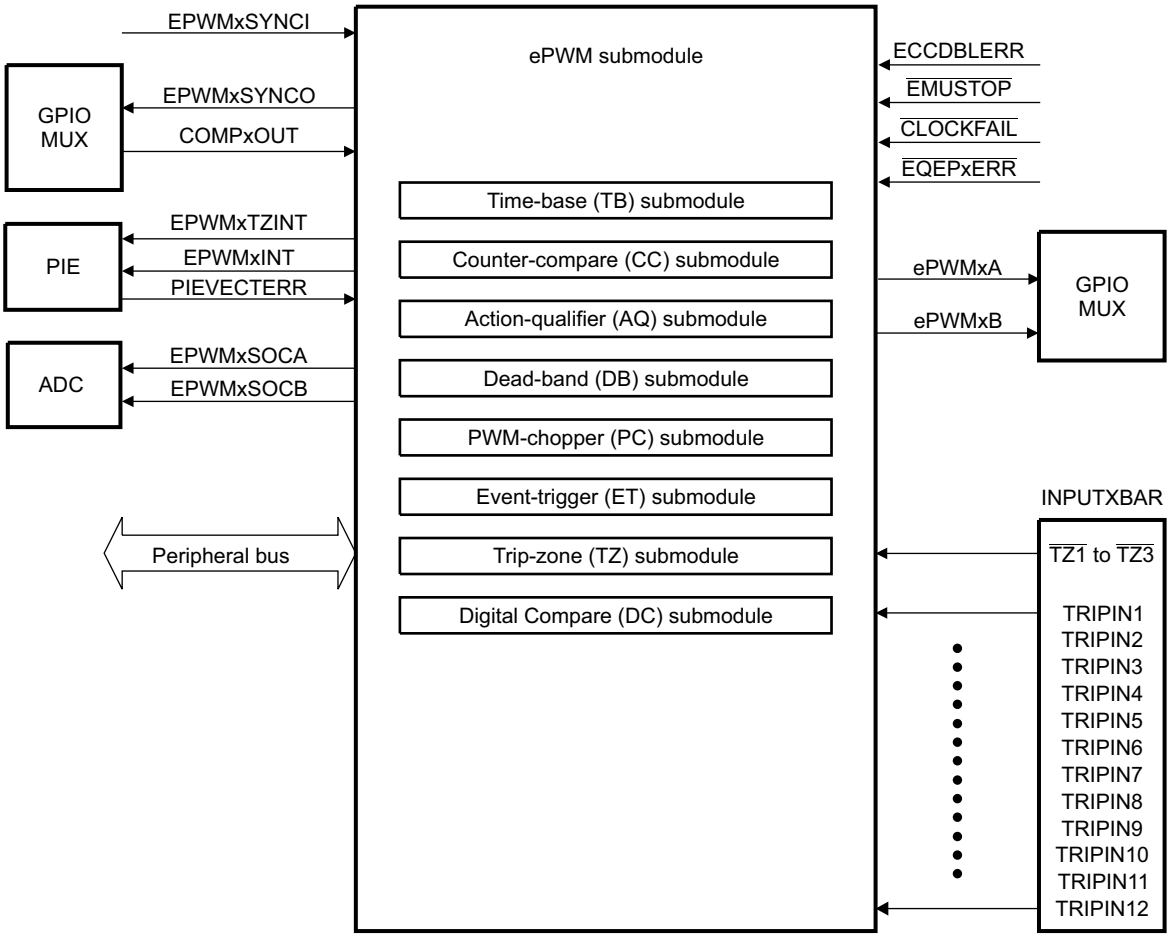


Figure 26-2. Submodules and Signal Connections for an ePWM Module

Figure 26-3 shows more internal details of a single ePWM module. The main signals used by the ePWM module are:

- **PWM output signals (EPWMxA and EPWMxB)**

  The PWM output signals are made available external to the device.

- **Trip-zone signals ($\overline{TZ1}$ to $\overline{TZ6}$)**

  These input signals alert the ePWM module of fault conditions external to the ePWM module. Each submodule on a device can be configured to either use or ignore any of the trip-zone signals. The $\overline{TZ1}$ to $\overline{TZ3}$ trip-zone signals can be configured as asynchronous inputs through the GPIO peripheral using the Input X-BAR logic, refer to Figure 26-51. $\overline{TZ4}$ is connected to an inverted EQEPx error signal (EQEPxERR), which can be generated from any one of the EQEP submodule (for those devices with an EQEP module). $\overline{TZ5}$ is connected to the system clock fail logic, and $\overline{TZ6}$ is connected to the $\overline{EMUSTOP}$ output from the CPU. This allows configuring a trip action when the clock fails or the CPU halts.

- **Time-base synchronization input (EPWMxSYNCI), output (EPWMxSYNCO), and peripheral (EPWMxSYNCPER) signals**

  Each ePWM module can be synchronized with other ePWM modules or other peripherals, using EPWMSYNCINSEL. Each ePWM module can also generate a synchronization output signal. The source of the EPWMxSYNCOUT can be selected and enabled by EPWMSYNCOUTEN and TBCTL2.OSHTSYNCMODE. For more information, see Section 26.4.3.3.

  Each ePWM module also generates another PWMSYNC signal called EPWMxSYNCPER. EPWMxSYNCPER goes to the GPDAC and CMPSS for synchronization purposes. Functionality is configured using the HRPCTL register, but has no relation with the HRPWM. For more information on how EPWMxSYNCPER is used by the GPDAC and CMPSS, see their respective chapters.

- **ADC start-of-conversion signals (EPWMxSOCA and EPWMxSOCB)**

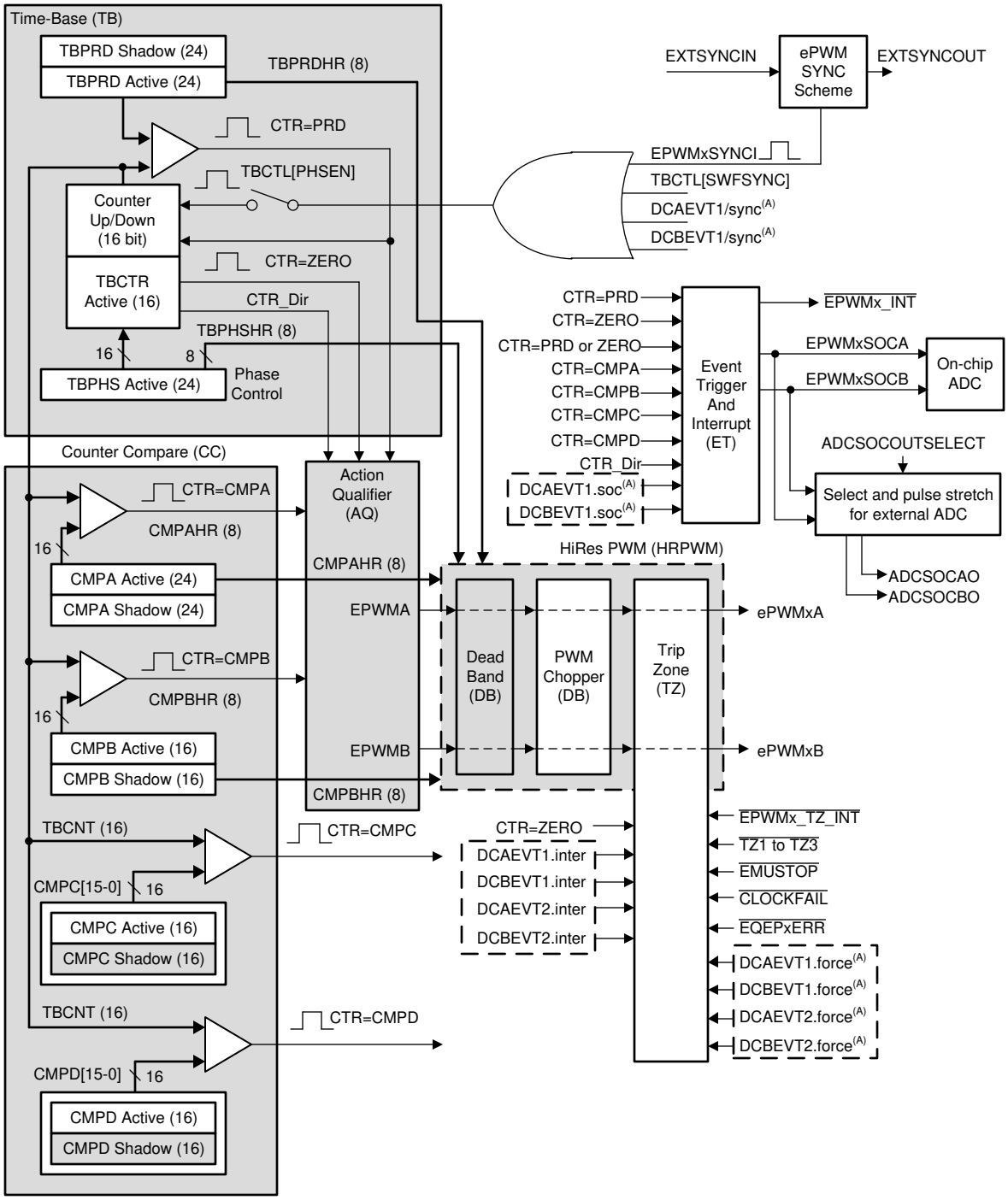
  Each ePWM module has two ADC start of conversion signals. Any ePWM module can trigger a start of conversion. Whichever event triggers the start of conversion is configured in the event-trigger submodule of the ePWM.

- **Comparator output signals (COMPxOUT)**

  Output signals from the comparator module can be fed through the Input X-BAR and EPWM X-BAR to one or all of the 12 trip inputs [TRIPIN1 - TRIPIN12] and in conjunction with the trip zone signals can generate digital compare events.

- **Peripheral bus**

  The peripheral bus is 32-bits wide and allows both 16-bit and 32-bit writes to the ePWM register file.



A. These events are generated by the ePWM Digital Compare (DC) submodule based on the levels of the TRIPIN inputs.

Figure 26-3. ePWM Modules and Critical Internal Signal Interconnects

26.2 Configuring Device Pins

To connect the device input pins to the module, the Input X-BAR and EPWM X-BAR must be used. Some examples of when an external signal can be needed are TZx, TRIPx, and EXTSYNCIN. Any GPIO on the device can be configured as an input. The GPIO input qualification can be set to asynchronous mode by setting the appropriate GPxQSEL register bits to 11b. The internal pullups can be configured in the GPyPUD register. Since the GPIO mode is used, the GPyINV register can invert the signals. Additionally, some TRIPx (TRIP4-12 excluding TRIP6) signals must be routed through the ePWM X-Bar in addition to the Input X-Bar.

The GPIO mux registers must be configured for this peripheral. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

See the *General-Purpose Input/Output (GPIO)* chapter for more details on GPIO mux, GPIO settings, and XBAR configuration.

26.3 ePWM Modules Overview

Eight submodules are included in every ePWM peripheral. Each of these submodules performs specific tasks that can be configured by software.

Table 26-1 lists the key submodules together with a list of their main configuration parameters. For example, if you need to adjust or control the duty cycle of a PWM waveform, see the counter-compare submodule in Section 26.5 for relevant details.

Table 26-1. Submodule Configuration Parameters

Submodule	Configuration Parameter or Option
Time Base (TB)	• Scale the time-base clock (TBCLK) relative to the ePWM clock (EPWMCLK). • Configure the PWM time-base counter (TBCTR) frequency or period. • Set the mode for the time-base counter: – count-up mode: used for asymmetric PWM – count-down mode: used for asymmetric PWM – count-up-and-down mode: used for symmetric PWM • Configure the time-base phase relative to another ePWM module. • Synchronize the time-base counter between modules through hardware or software. • Configure the direction (up or down) of the time-base counter after a synchronization event. • Simultaneous writes to the TBPRD registers on all PWM's corresponding to the configuration on EPWMXLINK. • Configure how the time-base counter behaves when the device is halted by an emulator. • Specify the source for the synchronization output of the ePWM module • Configure one shot and global load of registers in this module.
Counter Compare (CC)	• Specify the PWM duty cycle for output EPWMxA and output EPWMxB • Specify the time at which switching events occur on the EPWMxA or EPWMxB output • Specify the programmable delay for interrupt and SOC generation with additional comparators • Simultaneous writes to the CMPA, CMPB, CMPC, CMPD registers on all PWM's corresponding to the configuration on EPWMXLINK. • Configure one shot and global load of registers in this module.

Table 26-1. Submodule Configuration Parameters (continued)

Submodule	Configuration Parameter or Option
Action Qualifier (AQ)	• Specify the type of action taken when a time-base counter-compare, trip-zone submodule, or comparator event occurs: – No action taken – Output EPWMxA and EPWMxB switched high – Output EPWMxA and EPWMxB switched low – Output EPWMxA and EPWMxB toggled • Force the PWM output state through software control • Configure and control the PWM dead band through software • Configure one shot and global load of registers in this module.
Dead-Band Generator (DB)	• Control of traditional complementary dead-band relationship between upper and lower switches • Specify the output rising-edge-delay value • Specify the output falling-edge delay value • Bypass the dead-band module entirely. In this case the PWM waveform is passed through without modification. • Option to enable half-cycle clocking for double resolution. • Allow ePWMxB phase shifting with respect to the ePWMxA output. • Configure one shot and global load of registers in this module.
PWM Chopper (PC)	• Create a chopping (carrier) frequency. • Pulse width of the first pulse in the chopped pulse train. • Duty cycle of the second and subsequent pulses. • Bypass the PWM chopper module entirely. In this case the PWM waveform is passed through without modification.
Trip Zone (TZ)	• Configure the ePWM module to react to one, all, or none of the trip-zone signals or digital compare events. • Specify the trip action taken when a fault occurs: – Force EPWMxA and EPWMxB high – Force EPWMxA and EPWMxB low – Force EPWMxA and EPWMxB to a high-impedance state – Configure EPWMxA and EPWMxB to ignore any trip condition. • Configure how often the ePWM reacts to each trip-zone signal: – One-shot – Cycle-by-cycle • Enable the trip-zone to initiate an interrupt. • Bypass the trip-zone module entirely. • Programmable option for cycle-by-cycle trip clear • If desired, independently configure trip actions taken when time-base counter is counting down.
Event Trigger (ET)	• Enable the ePWM events that trigger an interrupt. • Enable ePWM events that trigger an ADC start-of-conversion event. • Specify the rate at which events cause triggers (every occurrence or every 2nd or up to 15th occurrence) • Poll, set, or clear event flags
Digital Compare (DC)	• Enables comparator (COMP) module outputs and trip zone signals which are configured using the Input X-BAR to create events and filtered events • Specify event-filtering options to capture TBCTR counter, generate blanking window, or insert delay in PWM output or time-base counter based on captured value.

26.4 Time-Base (TB) Submodule

Each ePWM module has their own time-base submodule that determines all of the event timing for the ePWM module. Built-in synchronization logic allows the time-base of multiple ePWM modules to work together as a single system.

Figure 26-4 illustrates the time-base submodule within the ePWM.

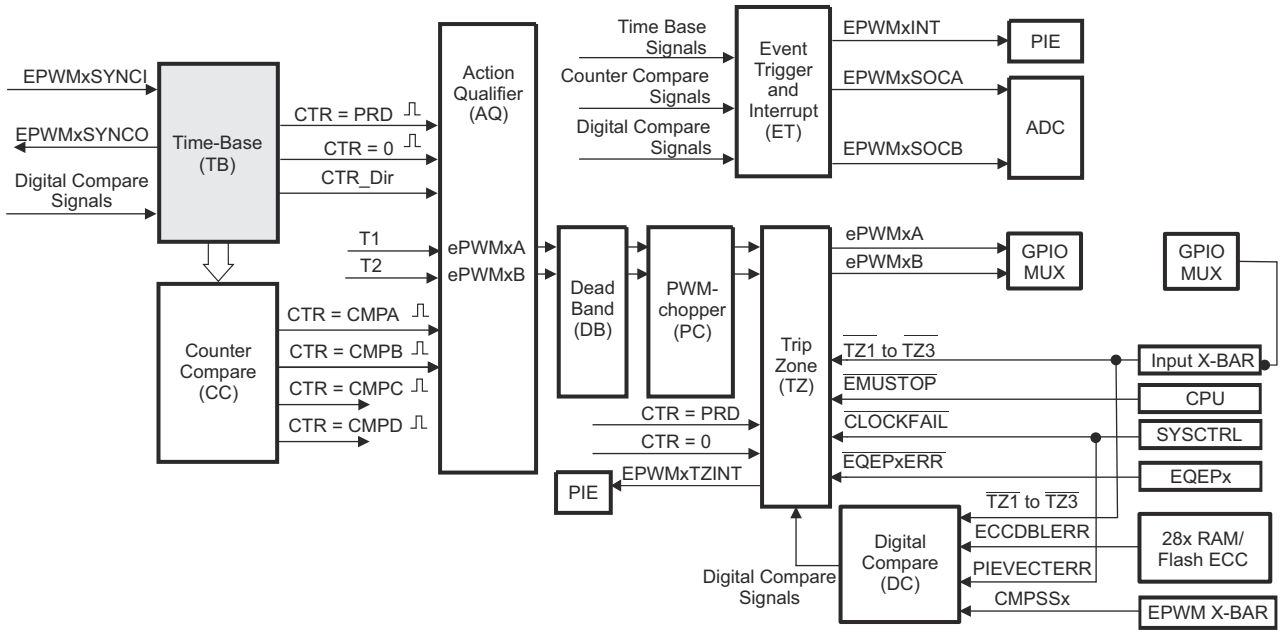


Figure 26-4. Time-Base Submodule

26.4.1 Purpose of the Time-Base Submodule

The time-base submodule can be configured for the following:

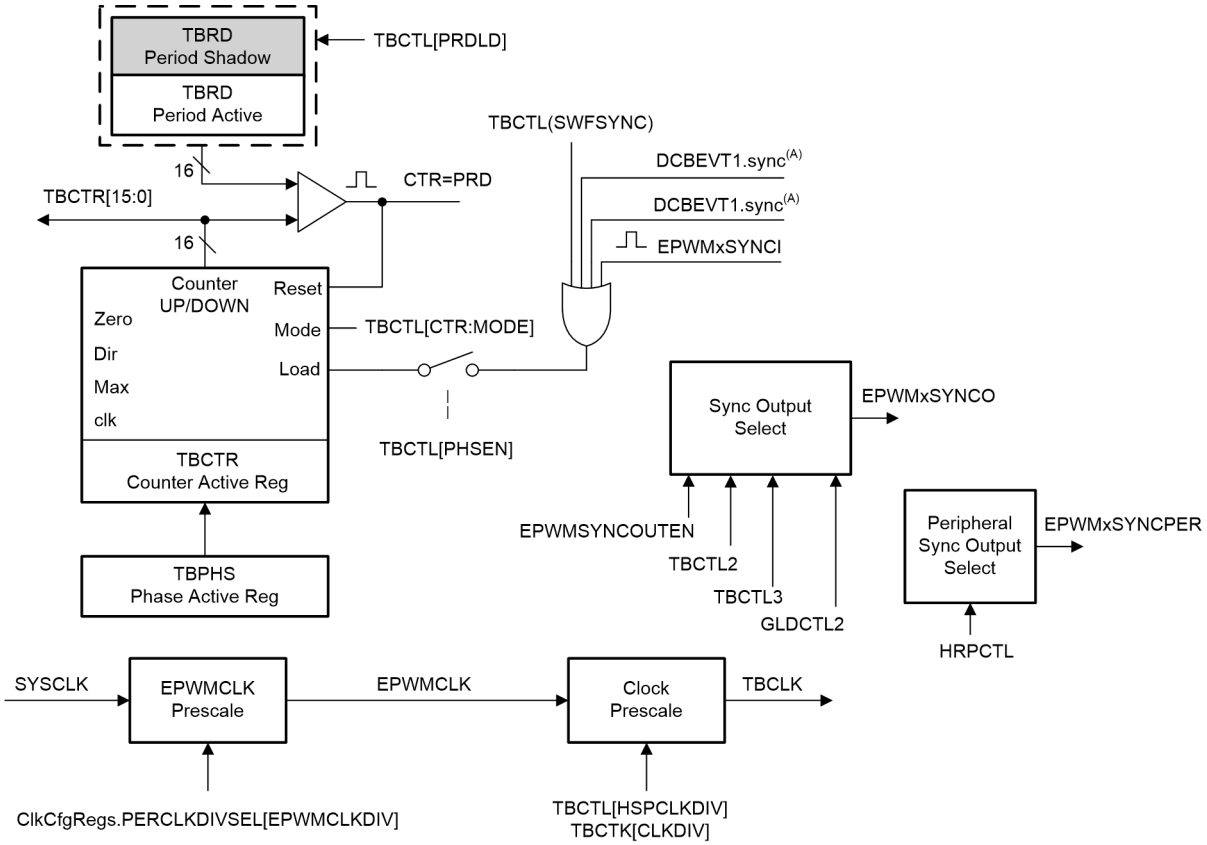
- Specify the ePWM time-base counter (TBCTR) frequency or period to control how often events occur.
- Manage time-base synchronization with other ePWM modules.
- Maintain a phase relationship with other ePWM modules.
- Set the time-base counter to count-up, count-down, or count-up-and-down mode.
- Generate the following events:
  - CTR = PRD: Time-base counter equal to the specified period (TBCTR = TBPRD).
  - CTR = Zero: Time-base counter equal to zero (TBCTR = 0x00).
- Configure the rate of the time-base clock; a prescaled version of the ePWM clock (EPWMCLK). This allows the time-base counter to increment/decrement at a slower rate.

Note

If required by the application code to update the TBCTR value through software while the TBCTR is counting, note that the time-base module needs at least 1 TBCLK cycle for the time-base related events to be realized. Hence, the TBCTR can be written with TBCTR = PRD-1 instead of TBCTR = PRD (in case the counter is counting up) and can be written as TBCTR = 1 instead of TBCTR = 0 (in case the counter is counting down) for the events to be realized.

26.4.2 Controlling and Monitoring the Time-Base Submodule

The block diagram in Figure 26-5 shows the critical signals and registers of the time-base submodule. Table 26-2 provides descriptions of the key signals associated with the time-base submodule.



A. These signals are generated by the digital compare (DC) submodule.

Figure 26-5. Time-Base Submodule Signals and Registers

Table 26-2. Key Time-Base Signals

Signal	Description
EPWMxSYNCI	Time-base synchronization input. Input pulse used to synchronize the time-base counter with the counter of other ePWM modules. For more information on all of the signals available for synchronization, see EPWMSYNCINSEL. For information on the synchronization order of a particular device, see Section 26.4.3.3.
EPWMxSYNCO	Time-base synchronization output. This output pulse is used to synchronize the counter of other ePWM modules. Using EPWMSYNCOUTEN, TBCTL2, TBCTL3 and GLDCTL2, the source of the output pulse is selected.
EPWMxSYNCPER	Time-base peripheral synchronization output. This output signal is used to synchronize the GPDAC and CMPSS to the EPWM. The output signal can be configured using the HRPCTL register. Note that this signal has no relation with the HRPWM.
CTR = PRD	Time-base counter equal to the specified period. This signal is generated whenever the counter value is equal to the active period register value. That is when TBCTR = TBPRD.
CTR = Zero	Time-base counter equal to zero This signal is generated whenever the counter value is zero. That is when TBCTR equals 0x00.
CTR = CMPB	Time-base counter equal to active counter-compare B register (TBCTR = CMPB). This event is generated by the counter-compare submodule and used by the synchronization out logic
CTR_dir	Time-base counter direction. Indicates the current direction of the ePWM's time-base counter. The signal is high when the counter is increasing and the signal is low when the counter is decreasing.
CTR_max	Time-base counter equal max value. (TBCTR = 0xFFFF) Generated event when the TBCTR value reaches the maximum value. This signal is only used only as a status bit
TBCLK	Time-base clock. This is a prescaled version of the ePWM clock (EPWMCLK) and is used by all submodules within the ePWM. This clock determines the rate at which time-base counter increments or decrements.

26.4.3 Calculating PWM Period and Frequency

The frequency of PWM events is controlled by the time-base period (TBPRD) register and the mode of the time-base counter. Figure 26-6 shows the period (T_{pwm}) and frequency (F_{pwm}) relationships for the up-count, down-count, and up-down-count time-base counter modes when the period is set to 4 (TBPRD = 4). The time increment for each step is defined by the time-base clock (TBCLK) which is a prescaled version of the ePWM clock (EPWMCLK).

The time-base counter has three modes of operation selected by the time-base control register (TBCTL):

- **Up-Down Count Mode:** In up-down count mode, the time-base counter starts from zero and increments until the period (TBPRD) value is reached. When the period value is reached, the time-base counter then decrements until the counter reaches zero. At this point, the counter repeats the pattern and begins to increment.
- **Up-Count Mode:** In up-count mode, the time-base counter starts from zero and increments until the counter reaches the value in the period register (TBPRD). When the period value is reached, the time-base counter resets to zero and begins to increment once again.
- **Down-Count Mode:** In down-count mode, the time-base counter starts from the period (TBPRD) value and decrements until the counter reaches zero. When the counter reaches zero, the time-base counter is reset to the period value and begins to decrement once again.

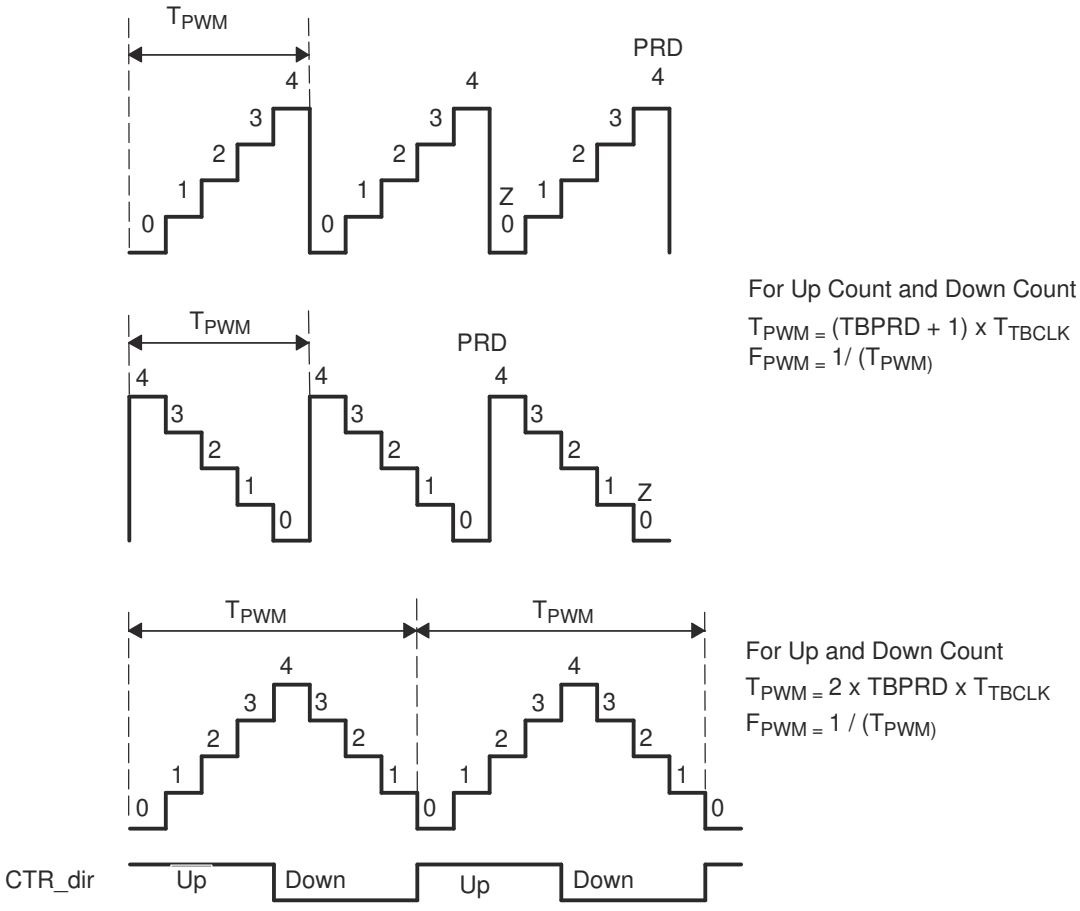


Figure 26-6. Time-Base Frequency and Period

26.4.3.1 Time-Base Period Shadow Register

The time-base period register (TBPRD) has a shadow register. Shadowing allows the register update to be synchronized with the hardware. The following definitions are used to describe all shadow registers in the ePWM module:

- **Active Register:** The active register controls the hardware and is responsible for actions that the hardware causes or invokes.
- **Shadow Register:** The shadow register buffers provide a temporary holding location for the active register and have no direct effect on any control hardware. At a strategic point in time, the shadow register content is transferred to the active register. This prevents corruption or spurious operation due to the register being asynchronously modified by software.

The memory address of the shadow period register is the same as the active register. Which register is written to or read from is determined by the TBCTL[PRDLD] bit. This bit enables and disables the TBPRD shadow register as follows:

- **Time-Base Period Shadow Mode:** The TBPRD shadow register is enabled when TBCTL[PRDLD] = 0. Reads from and writes to the TBPRD memory address go to the shadow register. The shadow register contents are transferred to the active register (TBPRD (Active) ← TBPRD (shadow)) when the time-base counter equals zero (TBCTR = 0x00) and/or a sync event as determined by the TBCTL2[PRDLDSYNC] bit. The PRDLDSYNC bit is valid only if TBCTL[PRDLD] = 0. By default the TBPRD shadow register is enabled. The sources for the SYNC input is explained in Section 26.4.3.3.

  The global load control mechanism can also be used with the time-base period register by configuring the appropriate bits in the global load configuration register (GLDCFG). When global load mode is selected the transfer of contents from shadow register to active register, for all registers that have this mode enabled, occurs at the same event as defined by the configuration bits in Global Shadow to Active Load Control Register (GLDCTL). Global load control mechanism is explained in Section 26.4.7.
- **Time-Base Period Immediate Load Mode:** If immediate load mode is selected (TBCTL[PRDLD] = 1), then a read from or a write to the TBPRD memory address goes directly to the active register.

26.4.3.2 Time-Base Clock Synchronization

The TBCLKSYNC bit in the peripheral clock enable registers allows all users to globally synchronize all enabled ePWM modules to the time-base clock (TBCLK). When set, all enabled ePWM module clocks are started with the first rising edge of TBCLK aligned. For perfectly synchronized TBCLKs, the prescalers for each ePWM module must be set identically.

The proper procedure for enabling ePWM clocks is as follows:

1. Enable ePWM module clocks in the PCLKCRx register
2. Set TBCLKSYNC= 0
3. Configure ePWM modules
4. Set TBCLKSYNC= 1

In a multi-core environment, GTBCLKSYNC can be used to override the core-specific TBCLKSYNC. When GTBCLKSYNC is set, TBCLKSYNC is ignored in all cores and therefore clearing or setting the TBCLKSYNC has no affect. If this feature is not required, GTBCLKSYNC should be cleared. In a multi-core environemt where different ePWM modules are assigned to different cores, the GTBCLKSYNC bit can be used to enable and disable the Time-Base clock of all ePWM modules simultaneously.

26.4.3.3 Time-Base Counter Synchronization

The ePWM synchronization scheme allows for increased flexibility of synchronization of the ePWM modules. Each ePWM module has a synchronization input (SYNCI), a synchronization output (SYNCO) and a peripheral synchronization output (SYNCPER). In Figure 26-7, EXTSYNCIN1 is sourced from INPUTXBAR5 and EXTSYNCIN2 is sourced from INPUTXBAR6, which can be configured to select any GPIO as the synchronization input. Refer to Table 26-3 for a list of all sync inputs including INPUTXBAR5 and INPUTXBAR6. Figure 26-8 shows the sources that can be used for EXTSYNCOUT.

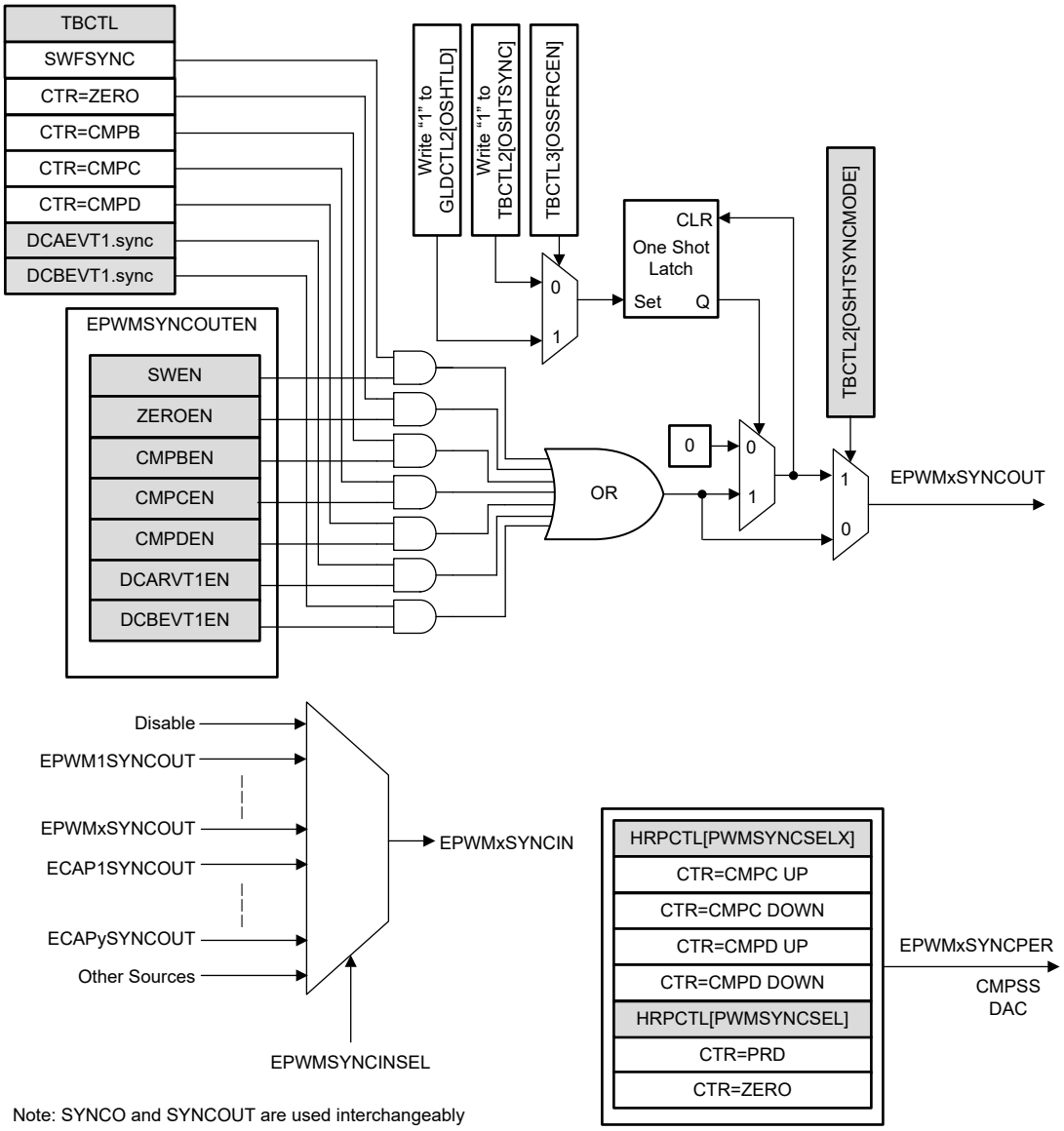


Figure 26-7. Time-Base Counter Synchronization Scheme

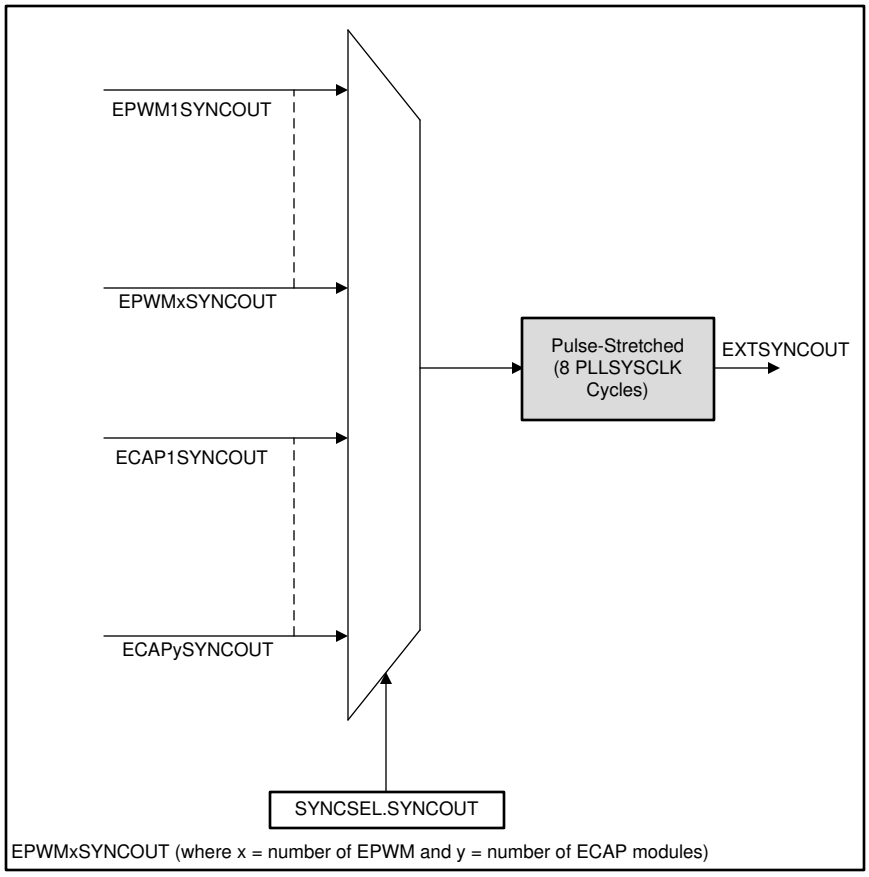


Figure 26-8. ePWM External SYNC Output

Note

See the data sheet for the number of ePWM and eCAP modules available on your specific device.

Each ePWM module can be configured to use or ignore the synchronization input. If the TBCTL[PHSEN] bit is set, then the time-base counter (TBCTR) of the ePWM module is automatically loaded with the phase register (TBPHS) contents when one of the following conditions occur:

- **EPWMxSYNCI: Synchronization Input Pulse:** The value of the phase register is loaded into the counter register when an input synchronization pulse is detected (TBPHS → TBCTR). This operation occurs on the next valid time-base clock (TBCLK) edge.
- **Software Forced Synchronization Pulse:** Writing a 1 to the TBCTL[SWFSYNC] control bit invokes a software forced synchronization. This pulse is ORed with the synchronization input signal, and therefore has the same effect as a pulse on EPWMxSYNCI.
- **Digital Compare Event Synchronization Pulse:** DCAEVT1 and DCBEVT1 digital compare events can be configured to generate synchronization pulses which have the same affect as EPWMxSYNCI.

Note

If the EPWMxSYNCI signal is held high, the sync does not continuously occur. The EPWMxSYNCI is rising edge activated.

This feature enables the ePWM module to be automatically synchronized to the time base of another ePWM module. Lead or lag phase control can be added to the waveforms generated by different ePWM modules to synchronize them. In up-down-count mode, the TBCTL[PHSDIR] bit configures the direction of the time-base counter immediately after a synchronization event. The new direction is independent of the direction prior to the synchronization event. The PHSDIR bit is ignored in count-up or count-down modes. See Figure 26-9 through Figure 26-12 for examples.

Clearing the TBCTL[PHSEN] bit configures the ePWM to ignore the synchronization input pulse.

26.4.3.4 ePWM SYNC Selection

Table 26-3 specifies the sources for the ePWM SYNC input and output.

Table 26-3. ePWM SYNC Selection

EPWMSYNCINSEL.SEL, ECAPSYNCINSEL.SEL	SYNC Source
0	Reserved
1	EPWM1_SYNCOUT
2	EPWM2_SYNCOUT
3	EPWM3_SYNCOUT
4	EPWM4_SYNCOUT
5	EPWM5_SYNCOUT
6	EPWM6_SYNCOUT
7	EPWM7_SYNCOUT
8	EPWM8_SYNCOUT
9	EPWM9_SYNCOUT
10	EPWM10_SYNCOUT
11	EPWM11_SYNCOUT
12	EPWM12_SYNCOUT
13	EPWM13_SYNCOUT
14	EPWM14_SYNCOUT
15	EPWM15_SYNCOUT
16	EPWM16_SYNCOUT
17	ECAP1_SYNCOUT
18	ECAP2_SYNCOUT
19	ECAP3_SYNCOUT
20	ECAP4_SYNCOUT

Table 26-3. ePWM SYNC Selection (continued)

EPWMSYNCINSEL.SEL, ECAPSYNCINSEL.SEL	SYNC Source
21	ECAP5_SYNCOUT
22	ECAP6_SYNCOUT
23	ECAP7_SYNCOUT
24	INPUTXBAR5
25	INPUTXBAR6
26	ECAT_SYNC0
27	ECAT_SYNC1
28-31	Reserved

26.4.4 Phase Locking the Time-Base Clocks of Multiple ePWM Modules

The TBCLKSYNC bit can be used to globally synchronize the time-base clocks of all enabled ePWM modules on a device. This bit is part of the device's clock enable registers and is described in the *System Control and Interrupts* section of this manual. When TBCLKSYNC = 0, the time-base clock of all ePWM modules is stopped (default). When TBCLKSYNC = 1, all ePWM time-base clocks are started with the rising edge of TBCLK aligned. For perfectly synchronized TBCLKs, the prescaler bits in the TBCTL register of each ePWM module must be set identically. The proper procedure for enabling the ePWM clocks is:

1. Enable the individual ePWM module clocks. This is described in the *System Control and Interrupts* chapter.
2. Set TBCLKSYNC= 0. This stops the time-base clock within any enabled ePWM module.
3. Configure the prescaler values and desired ePWM modes.
4. Set TBCLKSYNC = 1.

26.4.5 Simultaneous Writes to TBPRD and CMPx Registers Between ePWM Modules

For variable frequency applications, there is a need for simultaneous writes of TBPRD and CMPx registers between ePWM modules. This prevents situations where a CTR = 0 or CTR = PRD pulse forces a shadow to active load of these registers before all registers are updated between ePWM modules (resulting in some registers being loaded from new shadow values while others are loaded from old shadow values). To support this, an ePWM register linking scheme for TBPRD:TBPRDHR, CMPA:CMPAHR, CMPB:CMPBHR, CMPC, and CMPD registers between PWM modules has been added.

Refer to the register description for EPWMXLINK to see the linked register bit-field values for corresponding ePWM. An example of using the EPWMXLINK is linking ePWM2 CMPA with CMPA of ePWM1 through ePWM2's EPWMXLINK[CMPALINK] register bit-field. In this case, a write to CMPA of ePWM1 also changes the CMPA value for ePWM2.

26.4.6 Time-Base Counter Modes and Timing Waveforms

The time-base counter operates in one of four modes:

- Up-count mode that is asymmetrical
- Down-count mode that is asymmetrical
- Up-down-count that is symmetrical
- Frozen where the time-base counter is held constant at the current value

To illustrate the operation of the first three modes, the following timing diagrams show when events are generated and how the time-base responds to an EPWMxSYNCI signal.

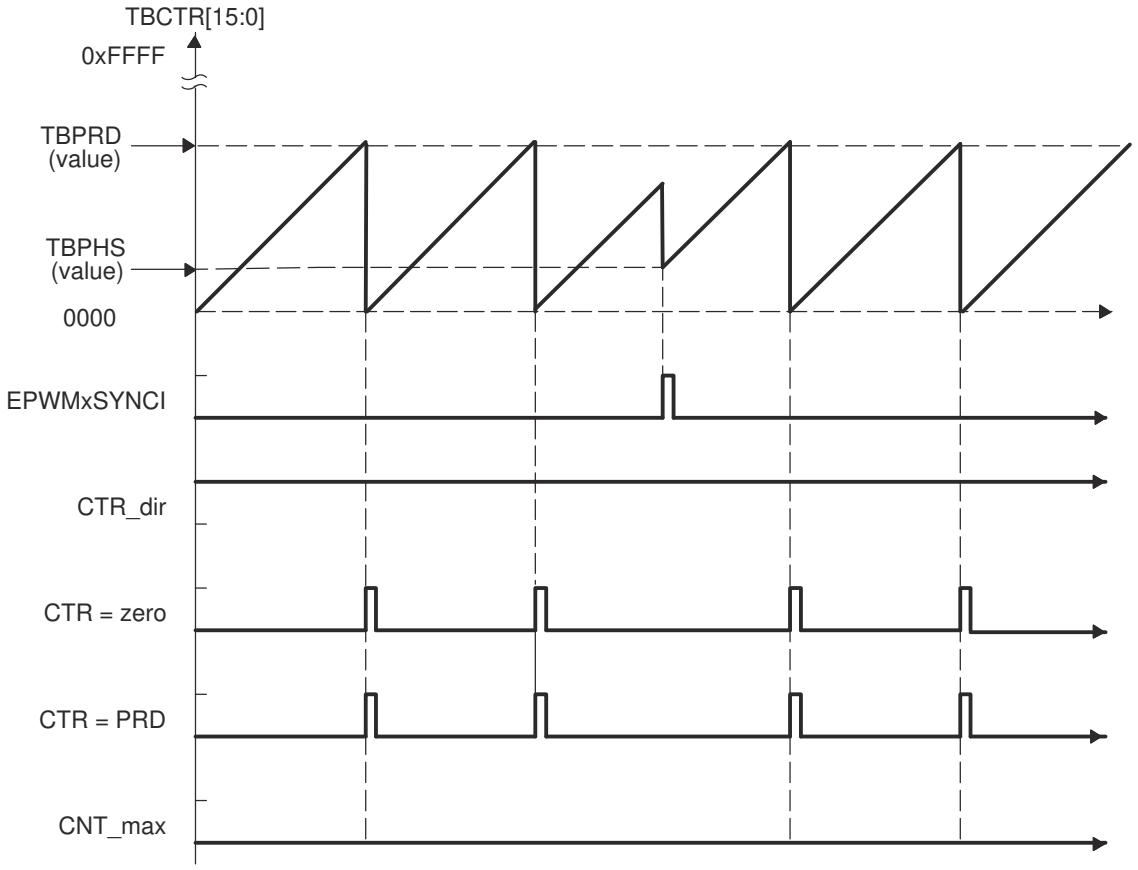


Figure 26-9. Time-Base Up-Count Mode Waveforms

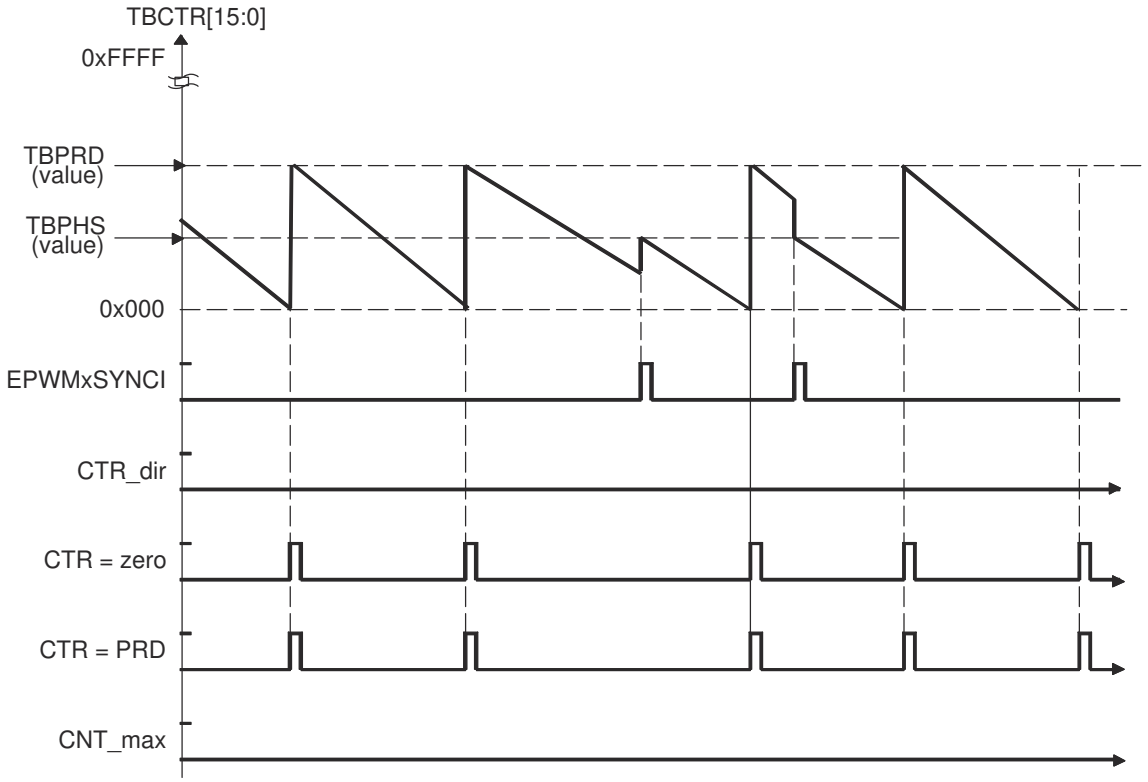


Figure 26-10. Time-Base Down-Count Mode Waveforms

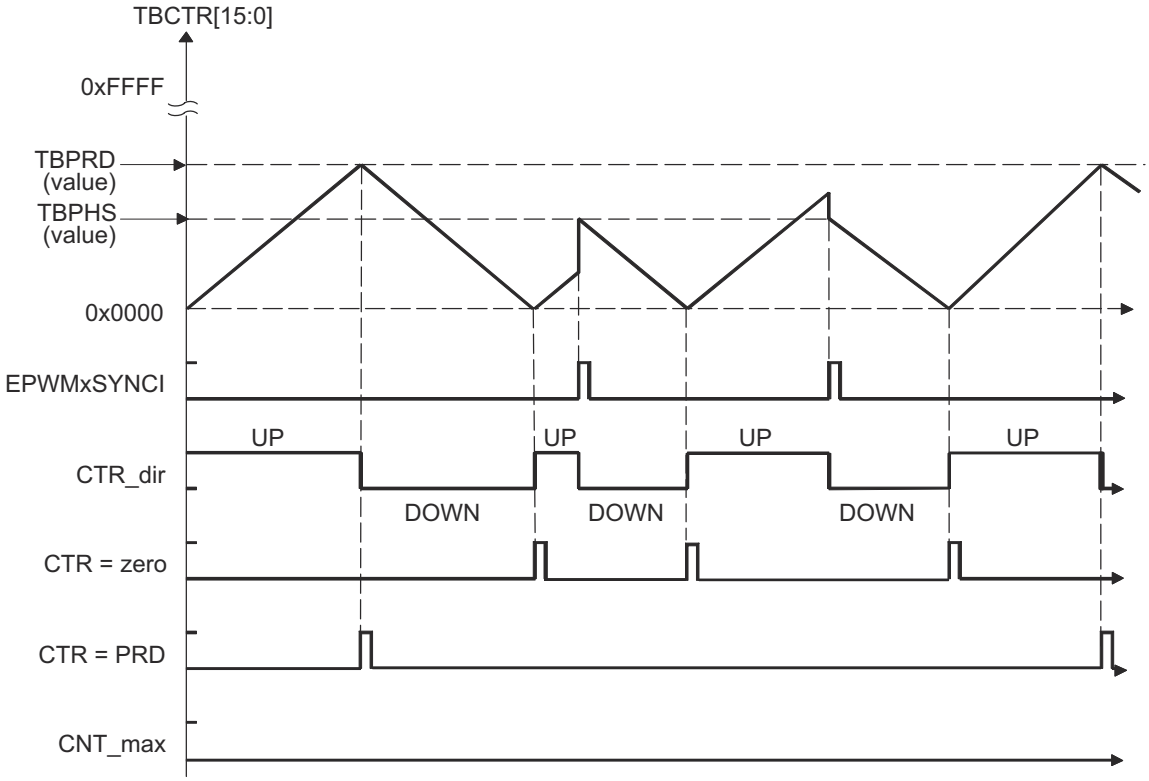


Figure 26-11. Time-Base Up-Down-Count Waveforms, TBCTL[PHSDIR = 0] Count Down On Synchronization Event

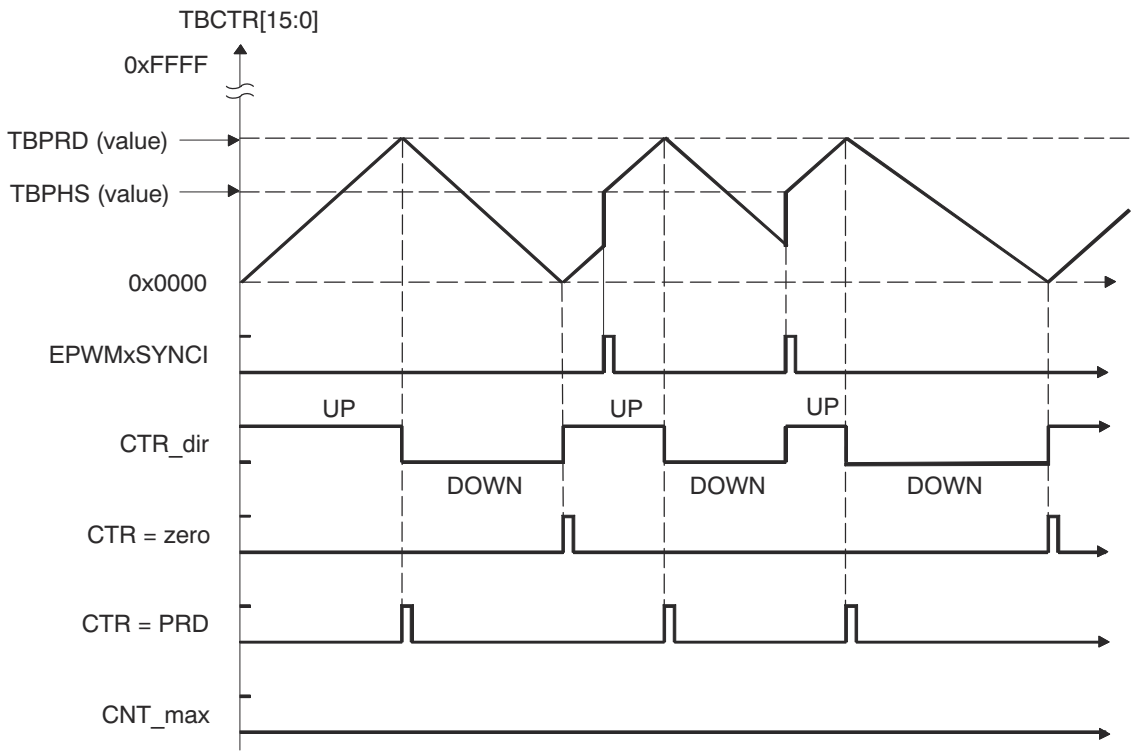


Figure 26-12. Time-Base Up-Down Count Waveforms, TBCTL[PHSDIR = 1] Count Up On Synchronization Event

26.4.7 Global Load

Figure 26-13 shows the signals and registers associated with the global load feature.

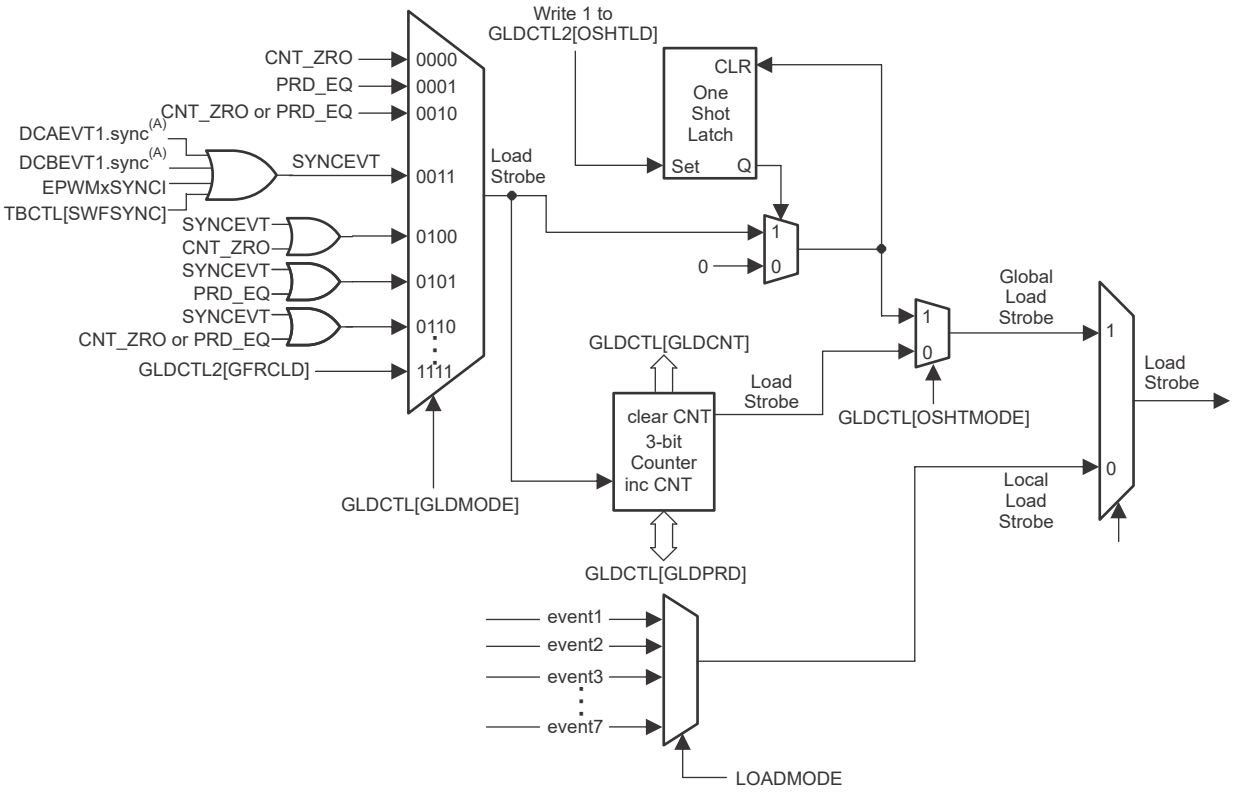


Figure 26-13. Global Load: Signals and Registers

Note

The SYNCEVT signal is only propagated through when PHSEN is SET.

When this feature is enabled, the transfer of contents from the shadow register to the active register, for all registers that have this mode enabled, occurs at the same event as defined by the configuration bits in Global Shadow to Active Load Control Register (GLDCTL[GLDMODE]). When GLDCTL[GLD] = 1, shadow to active load event selection bits for individual shadowed registers are ignored and global load mode takes effect for the corresponding registers enabled by GLDCFG[REGx].

When GLDCTL[GLD] = 1 and GLDCFG[REGx] = 0, global load mode does not affect the corresponding register (REGx). Shadow to active load event selection bits for individual shadowed registers decide how the transfer of contents from shadow register to active register takes place.

26.4.7.1 Global Load Pulse Pre-Scalar

This feature provides the capability to choose shadow to active transfers to happen once in 'N' occurrences of selected global load pulse (GLDCTL[GLDMODE]). This pre-scale functionality is not available for registers that cannot or are not configured to use the global load mechanism (that is, GLDCTL[GLD] = 0 or GLDCFG[REGx] = 0).

26.4.7.2 One-Shot Load Mode

This feature allows users to cause the shadow register to active register transfers to occur once. When GLDCTL2[OSHTLD] = 1 the shadow to active register transfer, for registers that are configured to use the global load mechanism, takes place on the event selected by GLDCTL[GLDMODE].

Software force loading of contents from shadow register to active register is possible by using GLDCTL2[GFRCLD]. The GLDCTL2 register can also be linked across multiple PWM modules by using EPWMXLINK[GLDCTL2LINK]. This, along with the one-shot load mode feature discussed above, provides a method to correctly update multiple PWM registers in one or more PWM modules at certain PWM events or, if desired, in the same clock cycle. This is very useful in variable frequency applications and/or multi-phase interleaved applications.

Note

One-shot load mode must not be used when high-resolution mode is enabled.

26.4.7.3 One-Shot Sync Mode

To enable the one-shot sync mode to generate a SYNCOUT pulse, configure the TBCTL2[OSHTSYNCMODE] bit and set the TBCTL2[OSHTSYNC] bit as shown in Figure 26-14.

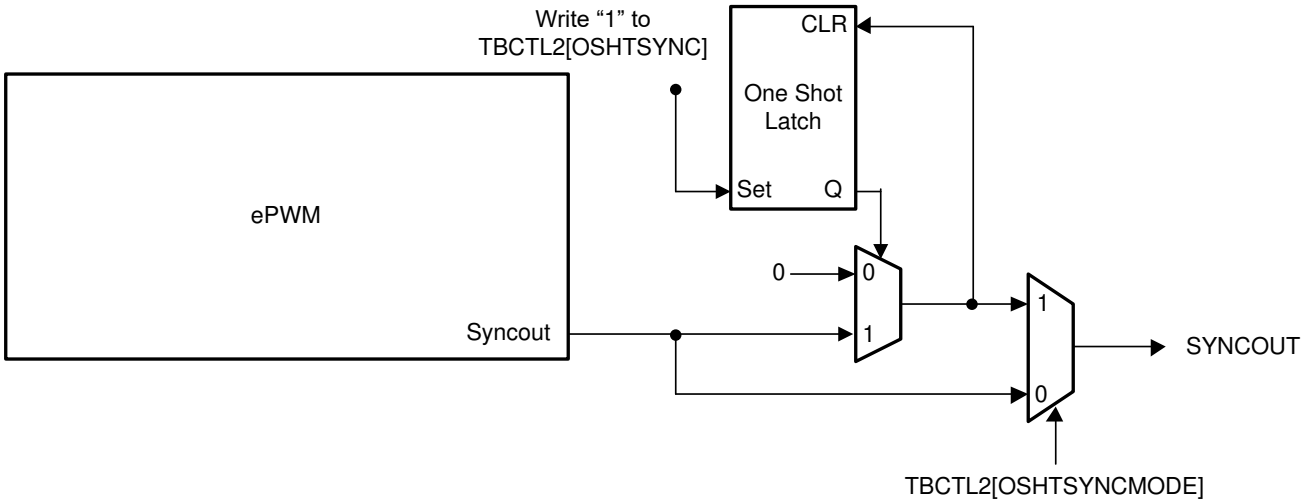


Figure 26-14. One-Shot Sync Mode

26.5 Counter-Compare (CC) Submodule

Figure 26-15 illustrates the counter-compare submodule within the ePWM.

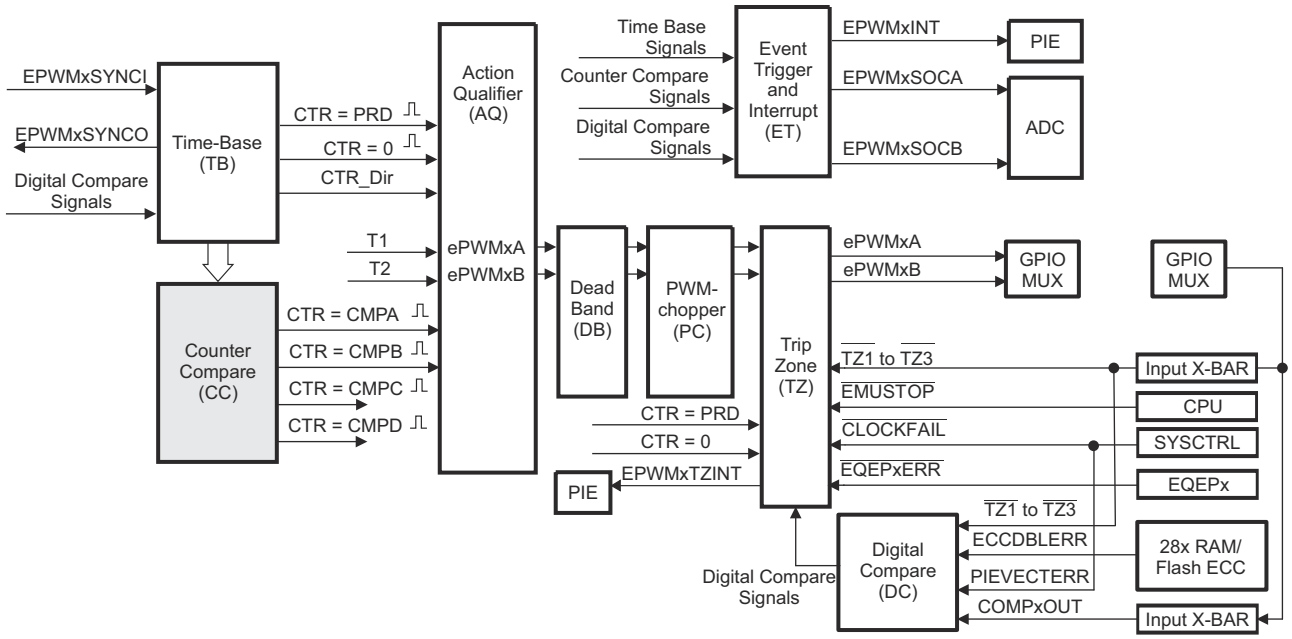


Figure 26-15. Counter-Compare Submodule

26.5.1 Purpose of the Counter-Compare Submodule

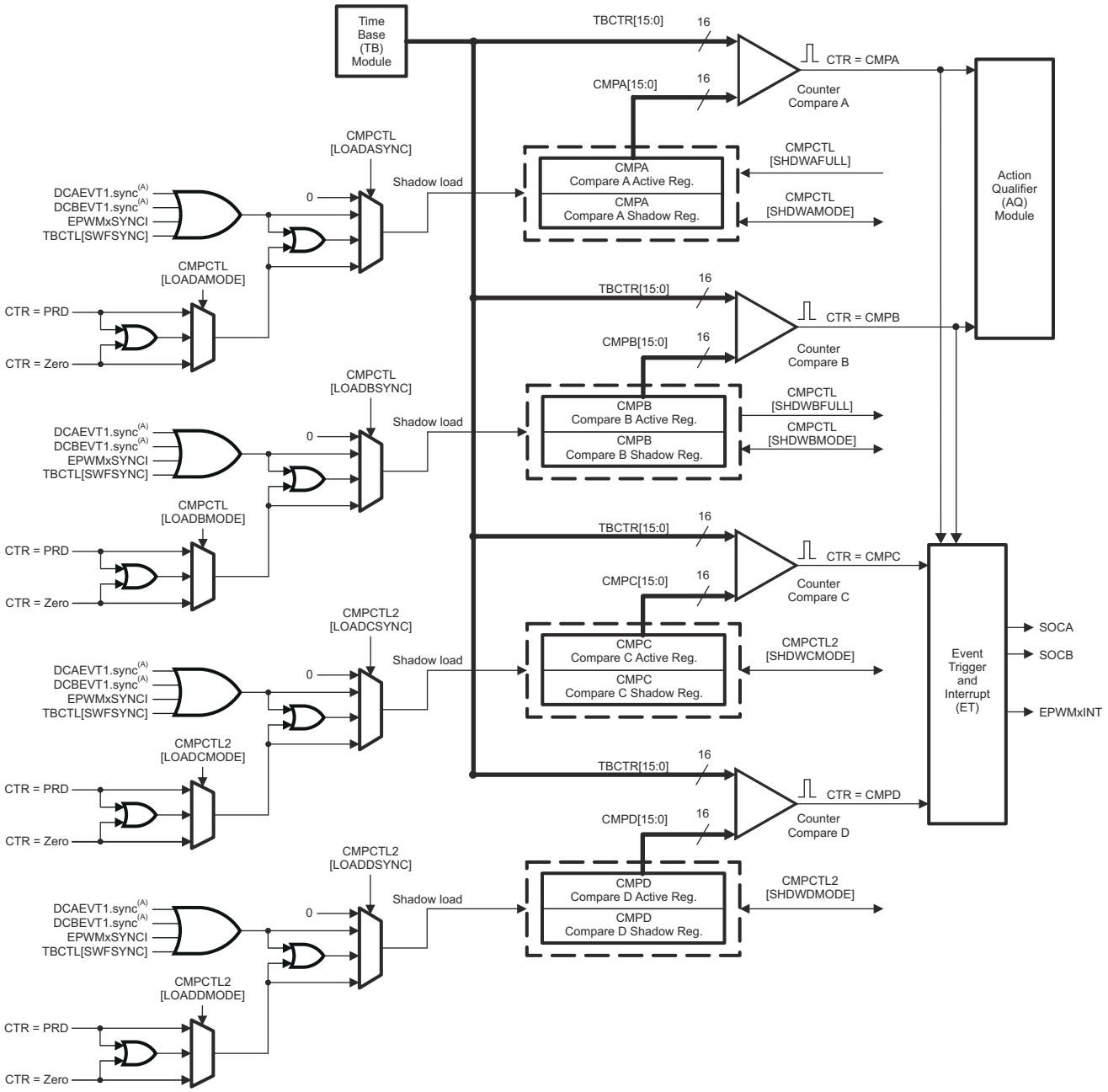
The counter-compare submodule takes as input the time-base counter value. This value is continuously compared to the counter-compare A (CMPA), counter-compare B (CMPB), counter-compare C (CMPC), and counter-compare D (CMPD) registers. When the time-base counter is equal to one of the compare registers, the counter-compare unit generates an appropriate event.

The counter-compare:

- Generates events based on programmable time stamps using the CMPA, CMPB, CMPC, and CMPD registers:
  - CTR = CMPA: Time-base counter equals counter-compare A register (TBCTR = CMPA)
  - CTR = CMPB: Time-base counter equals counter-compare B register (TBCTR = CMPB)
  - CTR = CMPC: Time-base counter equals counter-compare C register (TBCTR = CMPC)
  - CTR = CMPD: Time-base counter equals counter-compare D register (TBCTR = CMPD)
- Controls the PWM duty cycle, if the action-qualifier submodule is configured appropriately using counter-compare A (CMPA) and counter-compare B (CMPB)
- Shadows new compare values to prevent corruption or glitches during the active PWM cycle

26.5.2 Controlling and Monitoring the Counter-Compare Submodule

The counter-compare submodule operation is shown in Figure 26-16.



A. These events are generated by the ePWM digital compare (DC) submodule based on the levels of the TRIPIN inputs (for example, CMPSSx and $\overline{TZ}$ signals).

Figure 26-16. Detailed View of the Counter-Compare Submodule

26.5.3 Operational Highlights for the Counter-Compare Submodule

The counter-compare submodule is responsible for generating events that can be used in the action-qualifier and event-trigger submodules. There are four independent compare events:

1. CTR = CMPA: Time-base counter equal to counter-compare A register (TBCTR = CMPA).
2. CTR = CMPB: Time-base counter equal to counter-compare B register (TBCTR = CMPB).
3. CTR = CMPC: Time-base counter equal to counter-compare C register (TBCTR = CMPC). This event can be used to generate an event in the event trigger submodule only.
4. CTR = CMPD: Time-base counter equal to counter-compare D register (TBCTR = CMPD). This event can be used to generate an event in the event trigger submodule only

For up-count or down-count mode, each event occurs only once per cycle. For up-down count mode, each event occurs twice per cycle if the compare value is between 0x00-TBPRD; and once per cycle if the compare value is equal to 0x00 or equal to TBPRD. These events are applies to the action-qualifier submodule where the events are qualified by the counter direction and converted into actions if enabled. Refer to Section 26.6.1 for more details.

The counter-compare registers CMPA and CMPB each have an associated shadow register. Shadowing provides a way to keep updates to the registers synchronized with the hardware. When shadowing is used, updates to the active registers only occur at strategic points. This prevents corruption or spurious operation due to the register being asynchronously modified by software. The memory address of the active register and the shadow register is identical. The register that is written to or read from is determined by the CMPCTL[SHDWAMODE] and CMPCTL[SHDWBMODE] bits. These bits enable and disable the CMPC shadow register and CMPD shadow register, respectively. The behavior of the two load modes is:

Shadow Mode:

The shadow mode for the CMPA is enabled by clearing the CMPCTL[SHDWAMODE] bit and the shadow register for CMPB is enabled by clearing the CMPCTL[SHDWBMODE] bit. Shadow mode is enabled by default for both CMPA and CMPB.

If the shadow register is enabled then the content of the shadow register is transferred to the active register on one of the following events as specified by the CMPCTL[LOADAMODE], CMPCTL[LOADBMODE], CMPCTL[LOADASYNC], and CMPCTL[LOADBSYNC] register bits:

- CTR = PRD: Time-base counter equal to the period (TBCTR = TBPRD).
- CTR = Zero: Time-base counter equal to zero (TBCTR = 0x00)
- Both CTR = PRD and CTR = Zero
- SYNC event caused by DCAEVT1 or DCBEVT1 or EPWMxSYNCI or TBCTL[SWFSYNC]
- Both SYNC event or a selection made by LOADAMODE/LOADBMODE

Only the active register contents are used by the counter-compare submodule to generate events to be sent to the action-qualifier.

Note

Refer to Section 26.6.5 for valid configurations of CMPA/CMPB and LOADAMODE/LOADBMODE.

Immediate Load Mode:

If the immediate load mode is selected (that is, CMPCTL[SHDWAMODE] = 1 or CMPCTL[SHDWBMODE] = 1), then a read from or a write to the register goes directly to the active register.

Additional Comparators

The counter-compare submodule on ePWMs type 2 and later are responsible for generating two additional independent compare events based on two compare registers, which is fed to Event Trigger submodule:

1. CTR = CMPC: Time-base counter equal to counter-compare C register (TBCTR = CMPC).
2. CTR = CMPD: Time-base counter equal to counter-compare D register (TBCTR = CMPD).

The counter-compare registers CMPC and CMPD each have an associated shadow register. By default this register is shadowed. The memory address of the active register and the shadow register is identical. The value in the active CMPC and CMPD register is compared to the time-base counter (TBCTR). When the values are equal, the counter compare module generates a “time-base counter equal to counter compare C or counter compare D ” event respectively. Shadowing of this register is enabled and disabled by the CMPCTL2[SHDWCMODE] and CMPCTL2[SHDWDMODE] bit. These bits enable and disable the CMPC shadow register and CMPD shadow register respectively. The behavior of the two load modes is described below:

Shadow Mode:

The shadow mode for the CMPC is enabled by clearing the CMPCTL2[SHDWCMODE] bit and the shadow register for CMPD is enabled by clearing the CMPCTL2[SHDWDMODE] bit. Shadow mode is enabled by default for both CMPC and CMPD.

If the shadow register is enabled then the content of the shadow register is transferred to the active register on one of the following events as specified by the CMPCTL2[LOADCMODE], CMPCTL2[LOADDMODE], CMPCTL2[LOADCSYNC], and CMPCTL2[LOADDSYNC] register bits:

- CTR = PRD: Time-base counter equal to the period (TBCTR = TBPRD).
- CTR = Zero: Time-base counter equal to zero (TBCTR = 0x00)
- Both CTR = PRD and CTR = Zero
- SYNC event caused by DCAEVT1 or DCBEVT1 or EPWMxSYNCI or TBCTL[SWFSYNC]
- Both SYNC event or a selection made by LOADCMODE/LOADDMODE

Only the active register contents are used by the counter-compare submodule to generate events to be sent to the action-qualifier.

Immediate Load Mode:

If the immediate load mode is selected (that is, CMPCTL2[SHDWCMODE] = 1 or CMPCTL2[SHDWDMODE] = 1), then a read from or a write to the register goes directly to the active register.

Global Load Support

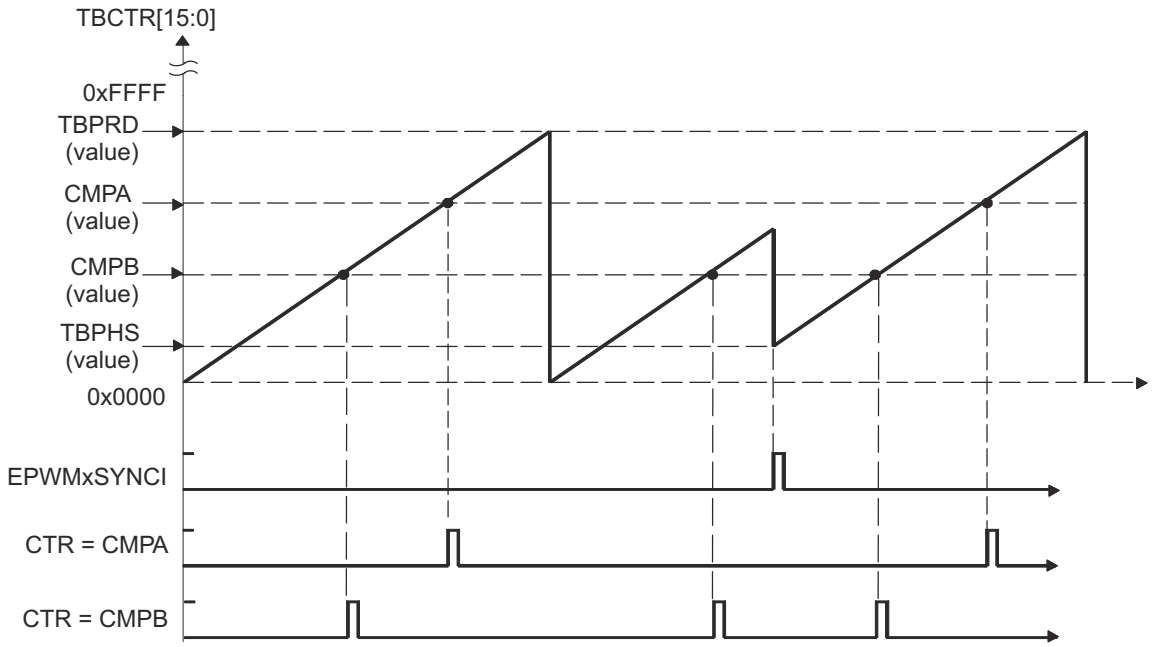
The global load control mechanism can also be used for all counter-compare registers by configuring the appropriate bits in the global load configuration register (GLDCFG). When the global load mode is selected the transfer of contents from shadow register to active register, for all registers that have this mode enabled, occurs at the same event as defined by the configuration bits in the Global Shadow to Active Load Control Register (GLDCTL). The global load control mechanism is explained in Section 26.4.7.

26.5.4 Count Mode Timing Waveforms

The counter-compare module can generate compare events in all three count modes:

- Up-count mode: used to generate an asymmetrical PWM waveform.
- Down-count mode: used to generate an asymmetrical PWM waveform.
- Up-down-count mode: used to generate a symmetrical PWM waveform.

To best illustrate the operation of the first three modes, the timing diagrams in Figure 26-17 through Figure 26-20 show when events are generated and how the EPWMxSYNCI signal interacts.



An EPWMxSYNCI external synchronization event can cause a discontinuity in the TBCTR count sequence. This can lead to a compare event being skipped. This skipping is considered normal operation and must be taken into account.

Figure 26-17. Counter-Compare Event Waveforms in Up-Count Mode

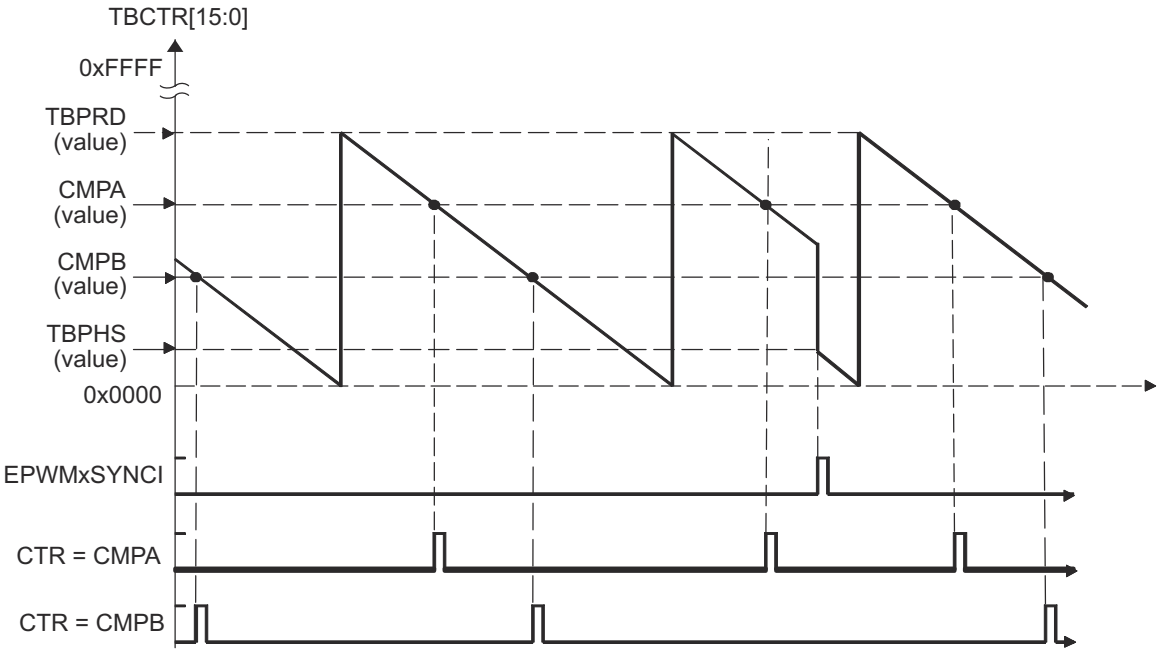


Figure 26-18. Counter-Compare Events in Down-Count Mode

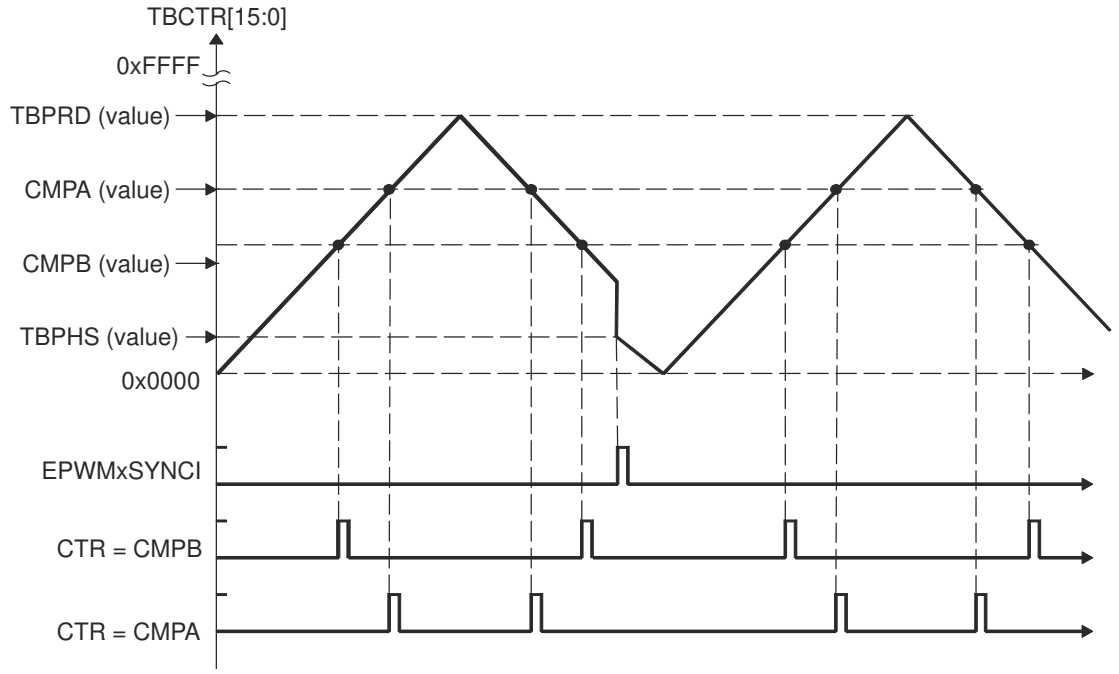


Figure 26-19. Counter-Compare Events In Up-Down-Count Mode, TBCTL[PHSDIR = 0] Count Down On Synchronization Event

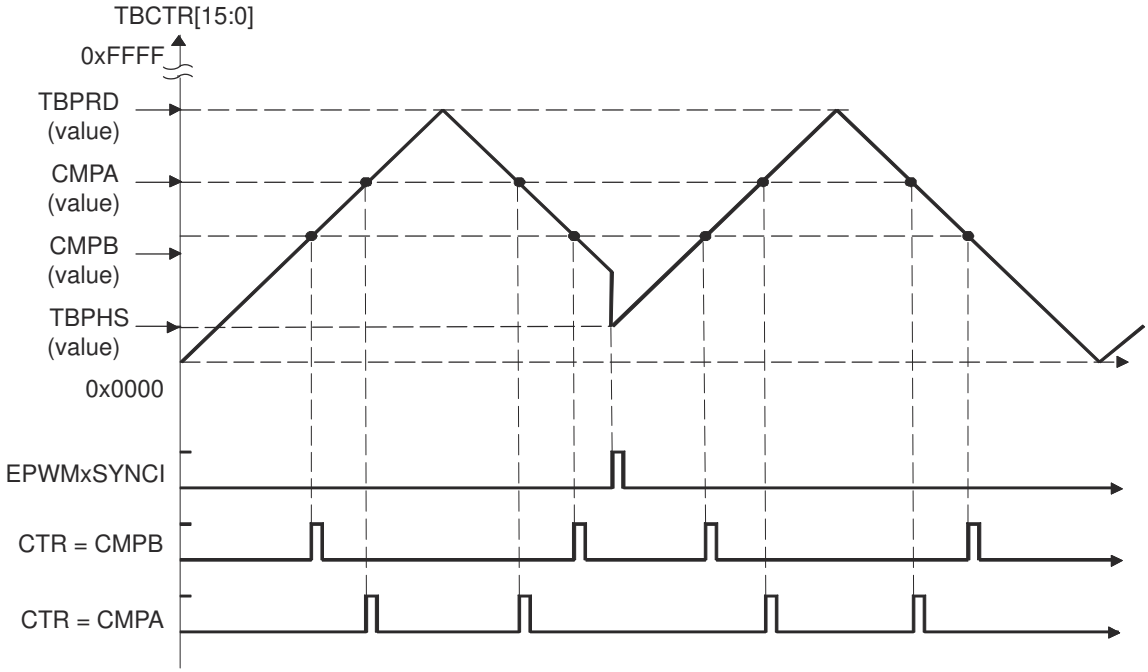


Figure 26-20. Counter-Compare Events In Up-Down-Count Mode, TBCTL[PHSDIR = 1] Count Up On Synchronization Event

26.6 Action-Qualifier (AQ) Submodule

The action-qualifier submodule has the most important role in waveform construction and PWM generation. The action-qualifier submodule decides which events are converted into various action types, thereby, producing the required switched waveforms at the EPWMxA and EPWMxB outputs.

Figure 26-21 illustrates the action-qualifier submodule within the ePWM.

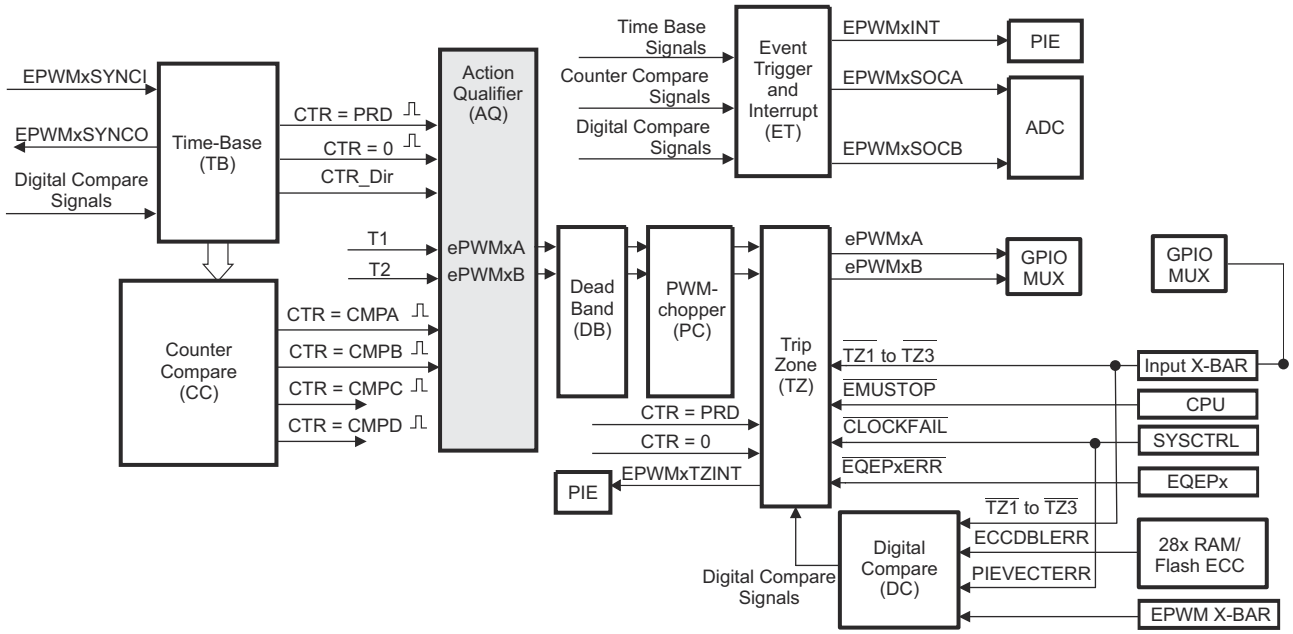


Figure 26-21. Action-Qualifier Submodule

26.6.1 Purpose of the Action-Qualifier Submodule

The action-qualifier submodule is responsible for the following:

- Qualifying and generating actions (set, clear, toggle) based on the following events:
  - CTR = PRD: Time-base counter equal to the period (TBCTR = TBPRD).
  - CTR = Zero: Time-base counter equal to zero (TBCTR = 0x00)
  - CTR = CMPA: Time-base counter equal to the counter-compare A register (TBCTR = CMPA)
  - CTR = CMPB: Time-base counter equal to the counter-compare B register (TBCTR = CMPB)
- T1, T2 events: Trigger events based on comparator, trip or syncin events
- Managing priority when these events occur concurrently
- Providing independent control of events when the time-base counter is increasing and when it is decreasing

26.6.2 Action-Qualifier Submodule Control and Status Register Definitions

The action-qualifier submodule operation is shown in Figure 26-22 and monitored by way of the registers in Section 26.17.

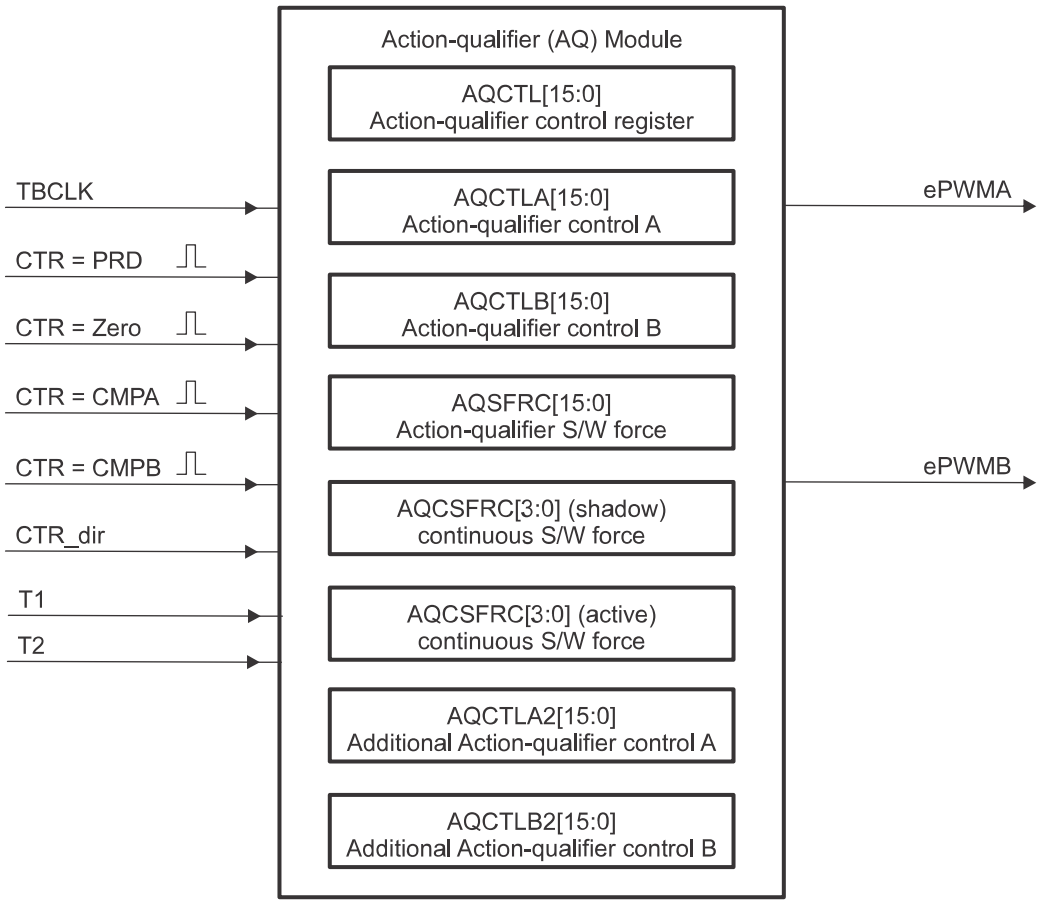


Figure 26-22. Action-Qualifier Submodule Inputs and Outputs

For convenience, the possible input events are summarized again in Table 26-4.

Table 26-4. Action-Qualifier Submodule Possible Input Events

Signal	Description	Registers Compared
CTR = PRD	Time-base counter equal to the period value	TBCTR = TBPRD
CTR = Zero	Time-base counter equal to 0	TBCTR = 0x00
CTR = CMPA	Time-base counter equal to the counter-compare A	TBCTR = CMPA
CTR = CMPB	Time-base counter equal to the counter-compare B	TBCTR = CMPB
T1 event	Based on comparator, trip, or syncin events	None
T2 event	Based on comparator, trip, or syncin events	None
Software forced event	Asynchronous event initiated by software	

The software forced action is a useful asynchronous event. This control is handled by the AQSFRC and AQCSFRC registers.

Note

If the CSFA is not used in shadow mode, the RLDCSF bit must be configured to disable shadow mode.

The action-qualifier submodule controls how the two outputs EPWMxA and EPWMxB behave when a particular event occurs. The event inputs to the action-qualifier submodule are further qualified by the counter direction (up or down). This allows for independent action on outputs on both the count-up and count-down phases.

The possible actions imposed on outputs EPWMxA and EPWMxB are:

- **Set High:** Set output EPWMxA or EPWMxB to a high level.
- **Clear Low:** Set output EPWMxA or EPWMxB to a low level.
- **Toggle:** If EPWMxA or EPWMxB is currently pulled high, then pull the output low. If EPWMxA or EPWMxB is currently pulled low, then pull the output high.
- **Do Nothing:** Keep outputs EPWMxA and EPWMxB at same level as currently set. Although the "Do Nothing" option prevents an event from causing an action on the EPWMxA and EPWMxB outputs, this event can still trigger interrupts and ADC start of conversion. See the description in Section 26.10 for details.

Actions are specified independently for either output (EPWMxA or EPWMxB). Any or all events can be configured to generate actions on a given output. For example, both CTR = CMPA and CTR = CMPB can operate on output EPWMxA. All qualifier actions are configured using the control registers found in the *ePWM Registers* section.

For clarity, the illustrations in this chapter use a set of symbolic actions. These symbols are summarized in Figure 26-23. Each symbol represents an action as a marker in time. Some actions are fixed in time (zero and period) while the CMPA and CMPB actions are moveable and their time positions are programmed by way of the counter-compare A and B registers, respectively. To turn off or disable an action, use the "Do Nothing option"(the default at reset).

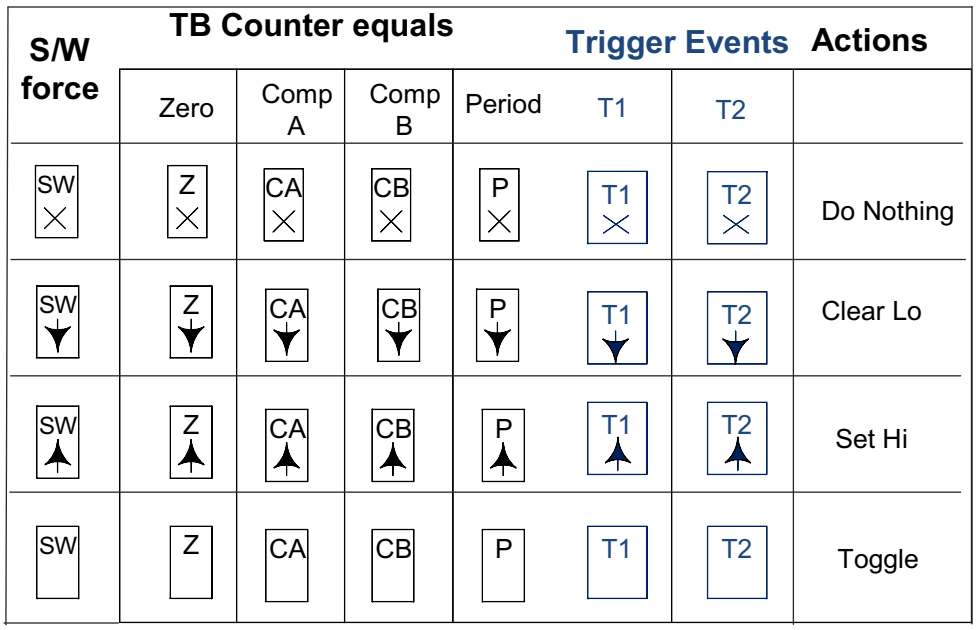


Figure 26-23. Possible Action-Qualifier Actions for EPWMxA and EPWMxB Outputs

The Action Qualifier Trigger Event Source Selection register (AQTSRCSEL) is used to select the source for T1 and T2 events. T1/T2 selection and configuration of a trip/digital-compare event in Action Qualifier submodule is independent of the configuration of that event in the Trip-Zone submodule. A particular trip event can or cannot be configured to cause trip action in the Trip Zone submodule, but the same event can be used by the Action Qualifier to generate T1/T2 for controlling PWM generation.

26.6.3 Action-Qualifier Event Priority

It is possible for the ePWM action qualifier to receive more than one event at the same time. In this case, events are assigned a priority by the hardware. The general rule is events occurring later in time have a higher priority and software forced events always have the highest priority. The event priority levels for up-down count mode are shown in Table 26-5. A priority level of 1 is the highest priority and level 10 is the lowest. The priority changes slightly depending on the direction of TBCTR.

Table 26-5. Action-Qualifier Event Priority for Up-Down-Count Mode

Priority Level	Event If TBCTR is Incrementing TBCTR = Zero up to TBCTR = TBPRD	Event If TBCTR is Decrementing TBCTR = TBPRD down to TBCTR = 1
1 (Highest)	Software forced event	Software forced event
2	T1 on up-count (T1U)	T1 on down-count (T1D)
3	T2 on up-count (T2U)	T2 on down-count (T2D)
4	Counter equals CMPB on up-count (CBU)	Counter equals CMPB on down-count (CBD)
5	Counter equals CMPA on up-count (CAU)	Counter equals CMPA on down-count (CAD)
6	Counter equals zero	Counter equals period (TBPRD)
7	T1 on down-count (T1D)	T1 on up-count (T1U)
8(Lowest)	T2 on down-count (T2D)	T2 on up-count (T2U)

Table 26-6 shows the action-qualifier priority for up-count mode. In this case, the counter direction is always defined as up; therefore, down-count events never are taken.

Table 26-6. Action-Qualifier Event Priority for Up-Count Mode

Priority Level	Event
1 (Highest)	Software forced event
2	Counter equal to period (TBPRD)
3	T1 on up-count (T1U)
4	T2 on up-count (T2U)
5	Counter equal to CMPB on up-count (CBU)
6	Counter equal to CMPA on up-count (CAU)
7 (Lowest)	Counter equal to Zero

Table 26-7 shows the action-qualifier priority for down-count mode. In this case, the counter direction is always defined as down; therefore, up-count events never are taken.

Table 26-7. Action-Qualifier Event Priority for Down-Count Mode

Priority Level	Event
1 (Highest)	Software forced event
2	Counter equal to Zero
3	T1 on down-count (T1D)
4	T2 on down-count (T2D)
5	Counter equal to CMPB on down-count (CBD)
6	Counter equal to CMPA on down-count (CAD)
7 (Lowest)	Counter equal to period (TBPRD)

It is possible to set the compare value greater than the period. In this case, the action takes place as shown in Table 26-8.

Table 26-8. Behavior if CMPA/CMPB is Greater than the Period

Counter Mode	Compare on Up-Count Event CAD/CBD	Compare on Down-Count Event CAD/CBD
Up-Count Mode	If CMPA/CMPB ≤ TBPRD period, then the event occurs on a compare match (TBCTR=CMPA or CMPB).	Never occurs.
	If CMPA/CMPB > TBPRD, then the event does not occur.	
Down-Count Mode	Never occurs.	If CMPA/CMPB < TBPRD, the event occurs on a compare match (TBCTR=CMPA or CMPB).
		If CMPA/CMPB ≥ TBPRD, the event occurs on a period match (TBCTR=TBPRD).
Up-Down Count Mode	If CMPA/CMPB < TBPRD and the counter is incrementing, the event occurs on a compare match (TBCTR=CMPA or CMPB).	If CMPA/CMPB < TBPRD and the counter is decrementing, the event occurs on a compare match (TBCTR=CMPA or CMPB).
	If CMPA/CMPB is ≥ TBPRD, the event occurs on a period match (TBCTR = TBPRD).	If CMPA/CMPB ≥ TBPRD, the event occurs on a period match (TBCTR=TBPRD).

26.6.4 AQCTLA and AQCTLB Shadow Mode Operations

To enable Action Qualifier mode changes which must occur at the end of a period even when the phase changes, shadowing of the AQCTLA and AQCTLB registers has been added on ePWMs type 2 and later. Additionally, shadow to active load on SYNC of these registers is supported as well. Shadowing of this register is enabled and disabled by the AQCTL[SHDWAQAMODE] and AQCTL[SHDWAQBMODE] bits. These bits enable and disable the AQCTLA shadow register and AQCTLB shadow register, respectively. The behavior of the two load modes is:

Shadow Mode:

The shadow mode for the AQCTLA is enabled by setting the AQCTL[SHDWAQAMODE] bit, and the shadow register for AQCTLB is enabled by setting the AQCTL[SHDWAQBMODE] bit. Shadow mode is disabled by default for both AQCTLA and AQCTLB

If the shadow register is enabled, then the content of the shadow register is transferred to the active register on one of the following events as specified by the AQCTL[LDAQAMODE], AQCTL[LDAQBMODE], AQCTL[LDAQASYNC], and AQCTL[LDAQBSYNC] register bits:

- CTR = PRD: Time-base counter equal to the period (TBCTR = TBPRD).
- CTR = Zero: Time-base counter equal to zero (TBCTR = 0x00)
- Both CTR = PRD and CTR = Zero
- SYNC event caused by DCAEVT1 or DCBEVT1 or EPWMxSYNCI or TBCTL[SWFSYNC]
- Both SYNC event or a selection made by LDAQAMODE/LDAQBMODE

Global Load Support

Global load control mechanism can also be used for AQCTLA:AQCTLA2, AQCTLB:AQCTLB2, and AQCSFRC registers by configuring the appropriate bits in the global load configuration register (GLDCFG). When global load mode is selected, the transfer of contents from shadow register to active register for all registers that have this mode enabled, occurs at the same event as defined by the configuration bits in the Global Shadow to Active Load Control Register (GLDCTL). The global load control mechanism is explained in Section 26.4.7.

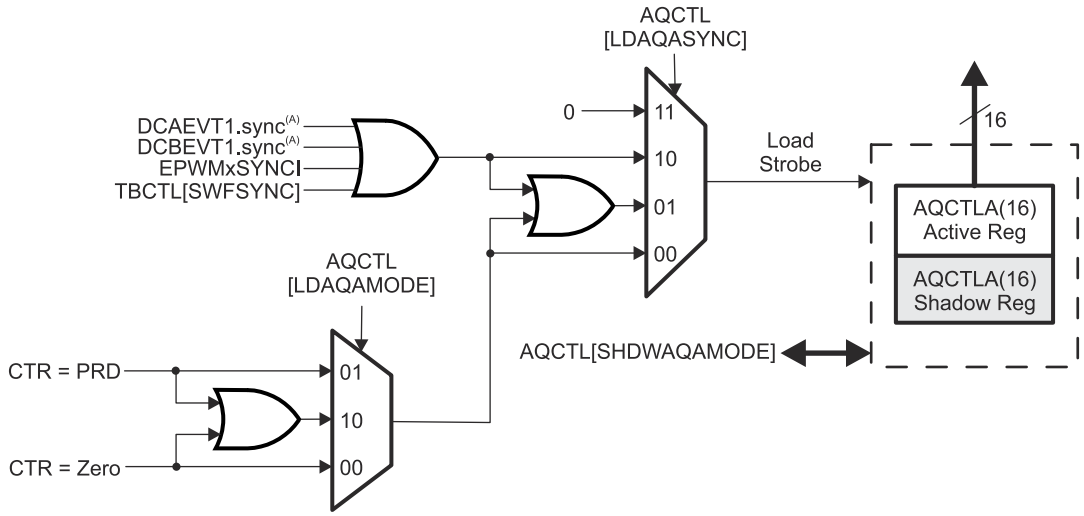
Immediate Load Mode:

If the immediate load mode is selected (that is, AQCTL[SHDWAQAMODE] = 0 or AQCTL[SHDWAQBMODE] = 0), then a read from or a write to the register goes directly to the active register. See Figure 26-24 and Figure 26-25.

Note

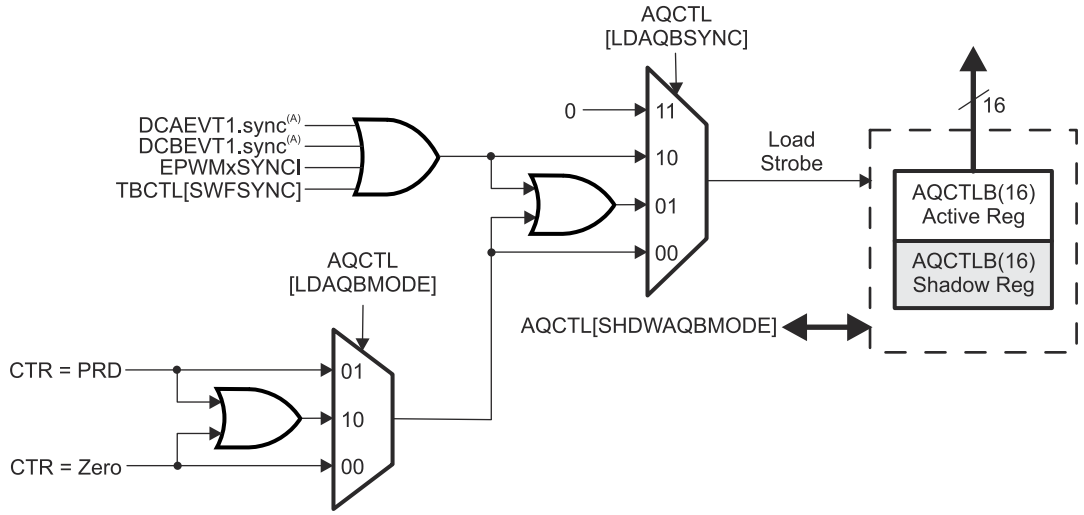
Shadow to Active Load of Action Qualifier Output A/B Control Register [AQCTLA and AQCTLB] on CMPA = 0 or CMPB = 0 boundary

If the Counter-Compare A Register (CMPA) or Counter-Compare B Register (CMPB) is set to a value of 0 and the action qualifier action on AQCTLA and AQCTLB is configured to occur in the same instant as a shadow to active load (that is, CMPA=0 and AQCTLA shadow to active load on TBCTR=0 using AQCTL register LDAQAMODE and LDAQAMODE bits), then both events enter contention. It is recommended to use a Non-Zero Counter-Compare when using Shadow to Active Load of Action Qualifier Output A/B Control Register on TBCTR = 0 boundary.



A. These events are generated by the ePWM digital compare (DC) submodule based on the levels of the TRIPIN inputs (for example, CMPSSx and $\overline{TZ}$ signals).

Figure 26-24. AQCTL[SHDWAQAMODE]



A. These events are generated by the ePWM digital compare (DC) submodule based on the levels of the TRIPIN inputs (for example, CMPSSx and $\overline{TZ}$ signals).

Figure 26-25. AQCTL[SHDWAQBMODE]

26.6.5 Configuration Requirements for Common Waveforms

Note

The waveforms in this document show the behavior of the ePWMs for a static compare register value. In a running system, the active compare registers (CMPA and CMPB) are typically updated from their respective shadow registers once every period. Specify when the update takes place: either when the time-base counter reaches zero or when the time-base counter reaches the period. There are some cases when the action based on the new value can be delayed by one period or the action based on the old value can take effect for an extra period. Some PWM configurations avoid this situation. These include, but are not limited to, the following:

Use up-down count mode to generate a symmetric PWM:

- If loading CMPA/CMPB on zero, then use CMPA/CMPB values greater than or equal to 1.
- If loading CMPA/CMPB on period, then use CMPA/CMPB values less than or equal to TBPRD-1.

  This means there is always a pulse of at least one TBCLK cycle in a PWM period which, when very short, tend to be ignored by the system.

Use up-down count mode to generate an asymmetric PWM:

- To achieve 50%-0% asymmetric PWM use the following configuration: Load CMPA/CMPB on period and use the period action to clear the PWM and a compare-up action to set the PWM. Modulate the compare value from 0 to TBPRD to achieve 50%-0% PWM duty.

When using up-count mode to generate an asymmetric PWM:

- To achieve 0-100% asymmetric PWM, you **must** load CMPA/CMPB on TBPRD. When CMPA/CMPB is not loaded on TBCTR=PRD, boundary conditions can occur depending on the timing of the write and the value written to CMPA/CMPB. Use the Zero action to set the PWM and a compare-up action to clear the PWM. Modulate the compare value from 0 to TBPRD+1 to achieve 0-100% PWM duty.

When using up-count mode to generate an asymmetric PWM with deadband enabled:

- To achieve 0%-100% PWM use the following configuration: When the CMPA value is too close to 0 or PRD such that the following conditions are met (CMPX < Deadband) or (CMPX > PRD – Deadband), the actions specified by the AQCTL register for CMPX do not take effect. To avoid this, the AQCTL settings have to be altered under these conditions only to generate either high or low pulses for both CAU or CAD events (both set or both clear). Make sure that this software update is occurring synchronous to the PWM carrier cycle, and shadow mode is enabled.

When using up-down count mode to generate an asymmetric PWM with deadband enabled:

- To achieve 0%-100% PWM use the following configuration: When the CMPA value is too close to 0 or PRD such that the following conditions are met (CMPX < Deadband/2) or (CMPX > PRD – (Deadband)/2), the actions specified by the AQCTL register for CMPX do not take effect. To avoid this, the AQCTL settings have to be altered under these conditions only to generate either high or low pulses for both CAU or CAD events (both set or both clear). Make sure that this software update is occurring synchronous to the PWM carrier cycle, and shadow mode is enabled.

See *Using Enhanced Pulse Width Modulator (ePWM) Module for 0-100% Duty Cycle Control*.

Figure 26-26 shows how a symmetric PWM waveform can be generated using the up-down-count mode of the TBCTR. In this mode, 0%-100% DC modulation is achieved by using equal compare matches on the up count and down count portions of the waveform. In the example shown, CMPA is used to make the comparison. When the counter is incrementing, the CMPA match pulls the PWM output high. Likewise when the counter is decrementing, the compare match pulls the PWM signal low. When CMPA = 0, the PWM signal is high for the entire period giving a 100% duty waveform. When CMPA = TBPRD, the PWM signal is low achieving 0% duty.

When using this configuration in practice, if loading CMPA/CMPB on zero, then use CMPA/CMPB values greater than or equal to 1. If loading CMPA/CMPB on period, then use CMPA/CMPB values less than or equal to TBPRD-1. This means there is always a pulse of at least one TBCLK cycle in a PWM period which, when very short, tend to be ignored by the system.

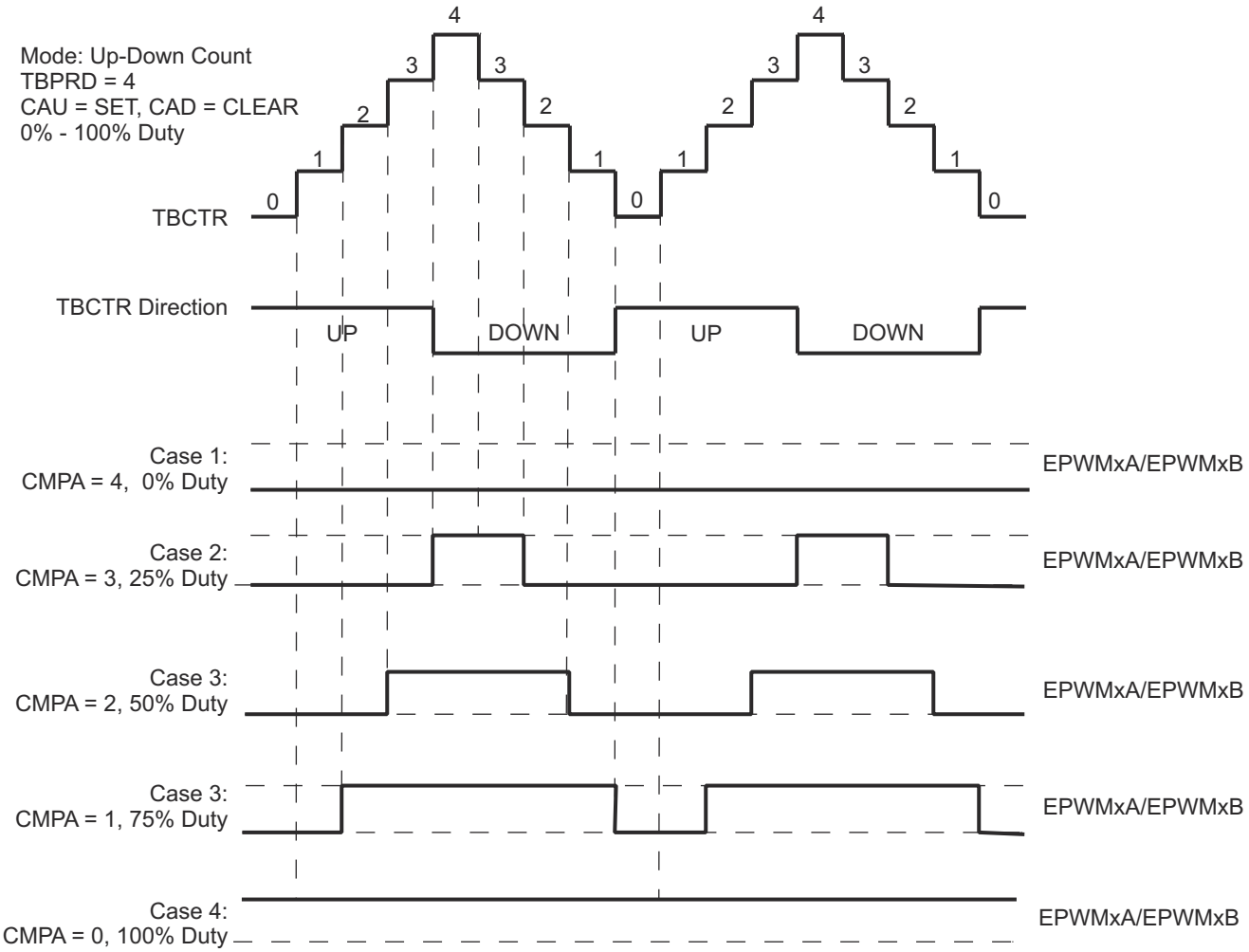
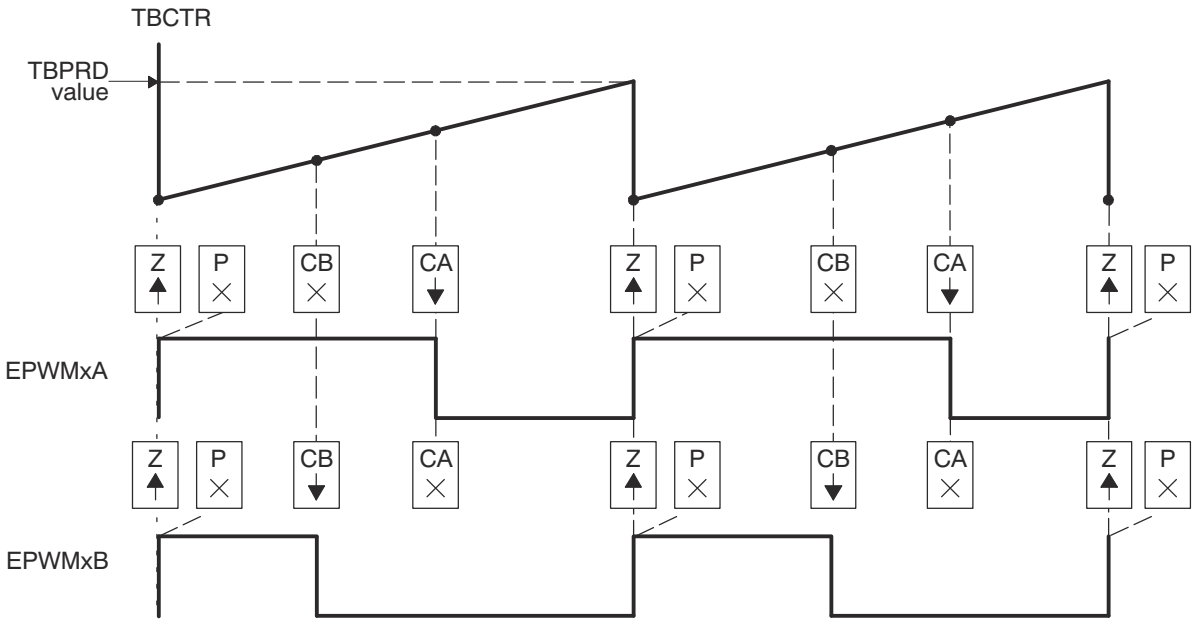


Figure 26-26. Up-Down Count Mode Symmetrical Waveform

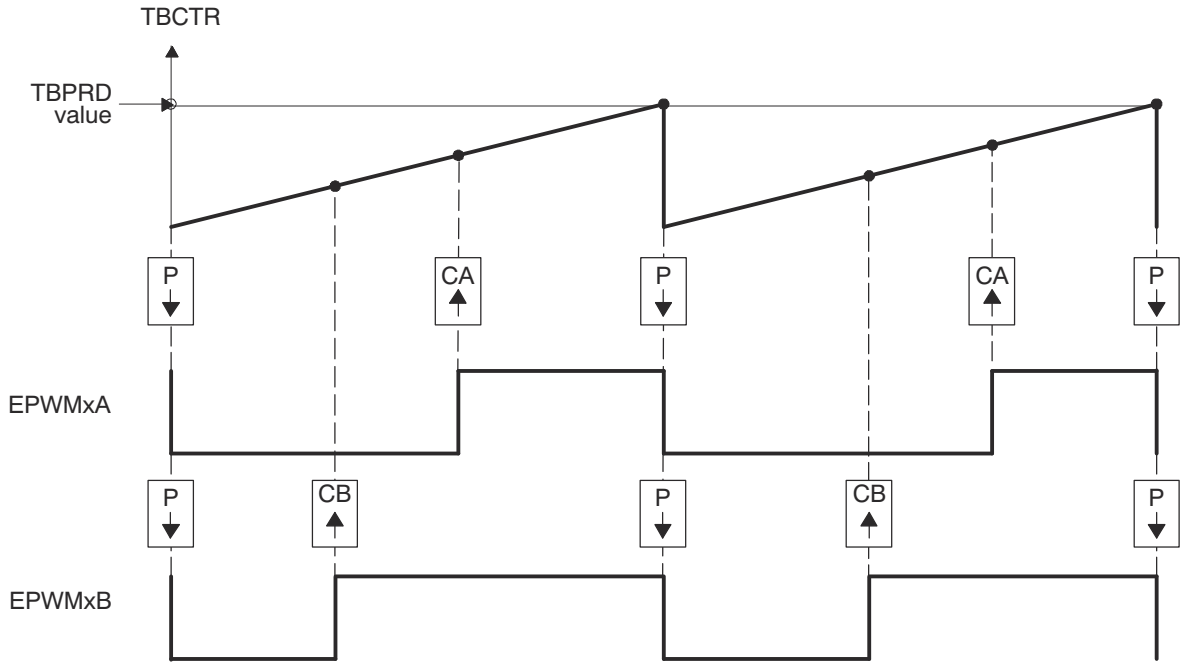
The PWM waveforms in Figure 26-27 through Figure 26-32 show some common action-qualifier configurations. Some conventions used in the figures and examples are as follows:

- TBPRD, CMPA, and CMPB refer to the value written in their respective registers. The active register, not the shadow register, is used by the hardware.
- CMPx, refers to either CMPA or CMPB.
- EPWMxA and EPWMxB refer to the output signals from ePWMx
- Up-Down means count-up-and count-down mode, Up means up-count mode and Down means down-count mode
- Sym = Symmetric, Asym = Asymmetric



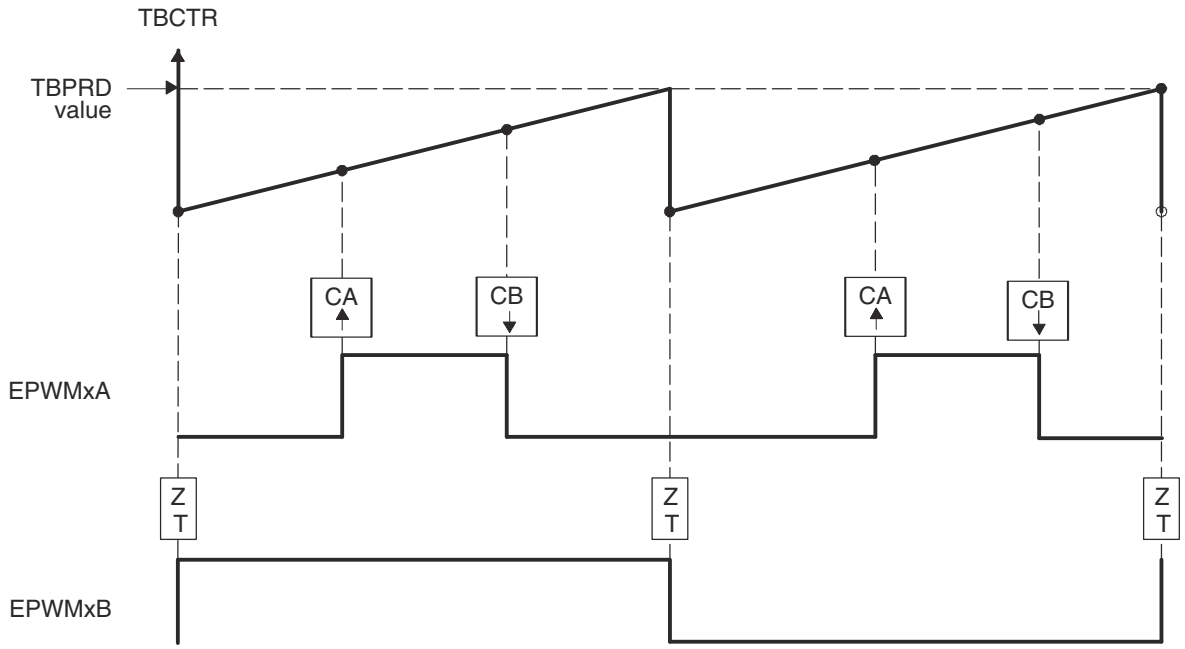
A. PWM period = (TBPRD + 1) × T_{TBCLK}
B. Duty modulation for EPWMxA is set by CMPA, and is active high (that is, high time duty proportional to CMPA).
C. Duty modulation for EPWMxB is set by CMPB and is active high (that is, high time duty proportional to CMPB).
D. The "Do Nothing" actions (X) are shown for completeness, but are not shown on subsequent diagrams.
E. Actions at zero and period, although appearing to occur concurrently, are actually separated by one TBCLK period. TBCTR wraps from period to 0000.

Figure 26-27. Up, Single Edge Asymmetric Waveform, with Independent Modulation on EPWMxA and EPWMxB—Active High



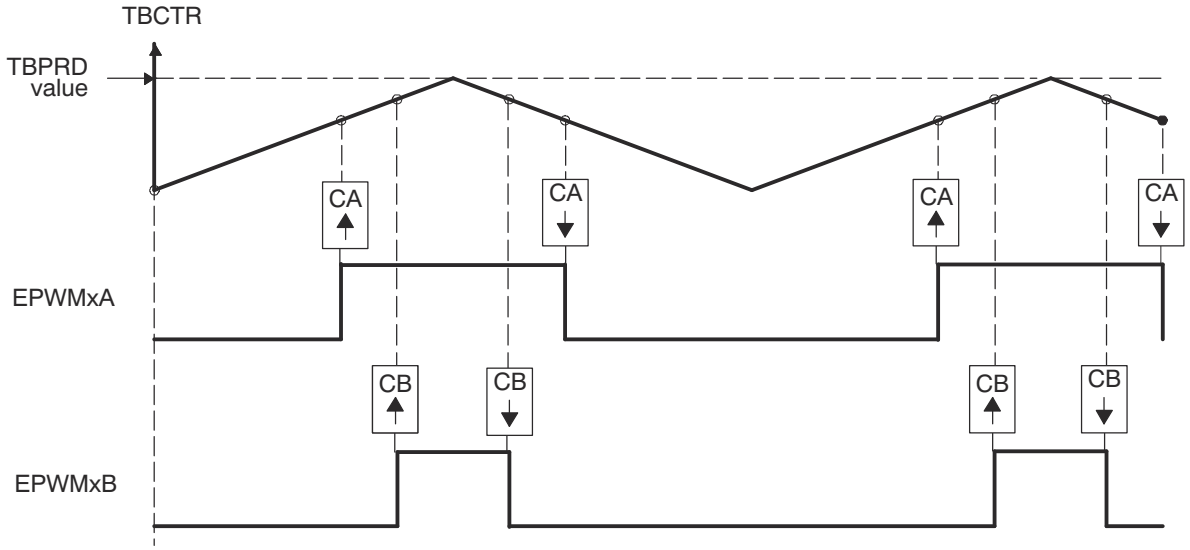
A. PWM period = (TBPRD + 1) × T_{TBCLK}
B. Duty modulation for EPWMxA is set by CMPA, and is active low (that is, the low time duty is proportional to CMPA).
C. Duty modulation for EPWMxB is set by CMPB and is active low (that is, the low time duty is proportional to CMPB).
D. Actions at zero and period, although appearing to occur concurrently, are actually separated by one TBCLK period. TBCTR wraps from period to 0000.

Figure 26-28. Up, Single Edge Asymmetric Waveform with Independent Modulation on EPWMxA and EPWMxB—Active Low



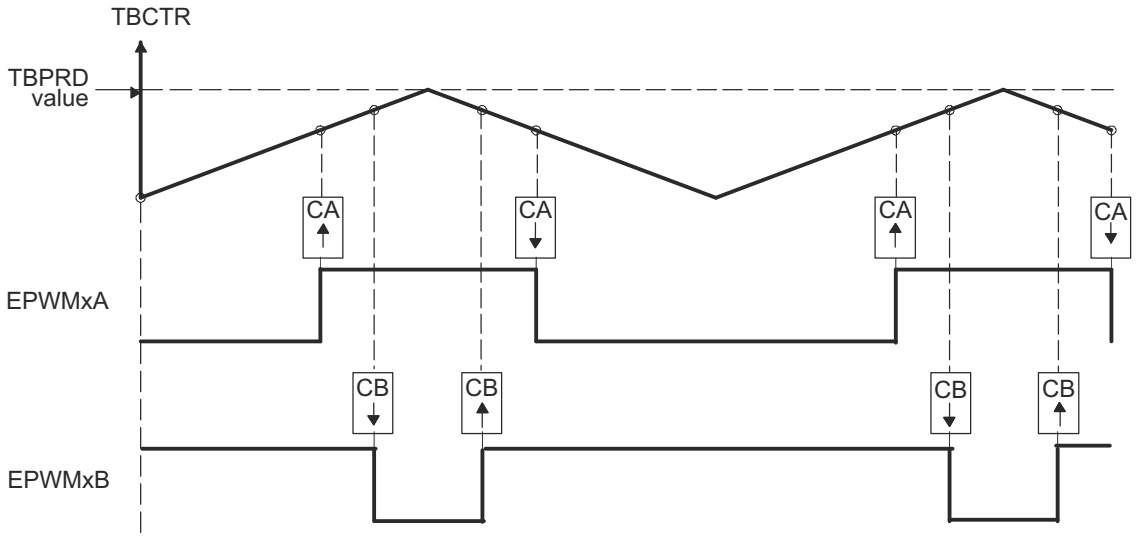
A. PWM frequency = 1/((TBPRD + 1) × T_{TBCLK})
B. Pulse can be placed anywhere within the PWM cycle (0000 - TBPRD)
C. High time duty proportional to (CMPB - CMPA)

Figure 26-29. Up-Count, Pulse Placement Asymmetric Waveform With Independent Modulation on EPWMxA



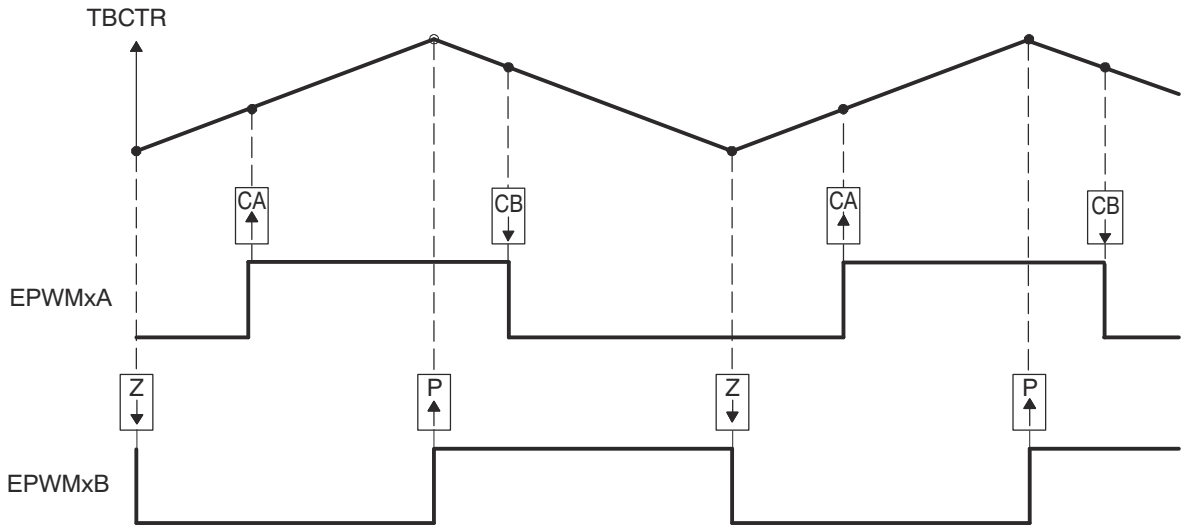
A. PWM period = 2 x TBPRD × T_{TBCLK}
B. Duty modulation for EPWMxA is set by CMPA, and is active low (that is, the low time duty is proportional to CMPA).
C. Duty modulation for EPWMxB is set by CMPB and is active low (that is, the low time duty is proportional to CMPB).
D. Outputs EPWMxA and EPWMxB can drive independent power switches.

Figure 26-30. Up-Down Count, Dual-Edge Symmetric Waveform, with Independent Modulation on EPWMxA and EPWMxB — Active Low



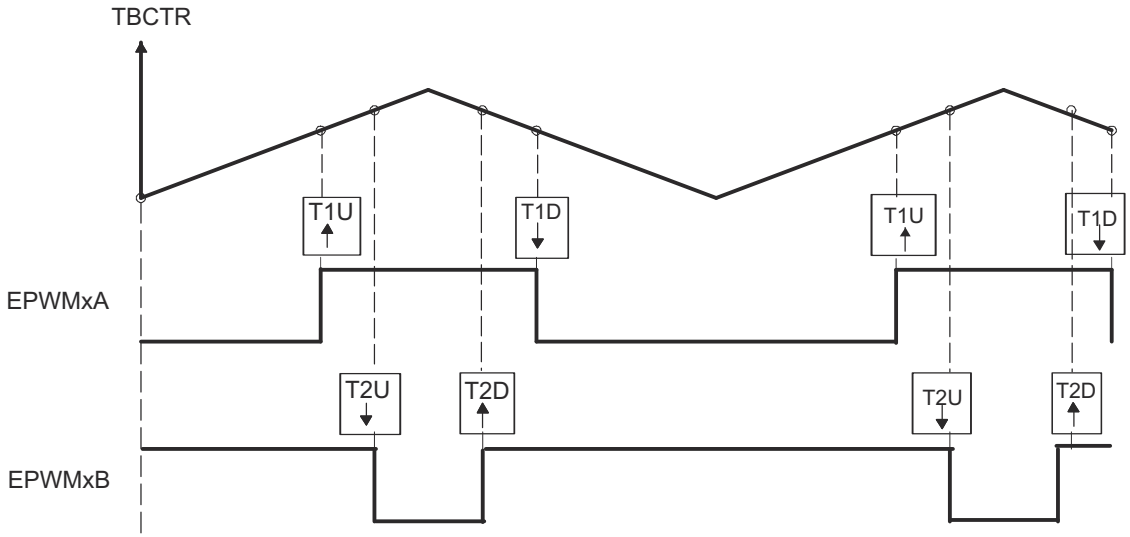
A. PWM period = 2 × TBPRD × T_{TBCLK}
B. Duty modulation for EPWMxA is set by CMPA, and is active low, that is, low time duty proportional to CMPA.
C. Duty modulation for EPWMxB is set by CMPB and is active high, that is, high time duty proportional to CMPB.
D. Outputs EPWMx can drive upper/lower (complementary) power switches.
E. Dead-band = CMPB - CMPA (fully programmable edge placement by software). Note the dead-band module is also available if the more classical edge delay method is required.

Figure 26-31. Up-Down Count, Dual-Edge Symmetric Waveform, with Independent Modulation on EPWMxA and EPWMxB — Complementary



A. PWM period = 2 × TBPRD × TBCLK
B. Rising edge and falling edge can be asymmetrically positioned within a PWM cycle. This allows for pulse placement techniques.
C. Duty modulation for EPWMxA is set by CMPA and CMPB.
D. Low time duty for EPWMxA is proportional to (CMPA + CMPB).
E. To change this example to active high, CMPA and CMPB actions need to be inverted (that is, Clear on CMPA, Set on CMPB).
F. Duty modulation for EPWMxB is fixed at 50% (utilizes spare action resources for EPWMxB).

Figure 26-32. Up-Down Count, Dual-Edge Asymmetric Waveform, with Independent Modulation on EPWMxA—Active Low



A. PWM period = 2 × TBPRD × TTBCLK
B. Independent T1 event actions when counter is counting up and when the counter is counting down are used to generate EPWMxA output.
C. Independent T2 event actions when counter is counting up and when the counter is counting down are used to generate EPWMxB output.
D. TZ1 is selected as the source for T1.
E. TZ2 is selected as the source for T2.

Figure 26-33. Up-Down Count, PWM Waveform Generation Utilizing T1 and T2 Events

26.7 Dead-Band Generator (DB) Submodule

Figure 26-34 illustrates the dead-band submodule within the ePWM.

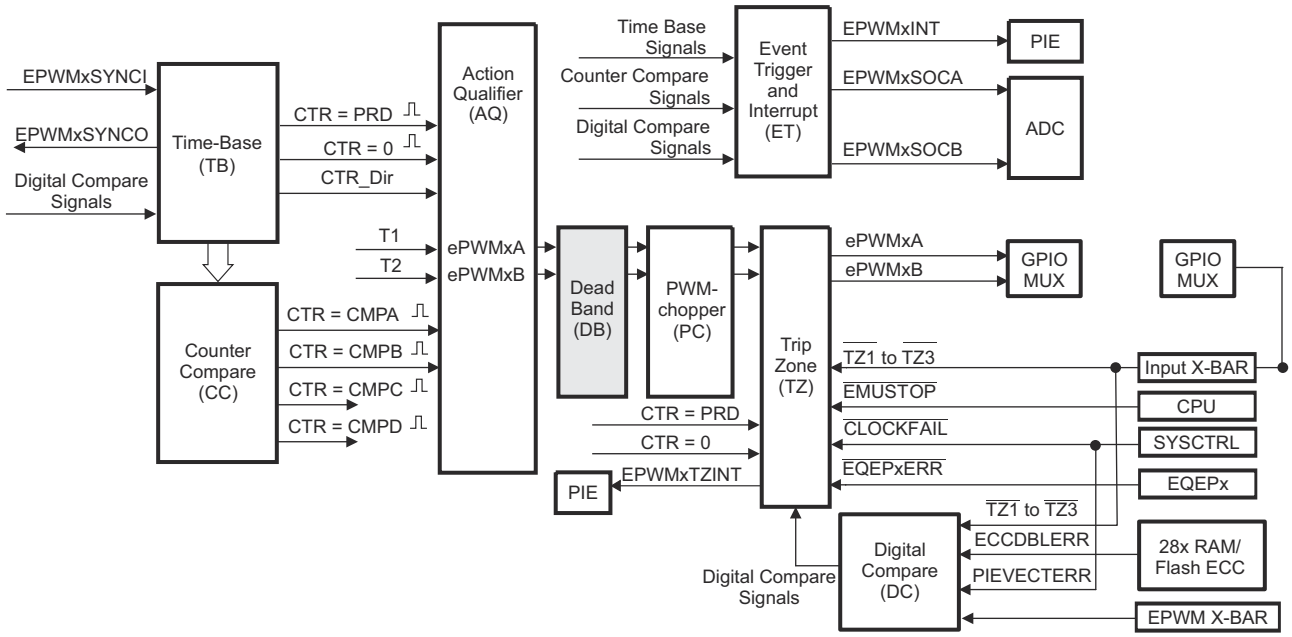


Figure 26-34. Dead_Band Submodule

26.7.1 Purpose of the Dead-Band Submodule

The action-qualifier (AQ) module section discussed how the AQ module is possible to generate the required dead band by having full control over edge placement using both the CMPA and CMPB resources of the ePWM module. However, if the more classical edge delay-based dead band with polarity control is required, then the dead-band submodule described here must be used.

The key functions of the dead-band module are:

- Generating appropriate signal pairs (EPWMxA and EPWMxB) with dead-band relationship from a single EPWMxA input
- Programming signal pairs for:
  - Active high (AH)
  - Active low (AL)
  - Active high complementary (AHC)
  - Active low complementary (ALC)
- Adding programmable delay to rising edges (RED)
- Adding programmable delay to falling edges (FED)
- Can be totally bypassed from the signal path (note dotted lines in diagram)

26.7.2 Dead-band Submodule Additional Operating Modes

On type 1 ePWM RED can appear on one channel output and FED can appear on the other channel output.

The following list shows the distinct difference between type 1 and type 4 modules with respect to dead-band operating modes:

- By adding S6, S7, and S8 in Figure 26-35, RED and FED can appear on both the A-channel and B-channel outputs. Additionally, both RED and FED together can be applied to either the A-channel or B-channel outputs to allow B-channel phase shifting with respect to the A-channel.

Note

Phase shifting B-channel with respect to the A-channel using the dead-band submodule additional operating modes has limitations with respect to the choice of RED and FED delay with respect to the operating duty cycle of the ePWMxA and ePWMxB outputs.

- The dead-band counters have also been increased to 14 bits
- Deadband and deadband high-resolution registers are now shadowed
- High-resolution deadband RED and FED have been enabled using the DBREDHR and DBFEDHR registers

Note

The PWM chopper is not enabled when high-resolution deadband is enabled.

High-resolution deadband RED and FED requires half-cycle clocking mode (DBCTL[HALFCYCLE] = 1).

Cannot have both RED and FED together applied to both ePWMxA and ePWMxB. RED and FED together can be applied only to either OutA OR OutB.

Phase shifting B-channel with respect to the A-channel: When PWMxB is derived from PWMxA using the DEDB_MODE bit and by delaying rising edge and falling edge by the phase shift amount. When the duty cycle value on PWMxA is less than this phase shift amount, PWMxA's falling edge has precedence over the delayed rising edge for PWMxB. It is recommended to make sure the duty cycle value of the current waveform fed to the dead-band module is greater than the required phase shift amount.

The Type 4 action qualifier and dead-band outputs of the ePWM module are delayed by one TBCLK cycle in comparison to the Type 2 ePWM module, although the Type 4 behavior is the same as the Type 3 PWM. Both PWMA and PWMB signals are delayed under all circumstances.

The shadow mode for the DBRED is enabled by setting the DBCTL[SHDWDBREDMODE] bit and the shadow register for DBFED is enabled by setting the DBCTL [SHDWDBFEDMODE] bit. Shadow mode is disabled by default for both DBRED and DBFED

If the shadow register is enabled, then the content of the shadow register is transferred to the active register on one of the following events as specified by the DBCTL [LOADREDMODE] and DBCTL [LOADFEDMODE] register bits:

- CTR = PRD: Time-base counter equal to the period (TBCTR = TBPRD).
- CTR = Zero: Time-base counter equal to zero (TBCTR = 0x00)
- Both CTR = PRD and CTR = Zero

The DBCTL register can be shadowed. The shadow mode for DBCTL is enabled by setting the DBCTL2[SHDWDBCTLMODE] bit. If the shadow register is enabled then the content of the shadow register is transferred to the active register on one of the following events as specified by the DBCTL2[LOADDBCTLMODE] register bit:

- CTR = PRD: Time-base counter equal to the period (TBCTR = TBPRD)
- CTR = Zero: Time-base counter equal to zero (TBCTR = 0x00)
- Both CTR = PRD and CTR = Zero

Note

The application software must enable shadow load mode in the DBCTL[SHDWDBREDMODE] and DBCTL[SHDWDBFEDMODE] **before** programming values for the DBRED and DBFED registers. If the shadow register is enabled **after** programming the DBRED and DBFED registers, the DBRED and DBFED registers will be loaded with value of 0.

Global Load Support

Global load control mechanism can also be used for DBRED:DBREDHR, DBFED:DBFEDHR, and DBCTL registers by configuring the appropriate bits in the global load configuration register (GLDCFG). When global load mode is selected the transfer of contents from shadow register to active register, for all registers that have this mode enabled, occurs at the same event as defined by the configuration bits in the Global Shadow to Active Load Control Register (GLDCTL). The Global load control mechanism is explained in Section 26.4.7.

Note

When DBRED/DBFED active is loaded with a new shadow value while DB counters are counting, the new DBRED/DBFED value only affects the NEXT PWMx edge and not the current edge.

A Deadband value of zero cannot be used when the Global Shadow to Active Load is set to occur at CTR=ZERO. Similarly, a Deadband value of PRD cannot be used when the Global Shadow to Active Load is set to occur at CTR=PRD.

TBPRDHR cannot be used with Global load. If high-resolution period must be changed in the application, users must write to the individual period registers from an ePWM ISR (The ISR must be synchronous with the PWM switching period), where the Global Load One-Shot bit is also written to.

26.7.3 Operational Highlights for the Dead-Band Submodule

The configuration options for the dead-band submodule are shown in Figure 26-35.

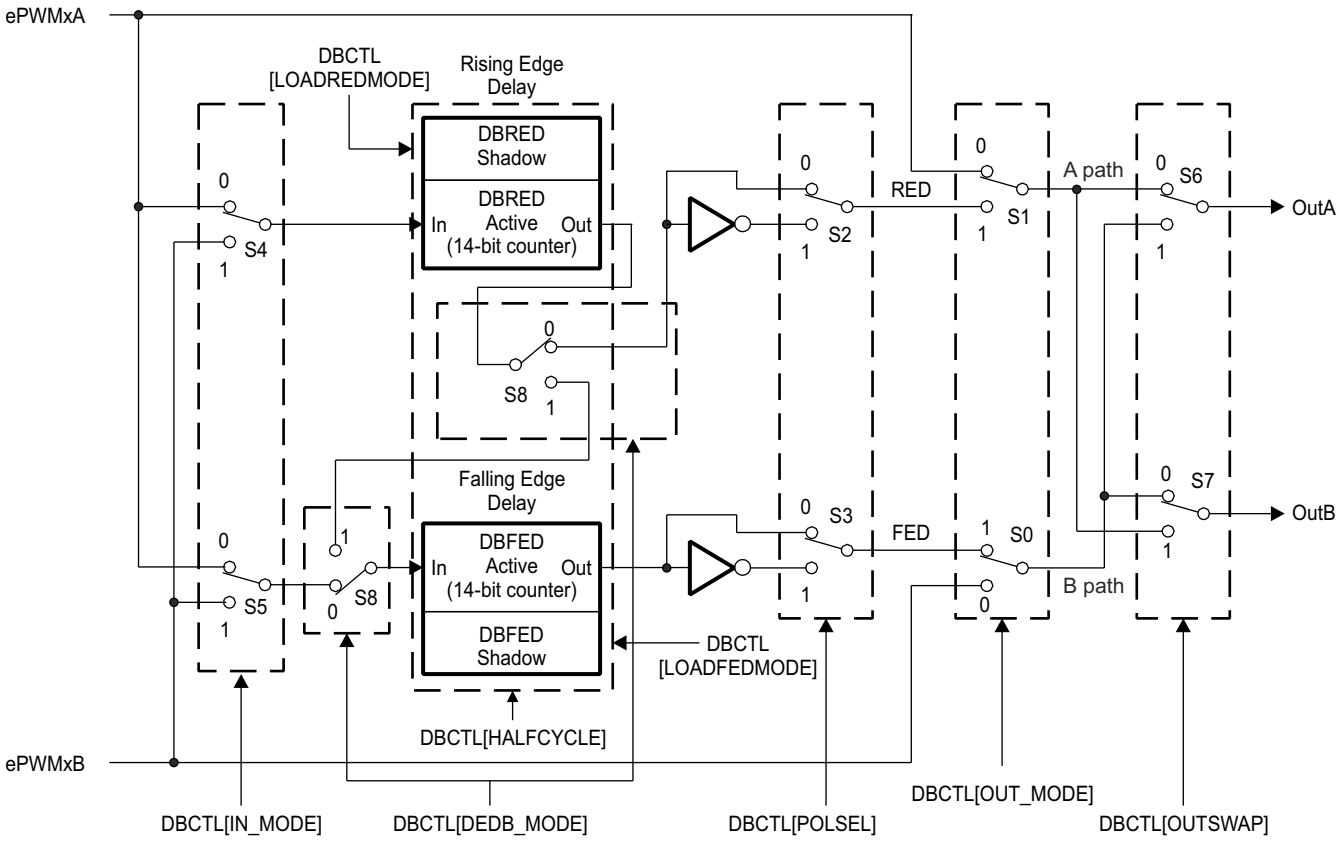


Figure 26-35. Configuration Options for the Dead-Band Submodule

Although all combinations are supported, not all are typical usage modes. Table 26-9 documents some classical dead-band configurations. These modes assume that the DBCTL[IN_MODE] is configured such that EPWMxA In is the source for both falling-edge and rising-edge delay. Enhanced, or non-traditional modes can be achieved by changing the input signal source. The modes shown in Table 26-9 fall into the following categories:

- **Mode 1: Bypass both falling-edge delay (FED) and rising-edge delay (RED):** Allows the user to fully disable the dead-band submodule from the PWM signal path.
- **Mode 2-5: Classical Dead-Band Polarity Settings:** These represent typical polarity configurations that can address all the active-high and active-low modes required by available industry power switch gate drivers. The waveforms for these typical cases are shown in Figure 26-36. Note that to generate equivalent waveforms to Figure 26-36, configure the action-qualifier submodule to generate the signal as shown for EPWMxA.
- **Mode 6: Bypass rising-edge-delay and Mode 7: Bypass falling-edge-delay:** Finally the last two entries in Table 26-9 show combinations where either the falling-edge-delay (FED) or rising-edge-delay (RED) blocks are bypassed.

Figure 26-36 shows waveforms for typical cases where 0% < duty < 100%.

Table 26-9. Classical Dead-Band Operating Modes

Mode	Mode Description	DBCTL[POLSEL]		DBCTL[OUT_MODE]	
		S3	S2	S1	S0
1	EPWMxA and EPWMxB Passed Through (No Delay)	X	X	0	0
2	Active High Complementary (AHC)	1	0	1	1
3	Active Low Complementary (ALC)	0	1	1	1
4	Active High (AH)	0	0	1	1
5	Active Low (AL)	1	1	1	1
6	EPWMxA Out = EPWMxA In (No Delay) EPWMxB Out = EPWMxA In with Falling Edge Delay	0 or 1	0 or 1	0	1
7	EPWMxA Out = EPWMxA In with Rising Edge Delay EPWMxB Out = EPWMxB In with No Delay	0 or 1	0 or 1	1	0

Table 26-10. Additional Dead-Band Operating Modes

Mode Description	DBCTL[DEDB-MODE]	DBCTL[OUTSWAP]	
	S8	S6	S7
EPWMxA and EPWMxB signals are as defined by OUT-MODE bits.	0	0	0
EPWMxA = A-path as defined by OUT-MODE bits. EPWMxB = A-path as defined by OUT-MODE bits (rising edge delay or delay-bypassed A-signal path)	0	0	1
EPWMxA = B-path as defined by OUT-MODE bits (falling edge delay or delay-bypassed B-signal path) EPWMxB = B-path as defined by OUT-MODE bits	0	1	0
EPWMxA = B-path as defined by OUT-MODE bits (falling edge delay or delay-bypassed B-signal path) EPWMxB = A-path as defined by OUT-MODE bits (rising edge delay or delay-bypassed A-signal path)	0	1	1
Rising edge delay applied to EPWMxA / EPWMxB as selected by S4 switch (IN-MODE bits) on A signal path only. Falling edge delay applied to EPWMxA / EPWMxB as selected by S5 switch (IN-MODE bits) on B signal path only.	0	X	X
Rising edge delay and falling edge delay applied to source selected by S4 switch (IN-MODE bits) and output to B signal path only.(1)	1	X	X

(1) When this bit is set to 1, the user can always either set OUT_MODE bits such that Apath = InA or set OUTSWAP bits such that EPWMxA=Bpath. Otherwise, EPWMxA is invalid.

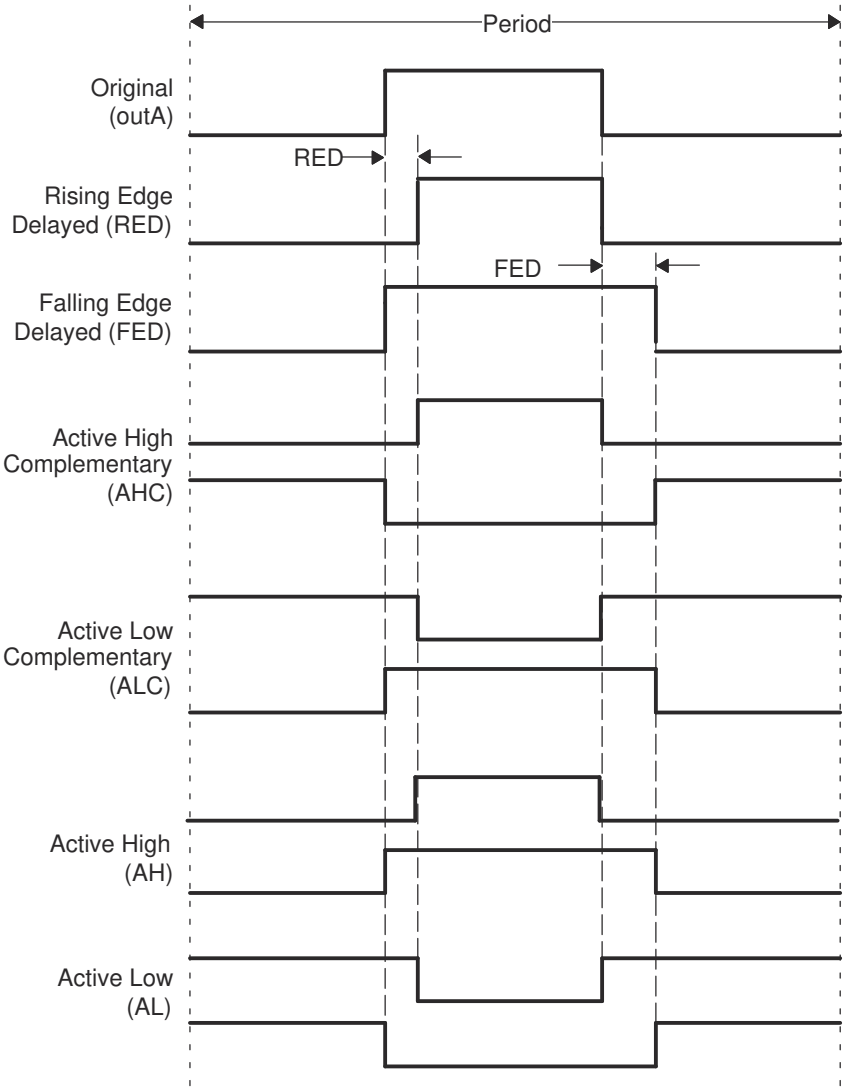


Figure 26-36. Dead-Band Waveforms for Typical Cases (0% < Duty < 100%)

The dead-band submodule supports independent values for rising-edge (RED) and falling-edge (FED) delays. The amount of delay is programmed using the DBRED and DBFED registers. These are 10-bit registers and their value represents the number of time-base clock, TBCLK, periods by which a signal edge is delayed. For example, the formula to calculate falling-edge-delay and rising-edge-delay is:

FED = DBFED × T_{TBCLK}

RED = DBRED × T_{TBCLK}

Where T_{TBCLK} is the period of TBCLK, the prescaled version of EPWMCLK.

For convenience, delay values for various TBCLK options are shown in Table 26-11. The ePWM input clock frequency that these delay values been computed by is 100 MHz.

Table 26-11. Dead-Band Delay Values in µS as a Function of DBFED and DBRED

Dead-Band Value	Dead-Band Delay in µS		
DBFED, DBRED	TBCLK = EPWMCLK/1	TBCLK = EPWMCLK /2	TBCLK = EPWMCLK/4
1	0.01 µS	0.02 µS	0.04 µS
5	0.05 µS	0.10 µS	0.20 µS
10	0.10 µS	0.20 µS	0.40 µS
100	1.00 µS	2.00 µS	4.00 µS
200	2.00 µS	4.00 µS	8.00 µS
400	4.00 µS	8.00 µS	16.00 µS
500	5.00 µS	10.00 µS	20.00 µS
600	6.00 µS	12.00 µS	24.00 µS
700	7.00 µS	14.00 µS	28.00 µS
800	8.00 µS	16.00 µS	32.00 µS
900	9.00 µS	18.00 µS	36.00 µS
1000	10.00 µS	20.00 µS	40.00 µS

When half-cycle clocking is enabled, the formula to calculate the falling-edge-delay and rising-edge-delay becomes:

$FED = DBFED \times T_{TBCLK}/2$

$RED = DBRED \times T_{TBCLK}/2$

26.8 PWM Chopper (PC) Submodule

The PWM chopper submodule allows a high-frequency carrier signal to modulate the PWM waveform generated by the action-qualifier and dead-band submodules. This capability is important if pulse transformer-based gate drivers to control the power switching elements are needed.

Figure 26-37 illustrates the PWM chopper submodule within the ePWM.

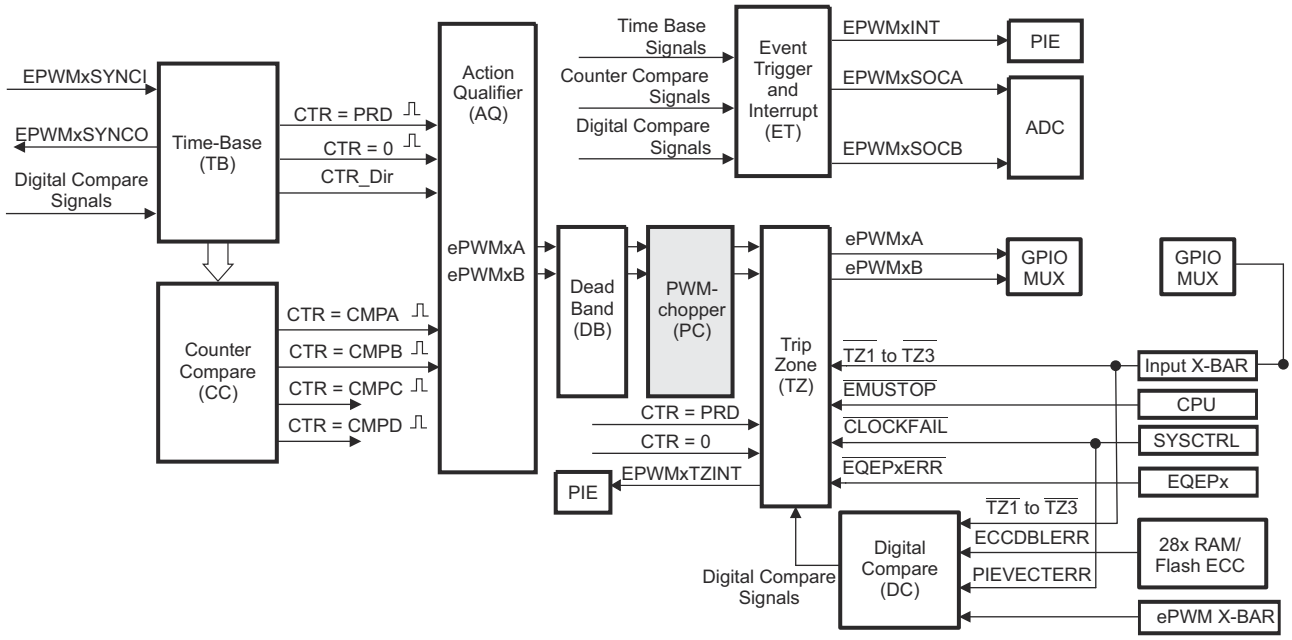


Figure 26-37. PWM Chopper Submodule

26.8.1 Purpose of the PWM Chopper Submodule

The key functions of the PWM chopper submodule are:

- Programmable chopping (carrier) frequency
- Programmable pulse width of first pulse
- Programmable duty cycle of second and subsequent pulses
- Can be fully bypassed if not required

26.8.2 Operational Highlights for the PWM Chopper Submodule

Figure 26-38 shows the operational details of the PWM chopper submodule. The carrier clock is derived from EPWMCLK. The clock frequency and duty cycle are controlled using the CHPFREQ and CHPDUTY bits in the PCCTL register. The one-shot block is a feature that provides a high energy first pulse to make sure hard and fast power switch turn on, while the subsequent pulses sustain pulses, making sure the power switch remains on. The one-shot width is programmed using the OSHTWTH bits. The PWM chopper submodule can be fully disabled (bypassed) using the CHPEN bit.

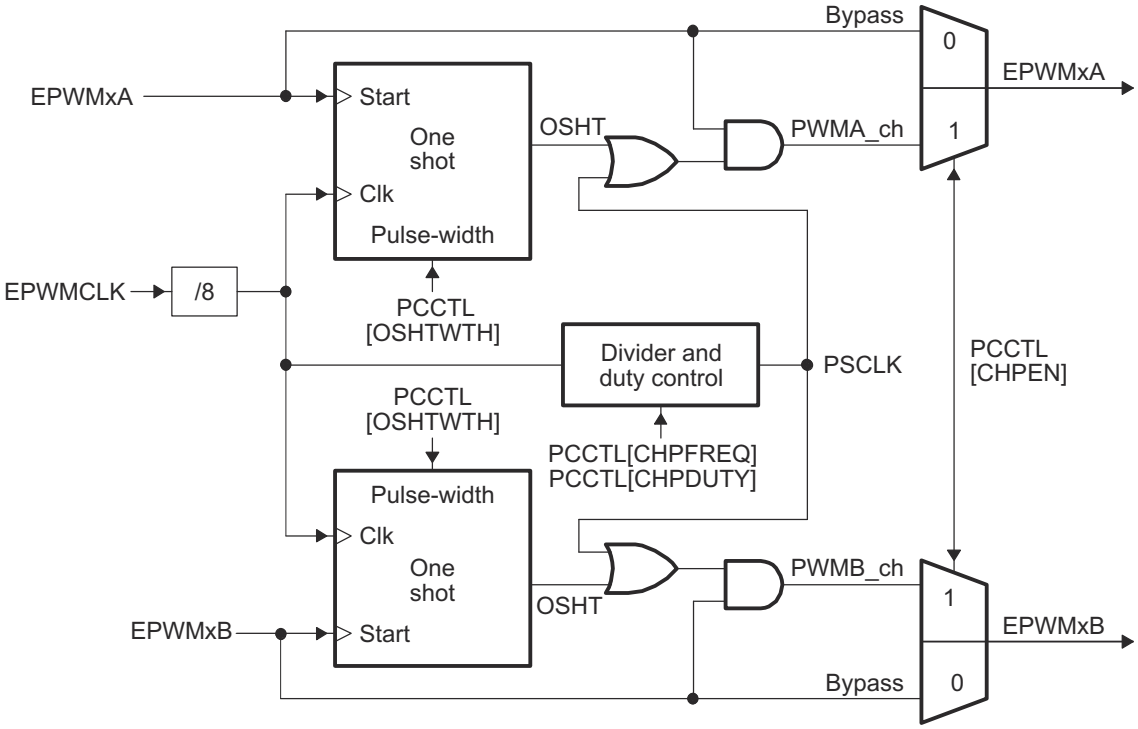


Figure 26-38. PWM Chopper Submodule Operational Details

26.8.3 Waveforms

Figure 26-39 shows simplified waveforms of the chopping action only; one-shot and duty-cycle control are not shown. Details of the one-shot and duty-cycle control are discussed in the following sections.

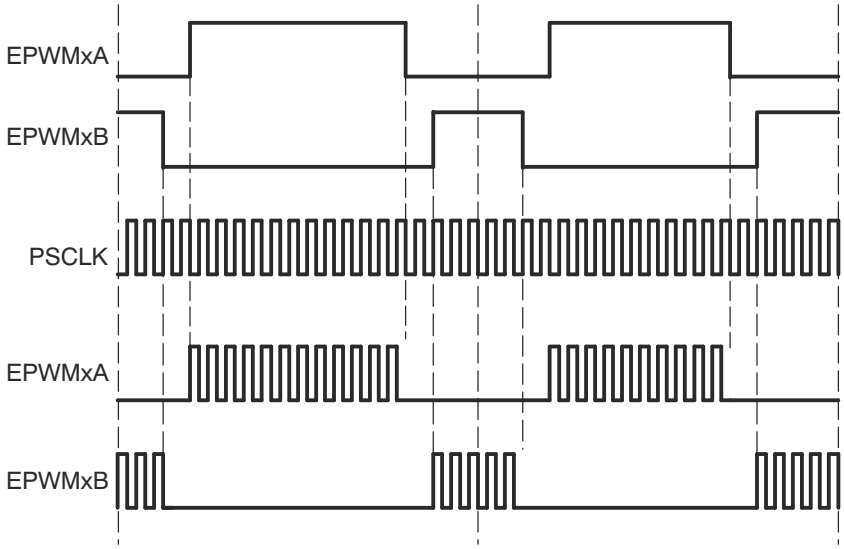


Figure 26-39. Simple PWM Chopper Submodule Waveforms Showing Chopping Action Only

26.8.3.1 One-Shot Pulse

The width of the first pulse can be programmed to any of 16 possible pulse width values. The width or period of the first pulse is given by:

$$T_{1stpulse} = T_{EPWMCLK} \times 8 \times OSHTWTH$$

Where $T_{EPWMCLK}$ is the period of the system clock (EPWMCLK) and OSHTWTH is the four control bits (value from 1 to 16)

Figure 26-40 shows the first and subsequent sustaining pulses and Table 26-12 gives the possible pulse width values for a EPWMCLK = 80 MHz.

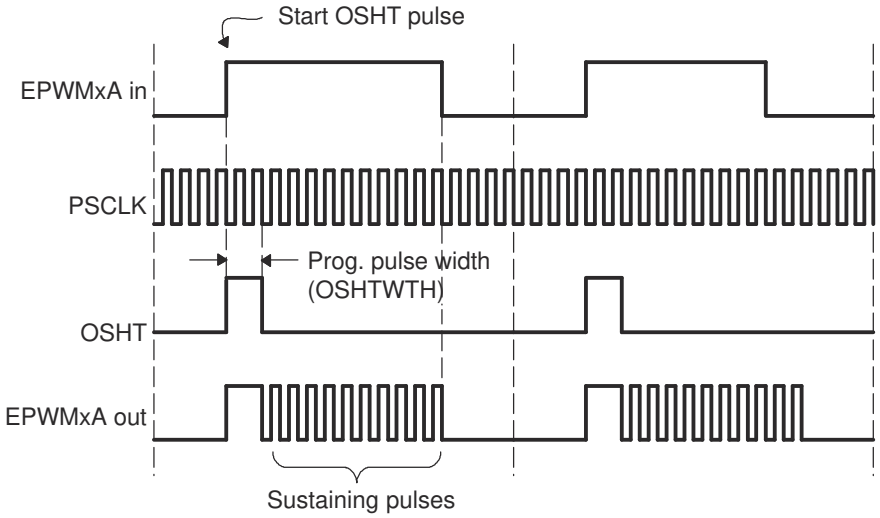


Figure 26-40. PWM Chopper Submodule Waveforms Showing the First Pulse and Subsequent Sustaining Pulses

Table 26-12. Possible Pulse Width Values for EPWMCLK = 80 MHz

OSHTWTH (hex)	Pulse Width (nS)
0	100
1	200
2	300
3	400
4	500
5	600
6	700
7	800
8	900
9	1000
A	1100
B	1200
C	1300
D	1400
E	1500
F	1600

26.8.3.2 Duty Cycle Control

Pulse transformer-based gate drive designs need to comprehend the magnetic properties or characteristics of the transformer and associated circuitry. Saturation is one such consideration. To assist the gate drive designer, the duty cycles of the second and subsequent pulses have been made programmable. These sustaining pulses make sure the correct drive strength and polarity is maintained on the power switch gate during the on period, and hence a programmable duty cycle allows a design to be tuned or optimized using software control.

Figure 26-41 shows the duty cycle control that is possible by programming the CHPDUTY bits. One of seven possible duty ratios can be selected ranging from 12.5% to 87.5%.

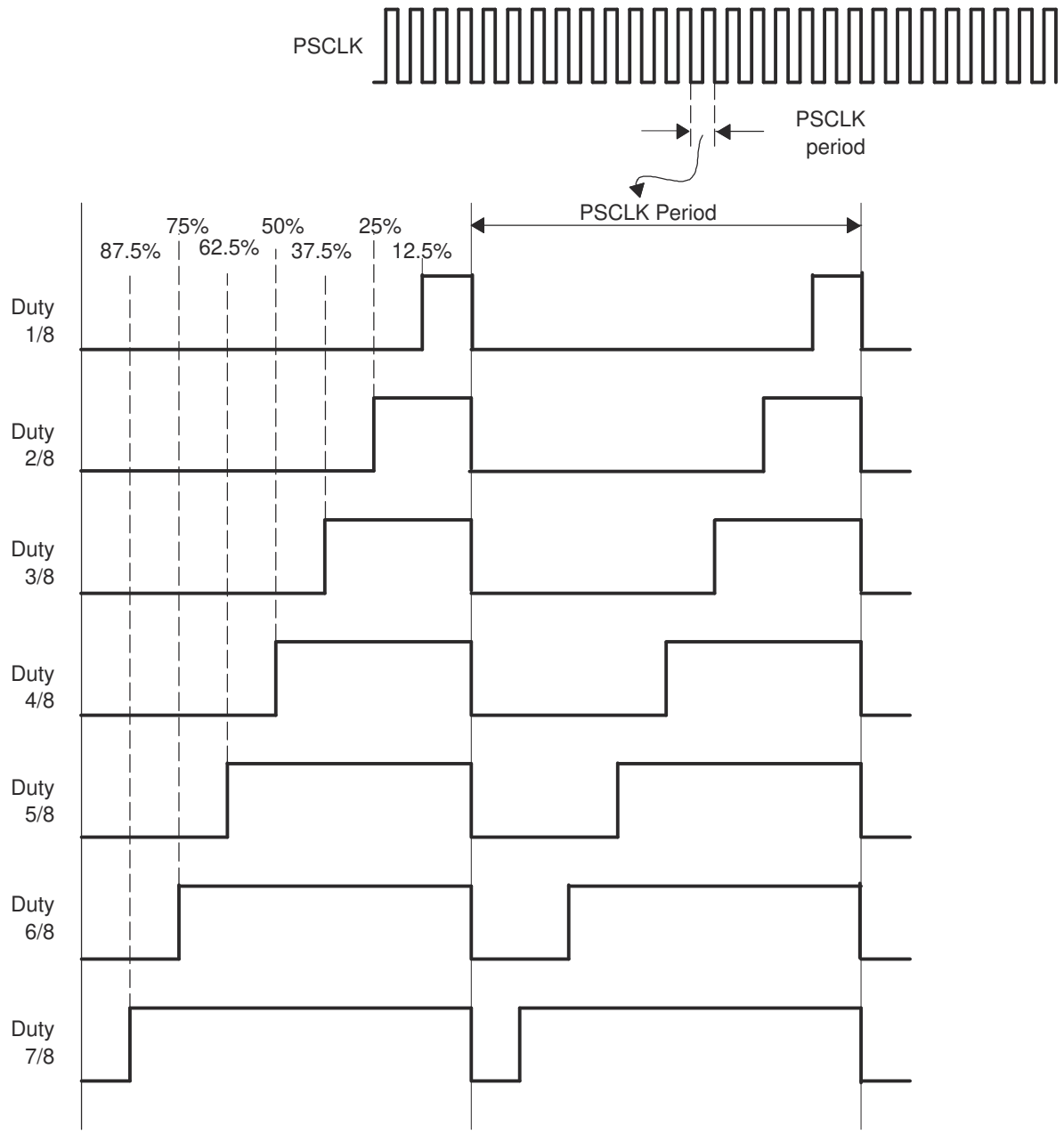


Figure 26-41. PWM Chopper Submodule Waveforms Showing the Pulse Width (Duty Cycle) Control of Sustaining Pulses

26.9 Trip-Zone (TZ) Submodule

Each ePWM module is connected to six $\overline{TZn}$ signals ($\overline{TZ1}$ to $\overline{TZ6}$). $\overline{TZ1}$ to $\overline{TZ3}$ are sourced from the GPIO mux. $\overline{TZ4}$ is sourced from an inverted EQEPxERR signal on those devices with an EQEP module. $\overline{TZ5}$ is connected to the system clock fail logic, and $\overline{TZ6}$ is sourced from the EMUSTOP output from the CPU. These signals indicate external fault or trip conditions, and the ePWM outputs can be programmed to respond accordingly when faults occur.

Figure 26-42 illustrates the trip-zone submodule within the ePWM.

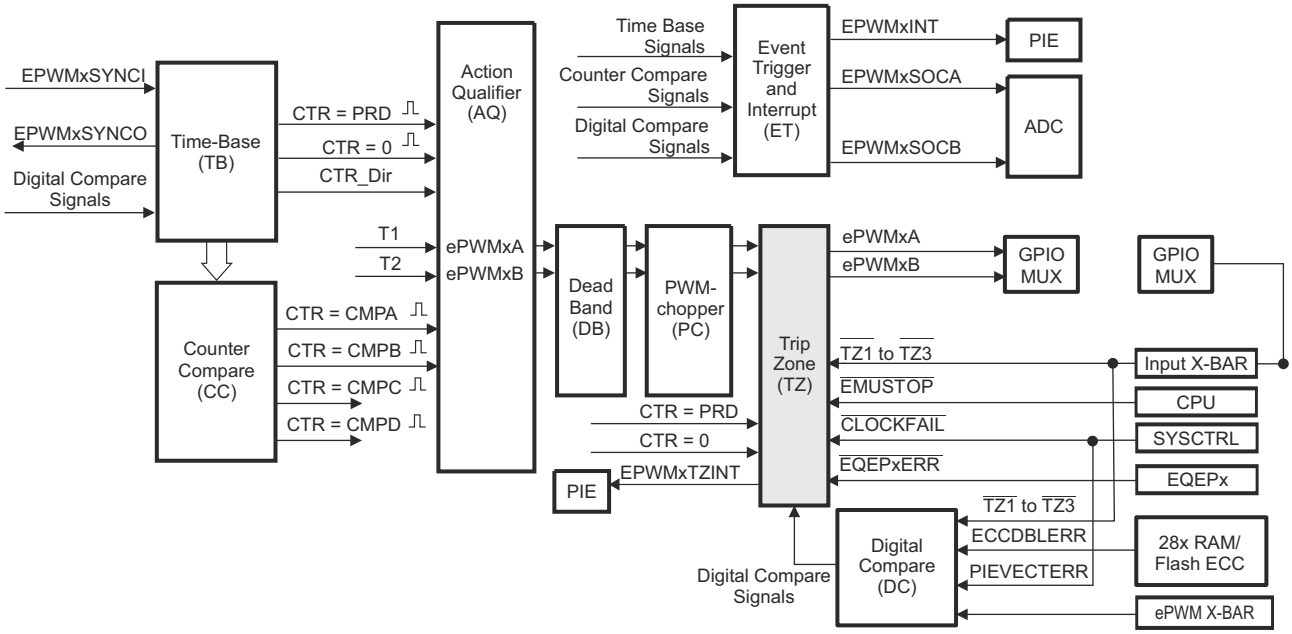


Figure 26-42. Trip-Zone Submodule

26.9.1 Purpose of the Trip-Zone Submodule

The key functions of the trip-zone submodule are:

- Trip inputs $\overline{TZ1}$ to $\overline{TZ6}$ can be flexibly mapped to any ePWM module.
- Upon a fault condition, outputs EPWMxA and EPWMxB can be forced to one of the following:
  - High
  - Low
  - High-impedance
  - No action taken
- Support for one-shot trip (OSHT) for major short circuits or over-current conditions.
- Support for cycle-by-cycle tripping (CBC) for current limiting operation.
- Support for digital compare tripping (DC) based on state of on-chip analog comparator module outputs and $\overline{TZ1}$ to $\overline{TZ3}$ signals.
- Each trip-zone input and digital compare (DC) submodule DCAEVT1/2 or DCBEVT1/2 force event can be allocated to either one-shot or cycle-by-cycle operation.
- Interrupt generation is possible on any trip-zone input.
- Software-forced tripping is also supported.
- The trip-zone submodule can be fully bypassed if the submodule is not required.

26.9.2 Operational Highlights for the Trip-Zone Submodule

The following sections describe the operational highlights and configuration options for the trip-zone submodule.

The trip-zone signals $\overline{TZ1}$ to $\overline{TZ6}$ (also collectively referred to as $\overline{TZn}$) are active-low input signals. When one of these signals goes low, or when a DCAEVT1/2 or DCBEVT1/2 force happens based on the TZDCSEL register event selection, the indication is that a trip event has occurred. Each ePWM module can be individually configured to ignore or use each of the trip-zone signals or DC events. Which trip-zone signals or DC events are used by a particular ePWM module is determined by the TZSEL register for that specific ePWM module. The trip-zone signals can or cannot be synchronized to the ePWMclock (EPWMCLK) and digitally filtered within the GPIO MUX block. A minimum of 3*TBCLK low pulse width on $\overline{TZn}$ inputs is sufficient to trigger a fault condition on the ePWM module. If the pulse width is less than this, the trip condition cannot be latched by CBC or OST latches. The asynchronous trip makes sure that if clocks are missing for any reason, the outputs can still be tripped by a valid event present on $\overline{TZn}$ inputs. The GPIOs or peripherals must be appropriately configured. For more information, see the *System Control and Interrupts* chapter.

Each $\overline{TZn}$ input can be individually configured to provide either a cycle-by-cycle or one-shot trip event for an ePWM module. DCAEVT1 and DCBEVT1 events can be configured to directly trip an ePWM module or provide a one-shot trip event to the module. Likewise, DCAEVT2 and DCBEVT2 events can also be configured to directly trip an ePWM module or provide a cycle-by-cycle trip event to the module. This configuration is determined by the TZSEL[DCAEVT1/2], TZSEL[DCBEVT1/2], TZSEL[CBCn], and TZSEL[OSHTn] control bits (where n corresponds to the trip input), respectively.

- **Cycle-by-Cycle (CBC):**

  When a cycle-by-cycle trip event occurs, the action specified in the TZCTL[TZA] and TZCTL[TZB] bits is carried out immediately on the EPWMxA and EPWMxB outputs. Table 26-13 lists some of the possible actions. Independent actions can be specified based on the occurrence of the event while the counter is counting up or while the counter is counting down by appropriately configuring bits in the TZCTL2 register. Actions specified in the TZCTL2 register take effect only when the ETZE bit in TZCTL2 is set.

  Additionally, when a cycle-by-cycle trip event occurs, the cycle-by-cycle trip event flag (TZFLG[CBC]) is set and a EPWMx_TZINT interrupt is generated when enabled in the TZEINT register and interrupt controller. A corresponding flag for the event that caused the CBC event is also set in register TZCBCFLG.

  If the CBC interrupt is enabled using the TZEINT register and DCAEVT2 or DCBEVT2 are selected as CBC trip sources using the TZSEL register, it is not necessary to also enable the DCAEVT2 or DCBEVT2 interrupts in the TZEINT register, as the DC events trigger interrupts through the CBC mechanism.

  The specified condition on the inputs is automatically cleared based on the selection made with TZCLR[CBCPULSE] if the trip event is no longer present. Therefore, in this mode, the trip event is cleared or reset every PWM cycle. The TZFLG[CBC] and TZCBCFLG flag bits remain set until the flag bits are manually cleared by writing to the TZCLR[CBC] and TZCBCCLR flag bits. If the cycle-by-cycle trip event is still present when the TZFLG[CBC] and TZCBCFLG register bits are cleared, then these bits are again immediately set.

- **One-Shot (OSHT):**

  When a one-shot trip event occurs, the action specified in the TZCTL[TZA] and TZCTL[TZB] bits is carried out immediately on the EPWMxA and EPWMxB output. Table 26-13 lists some of the possible actions. Independent actions can be specified based on the occurrence of the event while the counter is counting up and while the counter is counting down by appropriately configuring bits in TZCTL2 register. Actions specified in TZCTL2 register take effect only when ETZE bit in TZCTL2 is set.

  Additionally, when a one-shot trip event occurs, the one-shot trip event flag (TZFLG[OST]) is set and a EPWMx_TZINT interrupt is generated when enabled in the TZEINT register and interrupt controller. A corresponding flag for the event that caused the OST event is also set in register TZOSTFLG. The one-shot trip condition must be cleared manually by writing to the TZCLR[OST] bit. If desired, the TZOSTFLG register bit can be cleared by manually writing to the corresponding bit in the TZOSTCLR register.

If the one-shot interrupt is enabled using the TZEINT register and DCAEVT1 or DCBEVT1 are selected as OSHT trip sources using the TZSEL register, it is not necessary to also enable the DCAEVT1 or DCBEVT1 interrupts in the TZEINT register, as the DC events trigger interrupts through the OSHT mechanism.

Note

Clear the TZFLG and TZOSTFLG flags after making sure that the TRIPIN source of the OST has become inactive. Otherwise, if interrupts are enabled, depending on when the flags are cleared, an OST interrupt can occur and the OST flags are zero.

- **Digital Compare Events (DCAEVT1/2 and DCBEVT1/2):**

  A digital compare DCAEVT1/2 or DCBEVT1/2 event is generated based on a combination of the DCAH/DCAL and DCBH/DCBL signals as selected by the TZDCSEL register. The signals which source the DCAH/DCAL and DCBH/DCBL signals are selected using the DCTRIPSEL register and can be either trip zone input pins or analog comparator CMPSSx signals. For more information on the digital compare submodule signals, see Section 26.11.

  When a digital compare event occurs, the action specified in the TZCTL[DCAEVT1/2] and TZCTL[DCBEVT1/2] bits is carried out immediately on the EPWMxA and EPWMxB output. Table 26-13 lists the possible actions. Independent actions can be specified based on the occurrence of the event while the counter is counting up and while the counter is counting down by appropriately configuring bits in TZCTLDCA and TZCTLDCB register. Actions specified in TZCTLDCA and TZCTLDCB registers take effect only when ETZE bit in TZCTL2 is set.

  In addition, the relevant DC trip event flag (TZFLG[DCAEVT1/2] / TZFLG[DCBEVT1/2]) is set and a EPWMx_TZINT interrupt is generated when enabled in the TZEINT register and interrupt controller.

  The specified condition on the pins is automatically cleared when the DC trip event is no longer present. The TZFLG[DCAEVT1/2] or TZFLG[DCBEVT1/2] flag bit remains set until the flag is manually cleared by writing to the TZCLR[DCAEVT1/2] or TZCLR[DCBEVT1/2] bit. If the DC trip event is still present when the TZFLG[DCAEVT1/2] or TZFLG[DCBEVT1/2] flag is cleared, then the flag is again immediately set.

The action taken when a trip event occurs can be configured individually for each of the ePWM output pins by way of the TZCTL, TZCTL2, TZCTLDCA, and TZCTLDCB register bit fields. Some of the possible actions, shown in Table 26-13, can be taken on a trip event.

Table 26-13. Possible Actions On a Trip Event

TZCTL Register Bitfield Settings	EPWMxA and EPWMxB	Comment
0,0	High-Impedance	Tripped
0,1	Force to High State	Tripped
1,0	Force to Low State	Tripped
1,1	No Change	Do Nothing. No change is made to the output.

Example 26-1. Trip-Zone Configurations

Scenario A:
A one-shot trip event on $\overline{TZ1}$ pulls both EPWM1A, EPWM1B low and also forces EPWM2A and EPWM2B high.

- Configure the ePWM1 registers as follows:
  - TZSEL[OSHT1] = 1: enables $\overline{TZ1}$ as a one-shot event source for ePWM1
  - TZCTL[TZA] = 2: EPWM1A is forced low on a trip event.
  - TZCTL[TZB] = 2: EPWM1B is forced low on a trip event.
- Configure the ePWM2 registers as follows:
  - TZSEL[OSHT1] = 1: enables $\overline{TZ1}$ as a one-shot event source for ePWM2
  - TZCTL[TZA] = 1: EPWM2A is forced high on a trip event.
  - TZCTL[TZB] = 1: EPWM2B is forced high on a trip event.

Scenario B:
A cycle-by-cycle event on $\overline{TZ5}$ pulls both EPWM1A, EPWM1B low.
A one-shot event on $\overline{TZ1}$ or $\overline{TZ6}$ puts EPWM2A into a high impedance state.

- Configure the ePWM1 registers as follows:
  - TZSEL[CBC5] = 1: enables $\overline{TZ5}$ as a cycle-by-cycle event source for ePWM1
  - TZCTL[TZA] = 2: EPWM1A is forced low on a trip event.
  - TZCTL[TZB] = 2: EPWM1B is forced low on a trip event.
- Configure the ePWM2 registers as follows:
  - TZSEL[OSHT1] = 1: enables $\overline{TZ1}$ as a one-shot event source for ePWM2
  - TZSEL[OSHT6] = 1: enables $\overline{TZ6}$ as a one-shot event source for ePWM2
  - TZCTL[TZA] = 0: EPWM2A is put into a high-impedance state on a trip event.
  - TZCTL[TZB] = 3: EPWM2B ignores the trip event.

Note

When configuring the GPIOs and INPUT X-BAR/EPWM X-BAR options, be aware that a change in the X-BAR input selections can cause an unwanted event. Therefore, ideally, set up the GPIO and X-BAR input configurations before enabling the ePWM Trip-Zone. If it is a requirement to change the GPIO/X-BAR configurations while the ePWM Trip-Zone is enabled, the user can turn off the TRIPs by clearing the TZSEL register and reconfiguring the TRIP selection (TZSEL) after the INPUT XBAR selection is changed.

26.9.3 Generating Trip Event Interrupts

Figure 26-43 and Figure 26-44 illustrate the trip-zone submodule control and interrupt logic, respectively. DCAEVT1/2 and DCBEVT1/2 signals are described in further detail in Section 26.11.

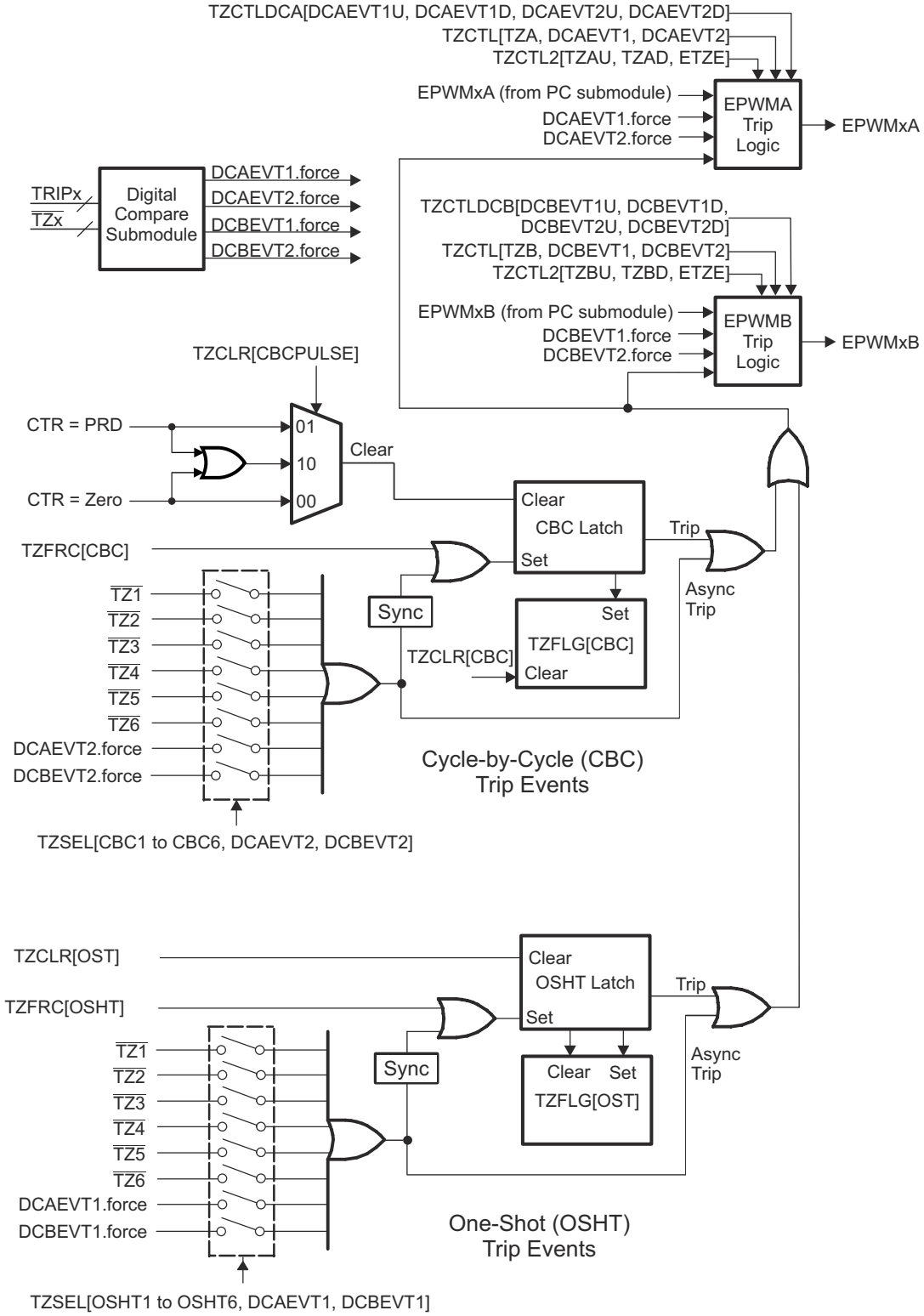


Figure 26-43. Trip-Zone Submodule Mode Control Logic

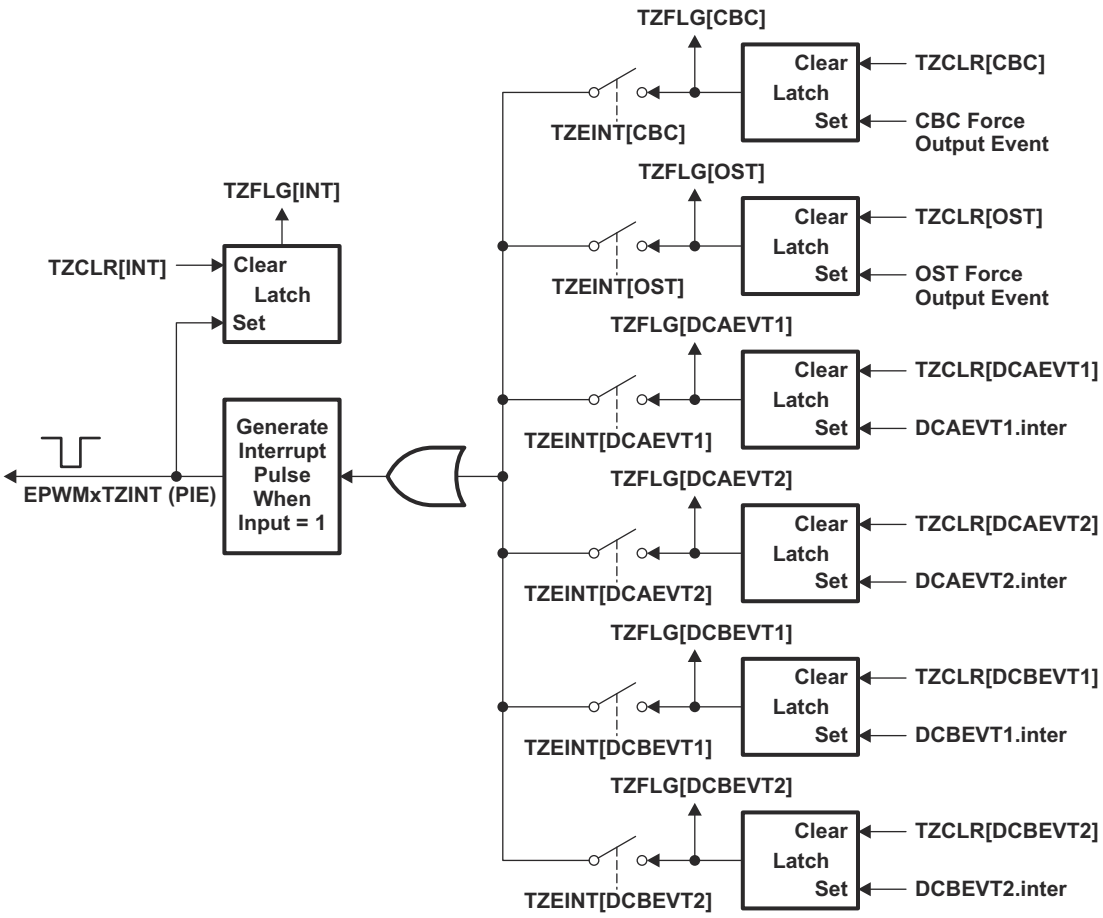


Figure 26-44. Trip-Zone Submodule Interrupt Logic

These individual flags for the CBC, OST and DCxEVTy can be used to detect the source of the EPWMxTZINT Interrupt. When multiple sources are used to generate the EPWMxTZINT interrupt, reading and clearing the flags takes different actions based on the specific event.

26.10 Event-Trigger (ET) Submodule

The key functions of the event-trigger submodule are:

- Receives event inputs generated by the time-base, counter-compare, and digital-compare submodules
- Uses the time-base direction information for up/down event qualification
- Uses prescaling logic to issue interrupt requests and ADC start of conversion at:
  - Every event
  - Every second event
  - Up to every fifteenth event
- Provides full visibility of event generation using event counters and flags
- Allows software forcing of Interrupts and ADC start of conversion

The event-trigger submodule manages the events generated by the time-base submodule, the counter-compare submodule, and the digital-compare submodule to generate an interrupt to the CPU and a start of conversion pulse to the ADC when a selected event occurs.

Figure 26-45 illustrates the event-trigger submodule within the ePWM.

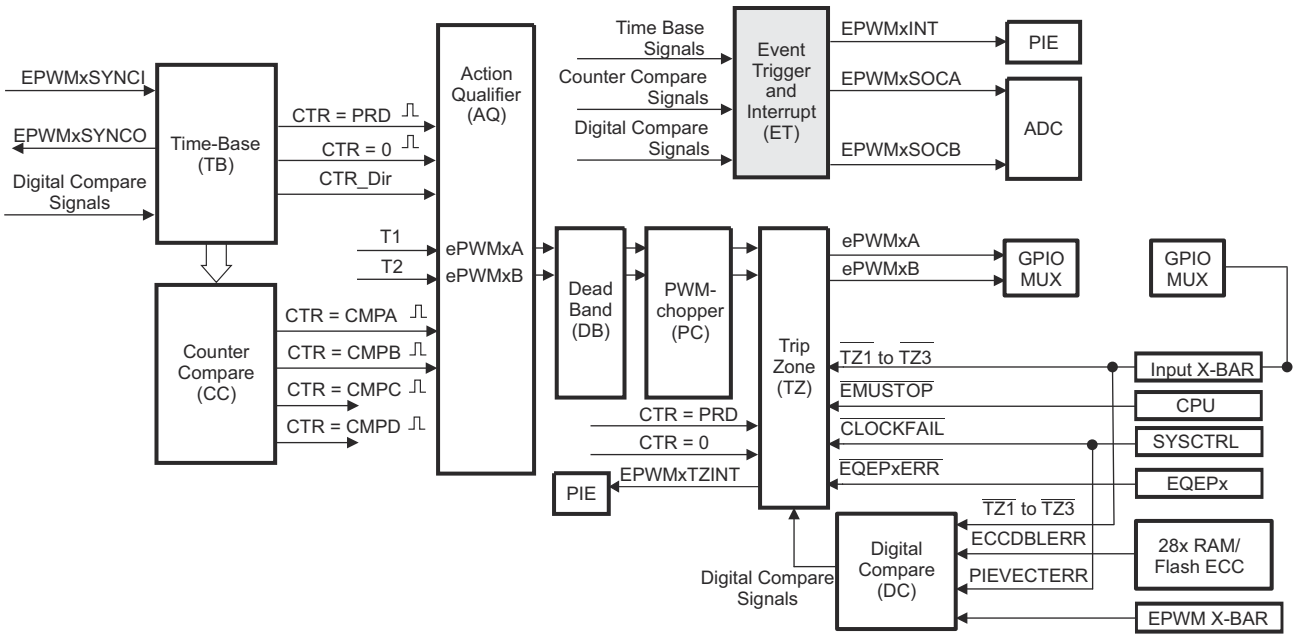


Figure 26-45. Event-Trigger Submodule

26.10.1 Operational Overview of the ePWM Event-Trigger Submodule

The event-trigger submodule monitors various event conditions (shown as inputs on the left side of Figure 26-46) and can be configured to prescale these events before issuing an Interrupt request or an ADC start of conversion. The event-trigger prescaling logic can issue Interrupt requests and ADC start of conversion at:

- Every event
- Every second event
- Up to every fifteenth event

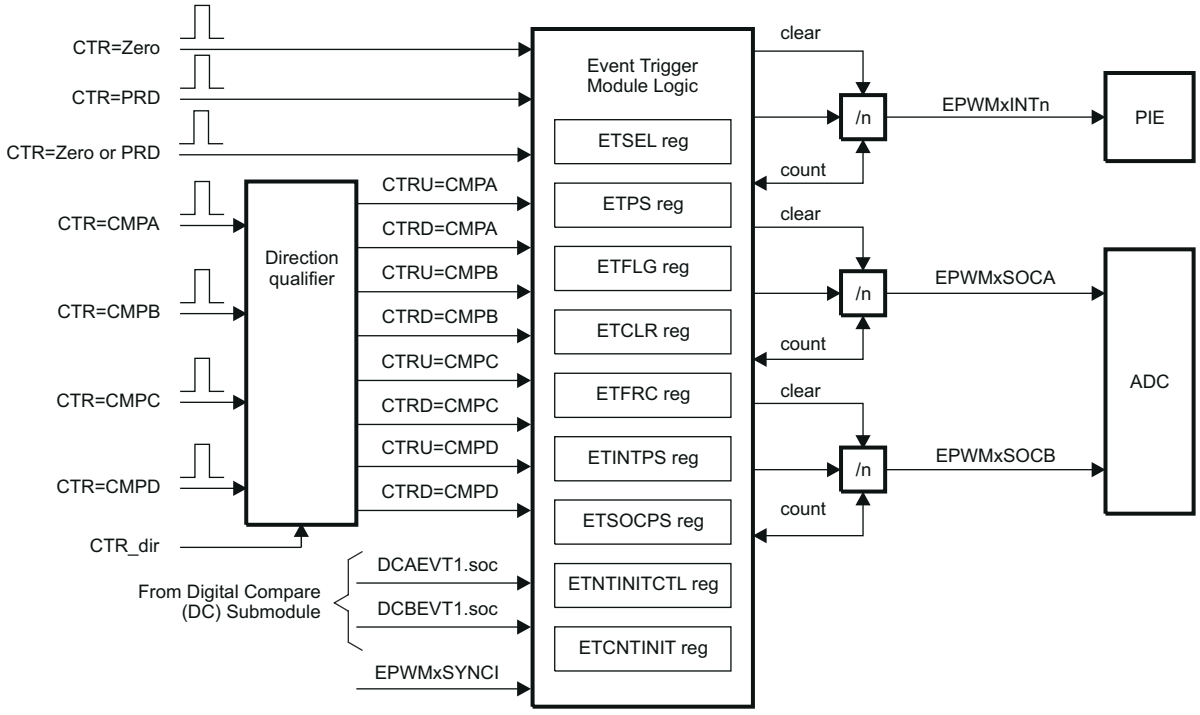


Figure 26-46. Event-Trigger Submodule Showing Event Inputs and Prescaled Outputs

- ETSEL - This selects which of the possible events trigger an interrupt or start an ADC conversion.
- ETPS - This programs the event prescaling options mentioned above.
- ETFLG - These are flag bits indicating status of the selected and prescaled events.
- ETCLR - These bits allow clearing the flag bits in the ETFLG register using software.
- ETFRC - These bits allow software forcing of an event. Useful for debugging or software intervention.
- ETINTPS - This programs the interrupt event prescaling options, supporting count and period up to 15 events.
- ETSOCPS - This programs the SOC event prescaling options, supporting count and period up to 15 events.
- ETCNTINITCTL - These bits enable ETCNTINIT initialization using SYNC event or using software force.
- ETCNTINIT - These bits allow initializing INT/SOCA/SOCB counters on SYNC events (or software force) with user programmed value.

A more detailed look at how the various register bits interact with the Interrupt and ADC start of conversion logic are shown in Figure 26-47, Figure 26-48, and Figure 26-49.

Figure 26-47 shows the event-trigger's interrupt generation logic. The interrupt-period (ETPS[INTPRD]) bits specify the number of events required to cause an interrupt pulse to be generated. The choices available are:

- Do not generate an interrupt.
- Generate an interrupt on every event.
- Generate an interrupt on every second event.
- Generate an interrupt on every third event.

The selection made on ETPS[INTPSSEL] bit determines whether ETINTPS register, INTCNT2 and INTPRD2 bit fields determine frequency of events (interrupt once every 0-15 events).

The event that can cause an interrupt is configured by the interrupt selection (ETSEL[INTSEL]) and (ETSEL[INTSELCMP]) bits. The event can be one of the following:

- Time-base counter equal to zero (TBCTR = 0x00).
- Time-base counter equal to period (TBCTR = TBPRD).
- Time-base counter equal to zero or period (TBCTR = 0x00 || TBCTR = TBPRD).
- Time-base counter equal to the compare A register (CMPA) when the timer is incrementing.
- Time-base counter equal to the compare A register (CMPA) when the timer is decrementing.
- Time-base counter equal to the compare B register (CMPB) when the timer is incrementing.
- Time-base counter equal to the compare B register (CMPB) when the timer is decrementing.
- Time-base counter equal to the compare C register (CMPC) when the timer is incrementing.
- Time-base counter equal to the compare C register (CMPC) when the timer is decrementing.
- Time-base counter equal to the compare D register (CMPD) when the timer is incrementing.
- Time-base counter equal to the compare D register (CMPD) when the timer is decrementing.

The number of events that have occurred can be read from the interrupt event counter ETPS[INTCNT] or ETINTPS[INTCNT2] register bits based off of the selection made using ETPS[INTPSSEL]. That is, when the specified event occurs the ETPS[INTCNT] or ETINTPS[INTCNT2] bits are incremented until the bits reach the value specified by ETPS[INTPRD] or ETINTPS[INTPRD2] determined again by the selection made in ETPS[INTPSSEL]. When ETPS[INTCNT] = ETPS[INTPRD], the counter stops counting and the counter output is set. The counter is only cleared when an interrupt is sent to the interrupt controller.

When ETPS[INTCNT] reaches ETPS[INTPRD], the following behavior occurs. [The following behavior is also applicable to ETINTPS[INTCNT2] and ETINTPS[INTPRD2]:

- If interrupts are enabled, ETSEL[INTEN] = 1 and the interrupt flag is clear, ETFLG[INT] = 0, then an interrupt pulse is generated and the interrupt flag is set, ETFLG[INT] = 1, and the event counter is cleared ETPS[INTCNT] = 0. The counter begins counting events again.
- If interrupts are disabled, ETSEL[INTEN] = 0, or the interrupt flag is set, ETFLG[INT] = 1, the counter stops counting events when the counter reaches the period value ETPS[INTCNT] = ETPS[INTPRD].
- If interrupts are enabled, but the interrupt flag is already set, then the counter holds the output high until the ENTFLG[INT] flag is cleared. This allows for one interrupt to be pending while one is serviced.

Writing a 0 to the INTPRD bits automatically clears the counter (INTCNT = 0) and the counter output resets (so no interrupts are generated). For all other writes to INTPRD, INTCNT retains the previous value. INTCNT resets when INTCNT overflows. Writing a 1 to the ETFRC[INT] bit increments the event counter INTCNT. The counter behaves as previously described when INTCNT = INTPRD. When INTPRD = 0, the counter is disabled and hence no events are detected and the ETFRC[INT] bit is also ignored. The same applies to ETINTPS[INTCNT2] and ETINTPS[INTPRD2].

The previous definition means that an interrupt on every event, on every second event, or on every third event if using the INTCNT and INTPRD can be generated. An interrupt on every event up to 15 events if using the INTCNT2 and INTPRD2 can be generated.

The INTCNT2 value can be initialized with the value from ETCNTINIT[INTINIT] based on the selection made in ETCNTINITCTL[INTINITEN]. When ETCNTINITCTL[INTINITEN] is set, then initialization of INTCNT2 counter with contents of ETCNTINIT[INTINIT] on a SYNC event or software force is determined by ETCNTINITCTL[INTINITFRC].

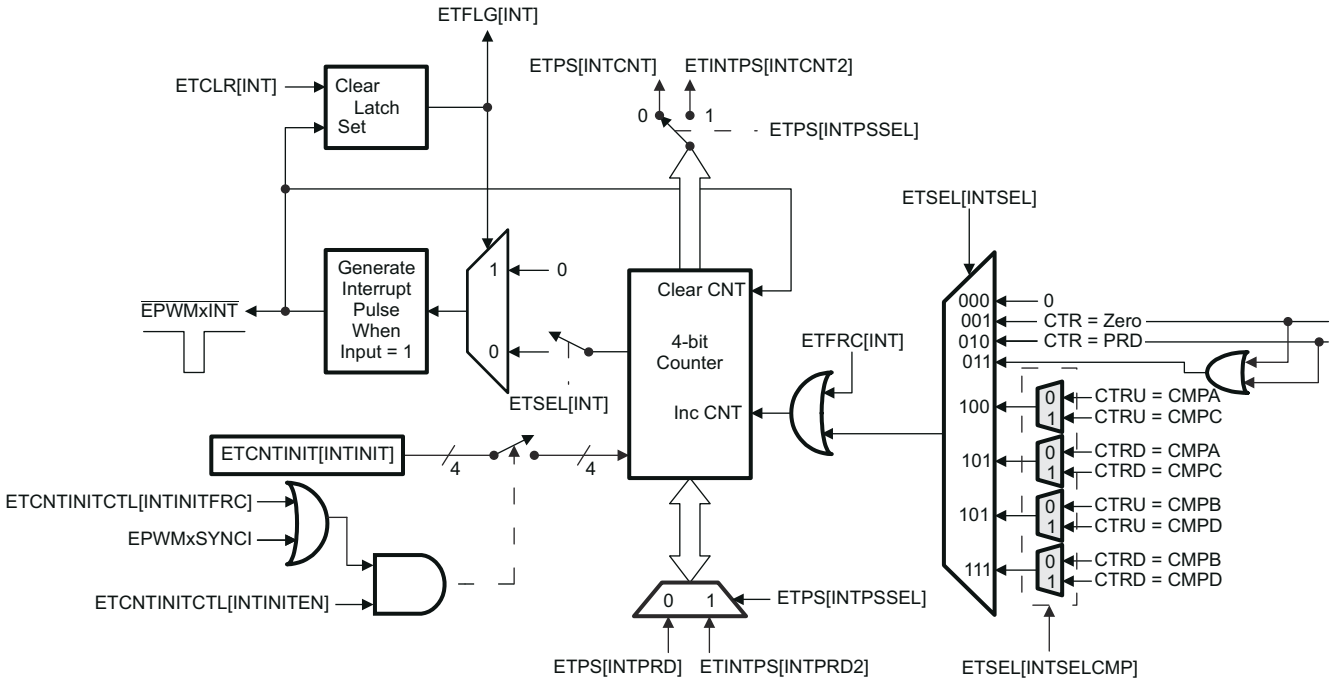
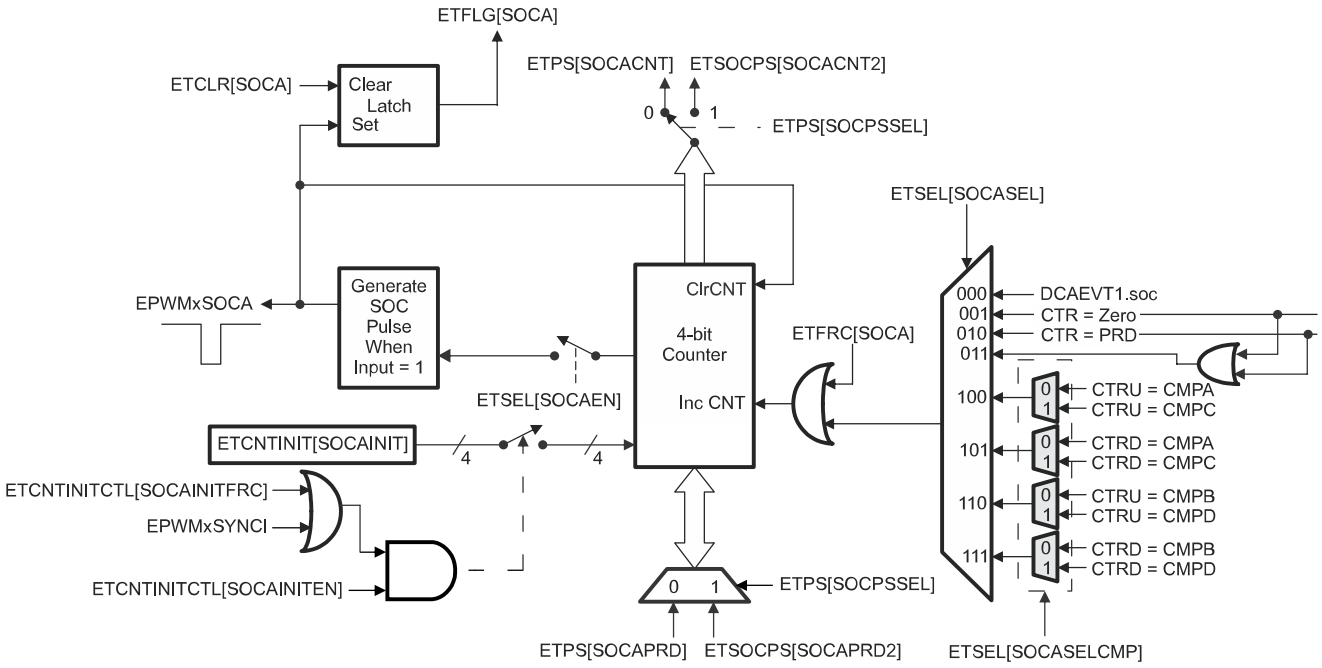


Figure 26-47. Event-Trigger Interrupt Generator

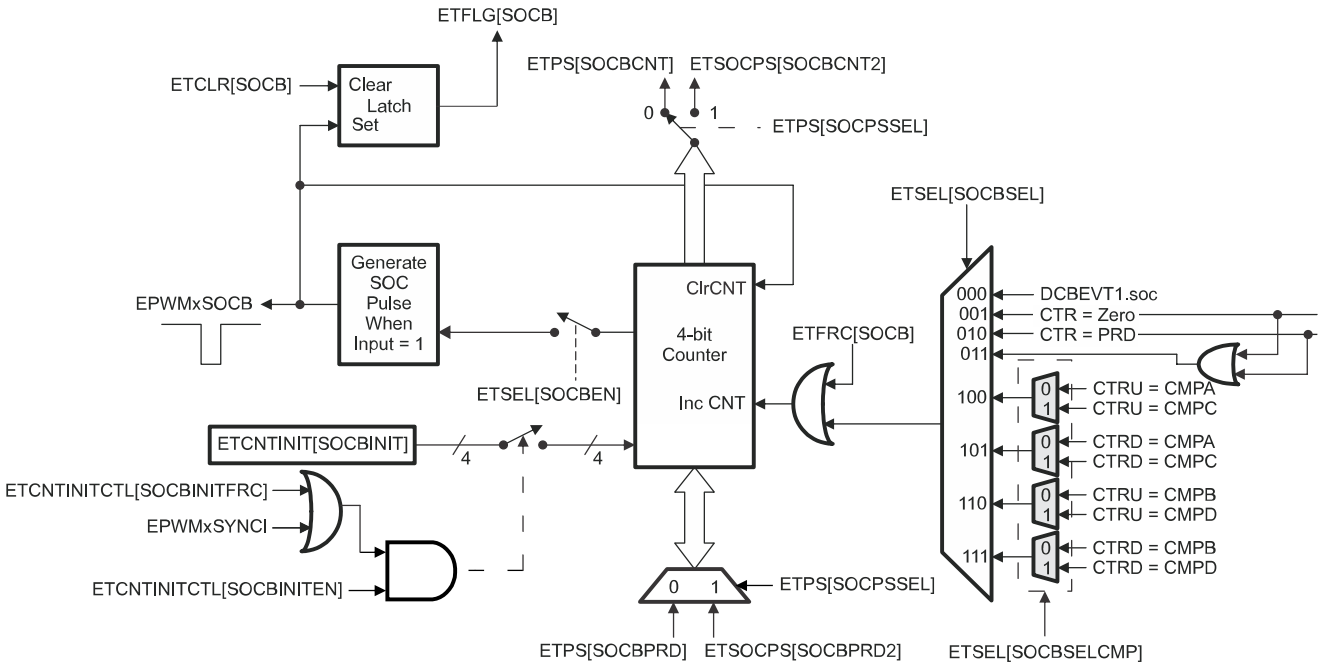
Figure 26-48 shows the operation of the event-trigger's start-of-conversion-A (SOCA) pulse generator. The enhancements include SOCASELCMP and SOCBSELCMP bit fields defined in the ETSEL register enable CMPC and CMPD events respectively to cause a start of conversion. The ETPS[SOCPSSEL] bit field determines whether SOCACNT2 and SOCAPRD2 take control or not. The ETPS[SOCACNT] counter and ETPS[SOCAPRD] period values behave similarly to the interrupt generator except that the pulses are continuously generated. That is, the pulse flag ETFLG[SOCA] is latched when a pulse is generated, but the interrupt generator does not stop further pulse generation. The enable and disable bit ETSEL[SOCAEN] stops pulse generation, but input events can still be counted until the period value is reached as with the interrupt generation logic. The event that triggers an SOCA and SOCB pulse can be configured separately in the ETSEL[SOCASEL] and ETSEL[SOCBSEL] bits. The possible events are the same events that can be specified for the interrupt generation logic with the addition of the DCAEVT1.soc and DCBEVT1.soc event signals from the digital compare (DC) submodule. The SOCACNT2 initialization scheme is very similar to the interrupt generator with respective enable, value initialize and SYNC or software force options.



NOTE: The DCAEVT1.soc signals are generated by the Digital Compare (DC) submodule in Section 26.11.

Figure 26-48. Event-Trigger SOCA Pulse Generator

Figure 26-49 shows the operation of the event-trigger's start-of-conversion-B (SOCB) pulse generator. The event-trigger's SOCB pulse generator operates the same way as the SOCA.



NOTE: The DCBEVT1.soc signals are generated by the Digital Compare (DC) submodule in Section 26.11.

Figure 26-49. Event-Trigger SOCB Pulse Generator

26.11 Digital Compare (DC) Submodule

Figure 26-50 illustrates where the digital compare (DC) submodule signals interface to other submodules in the ePWM system.

The eCAP input signals are sourced from the Input X-BAR signals as shown in Figure 26-51.

On this device, any of the GPIO pins can be flexibly mapped to be the trip-zone input and trip inputs to the trip-zone submodule and digital compare submodule. The Input X-BAR Input Select (INPUTxSELECT) register defines which GPIO pins gets assigned to be the trip-zone inputs / trip inputs.

The digital compare (DC) submodule compares signals external to the ePWM module (for instance, CMPSSx signals from the analog comparators) to directly generate PWM events/actions which then feed to the event-trigger, trip-zone, and time-base submodules. Additionally, blanking window functionality is supported to filter noise or unwanted pulses from the DC event signals.

Note

The user is responsible for driving the correct state on the selected pin before enabling the clock and configuring the trip input for the respective ePWM peripheral to avoid spurious latch of the TRIP signal.

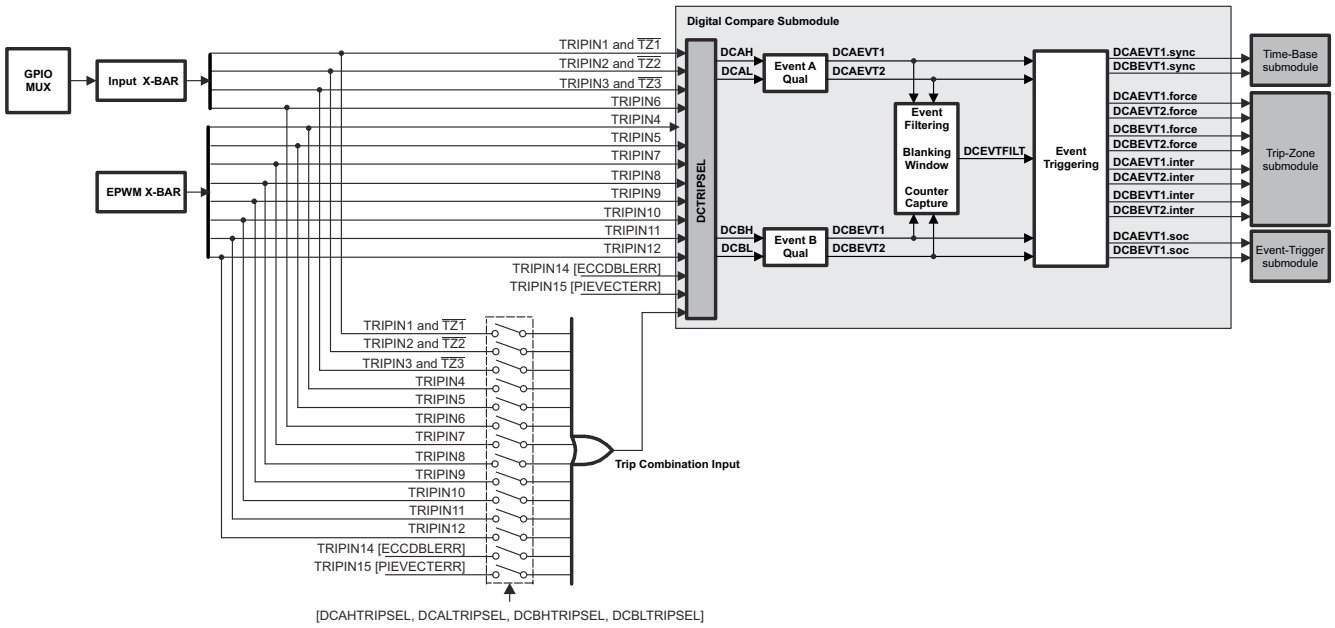


Figure 26-50. Digital-Compare Submodule High-Level Block Diagram

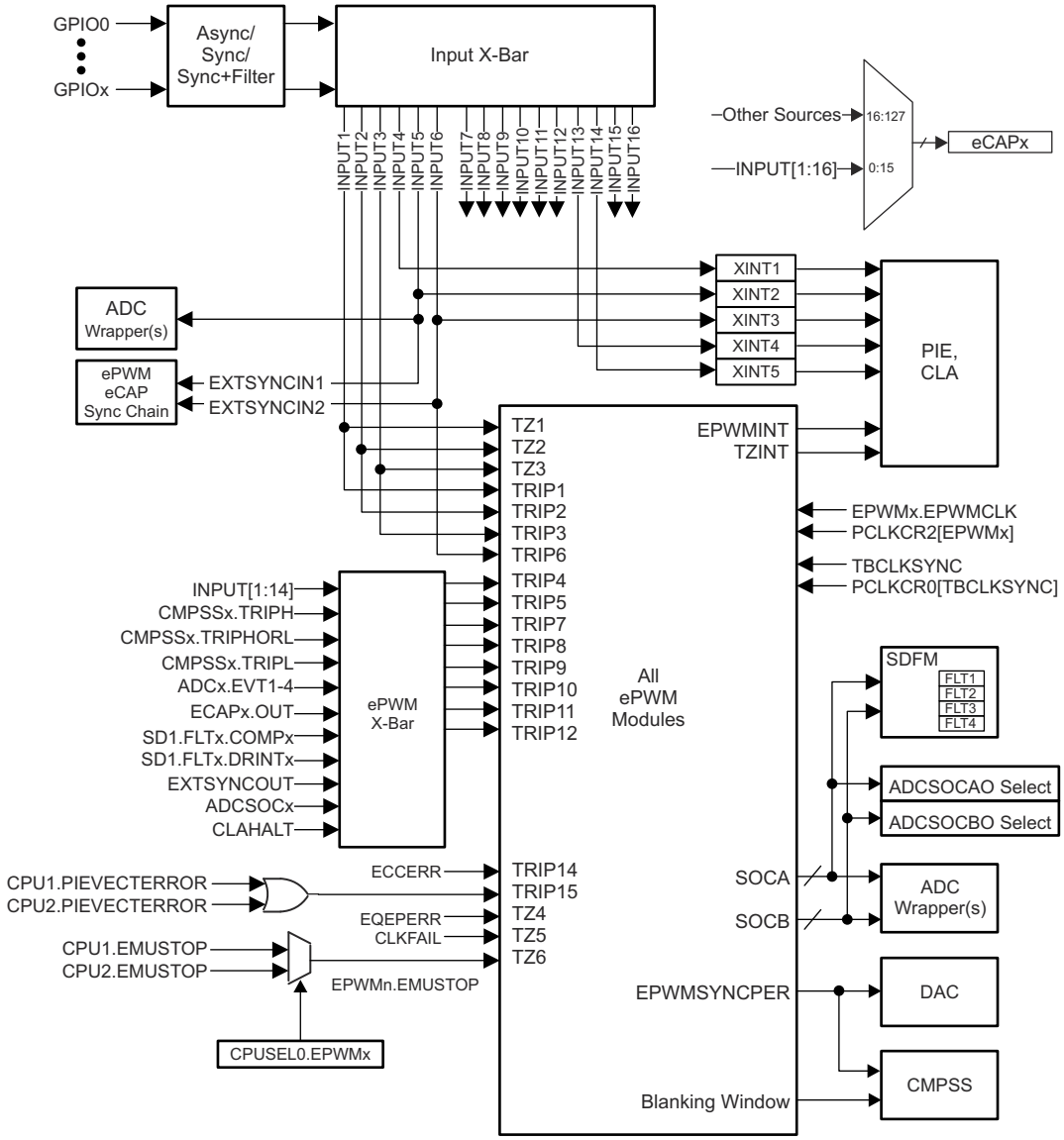


Figure 26-51. GPIO MUX-to-Trip Input Connectivity

26.11.1 Purpose of the Digital Compare Submodule

The key functions of the digital compare submodule are:

- Analog comparator (COMP) module outputs fed though the Input X-BAR, EPWM X-BAR, externally using the GPIO peripheral, interrupt controller signals, ECC error signals, TZ1, $\overline{TZ2}$, and $\overline{TZ3}$ inputs generate Digital Compare A High/Low (DCAH, DCAL) and Digital Compare B High/Low (DCBH, DCBL) signals.
- DCAH/L and DCBH/L signals trigger events that can then either be filtered or applied directly to the trip-zone, event-trigger, and time-base submodules to:
  - generate a trip zone interrupt
  - generate an ADC start of conversion
  - force an event
  - generate a synchronization event for synchronizing the ePWM module TBCTR.
- Event filtering (blanking window logic) can optionally blank the input signal to remove noise.

26.11.2 Enhanced Trip Action Using CMPSS

To allow multiple CMPSS at a time to affect DCA/BEVTx events and trip actions, there is a OR logic to bring together ALL trip inputs (up to 15) from sources external to the ePWM module and feed into DCAH, DCAL, DCBH, and DCBL as a “combinational input” using the DCTRIPSEL register. This is configured by selecting “Trip combination input” (value of 0xF) in the DCTRIPSEL register.

There is a discrete choice of which trip inputs to put through the combinational logic for generating the DCAH, DCAL, DCBH, and DCBL signals. This is achieved using the DCAHTRIPSEL, DCALTRIPSEL, DCBHTRIPSEL, and DCBLTRIPSEL register selections. Inputs selected for combinational input are passed through to the DCTRIPSEL register.

26.11.3 Using CMPSS to Trip the ePWM on a Cycle-by-Cycle Basis

When using the CMPSS to trip the ePWM on a cycle-by-cycle basis, steps can be taken to prevent an asserted comparator trip state in one PWM cycle from extending into the following cycle. The CMPSS can be used to signal a trip condition to the downstream ePWM modules. For applications like peak current mode control, only one trip event per PWM cycle is expected. Under certain conditions, it is possible for a sustained or late trip event (arriving near the end of a PWM cycle) to carry over into the next PWM cycle if precautions are not taken. If either the CMPSS Digital Filter or the ePWM Digital Compare (DC) submodule is configured to qualify the comparator trip signal, “N” number of clock cycles of qualification are introduced before the ePWM trip logic can respond to logic changes of the trip signal. Once an ePWM trip condition is qualified, the trip condition remains active for N clock cycles after the comparator trip signal has de-asserted. If a qualified comparator trip signal remains asserted within N clock cycles prior to the end of a PWM cycle, the trip condition is not cleared until after the following PWM cycle has started. Thus, the new PWM cycle detects a trip condition as soon as the cycle begins.

To avoid this undesired trip condition, the application can take steps to make sure that the qualified trip signal seen by the ePWM trip logic is deasserted prior to the end of each PWM cycle. This can be accomplished through various methods:

- Design the system such that a comparator trip is not asserted within N clock cycles prior to the end of the PWM cycle.
- Activate blanking of the comparator trip signal using the ePWM event filter at least two clock cycles prior to the PWMSYNCPER signal and continue blanking for at least N clock cycles into the next PWM cycle.
- If the CMPSS COMPxLATCH path is used, clear the COMPxLATCH at least N clock cycles prior to the end of the PWM cycle. The latch can be cleared by software (using COMPSTSCLR) or by generating an early PWMSYNCPER signal. The ePWM modules on this device include the ability to generate PWMSYNCPER upon a CMPC or CMPD match (using HRPCTL) for arbitrary PWMSYNCPER placement within the PWM cycle.

26.11.4 Operation Highlights of the Digital Compare Submodule

The following sections describe the operational highlights and configuration options for the digital compare submodule.

26.11.4.1 Digital Compare Events

As described in Section 26.11.1, trip zone inputs ($\overline{TZ1}$, $\overline{TZ2}$, and $\overline{TZ3}$) and CMPSSx signals from the analog comparator (COMP) module can be selected using the DCTRIPSEL bits to generate the Digital Compare A High and Low (DCAH/L) and Digital Compare B High and Low (DCBH/L) signals. Then, the configuration of the TZDCSEL register qualifies the actions on the selected DCAH/L and DCBH/L signals, which generate the DCAEVT1/2 and DCBEVT1/2 events (Event Qualification A and B).

Note

The $\overline{TZn}$ signals, when used as a DCEVT tripping functions, are treated as a normal input signal and can be defined to be active-high or active-low inputs. ePWM outputs are asynchronously tripped when either the $\overline{TZn}$, DCAEVTx.force, or DCBEVTx.force signals are active. For the condition to remain latched, a minimum of 3 * TBCLK sync pulse width is required. If pulse width is < 3*TBCLK sync pulse width, the trip condition can or can not get latched by CBC or OST latches.

The DCAEVT1/2 and DCBEVT1/2 events can then be filtered to provide a filtered version of the event signals (DCEVTFILT) or the filtering can be bypassed. Filtering is discussed further in Event Filtering. Either the DCAEVT1/2 and DCBEVT1/2 event signals or the filtered DCEVTFILT event signals can generate a force to the trip zone module, a TZ interrupt, an ADC SOC, or a PWM sync signal.

- **force signal:** DCAEVT1/2.force signals force trip zone conditions which either directly influence the output on the EPWMxA pin (using TZCTL, TZCTLDCA, TZCTLDCB register configurations) or, if the DCAEVT1/2 signals are selected as one-shot or cycle-by-cycle trip sources (using the TZSEL register), the DCAEVT1/2.force signals can effect the trip action using the TZCTL or TZCTL2 register configurations. The DCBEVT1/2.force signals behaves similarly, but affect the EPWMxB output pin instead of the EPWMxA output pin.

  The priority of conflicting actions on the TZCTL, TZCTL2, TZCTLDCA and TZCTLDCB registers is as follows (highest priority overrides lower priority):

  Output EPWMxA:
  - TZA (highest) -> DCAEVT1 -> DCAEVT2 (lowest)
  - TZAU (highest) -> DCAEVT1U -> DCAEVT2U (lowest)
  - TZAD (highest) -> DCAEVT1D -> DCAEVT2D (lowest)

  Output EPWMxB:
  - TZB (highest) -> DCBEVT1 -> DCBEVT2 (lowest)
  - TZBU (highest) -> DCBEVT1U -> DCBEVT2U (lowest)
  - TZBD (highest) -> DCBEVT1D -> DCBEVT2D (lowest)
- **interrupt signal:** DCAEVT1/2.interrupt signals generate trip zone interrupts to the interrupt controller. To enable the interrupt, set the DCAEVT1, DCAEVT2, DCBEVT1, or DCBEVT2 bits in the TZEINT register. Once one of these events occurs, an EPWMxTZINT interrupt is triggered, and the corresponding bit in the TZCLR register must be set to clear the interrupt.
- **soc signal:** The DCAEVT1.soc signal interfaces with the event-trigger submodule and can be selected as an event which generates an ADC start-of-conversion-A (SOCA) pulse using the ETSEL[SOCASEL] bit. Likewise, the DCBEVT1.soc signal can be selected as an event which generates an ADC start-of-conversion-B (SOCB) pulse using the ETSEL[SOCBSEL] bit.
- **sync signal:** The DCAEVT1.sync and DCBEVT1.sync events are ORed with the EPWMxSYNCI input signal and the TBCTL[SWFSYNC] signal to generate a synchronization pulse to the time-base counter.

Figure 26-52 and Figure 26-53 show how the DCxEVT1, DCxEVT2, or DCEVTFLT signals are processed to generate the digital compare A and B event force, interrupt, soc and sync signals.

In some of the applications like Phase Shifted Full Bridge (PSFB) Converters, it is required that different actions are taken on a CBC trip event and an OST trip event. This can be achieved using the DCxEVT1LAT.

- This latch can be cleared on CNT=0, CTR=PRD, and CNT=0 OR CTR=PRD events based on the setting of DCxCTL.EVTy.LATCLRSEL setting. This is similar to CBC latch clear mechanism.
- DCxEVTy.force signal can be chosen to be either the latched version or the unlatched version based on DCxCTL.EVTyLATSEL value.
- The status of DCxEVTyLAT signal can be accessed by reading DCxCTL.EVTyLAT field.

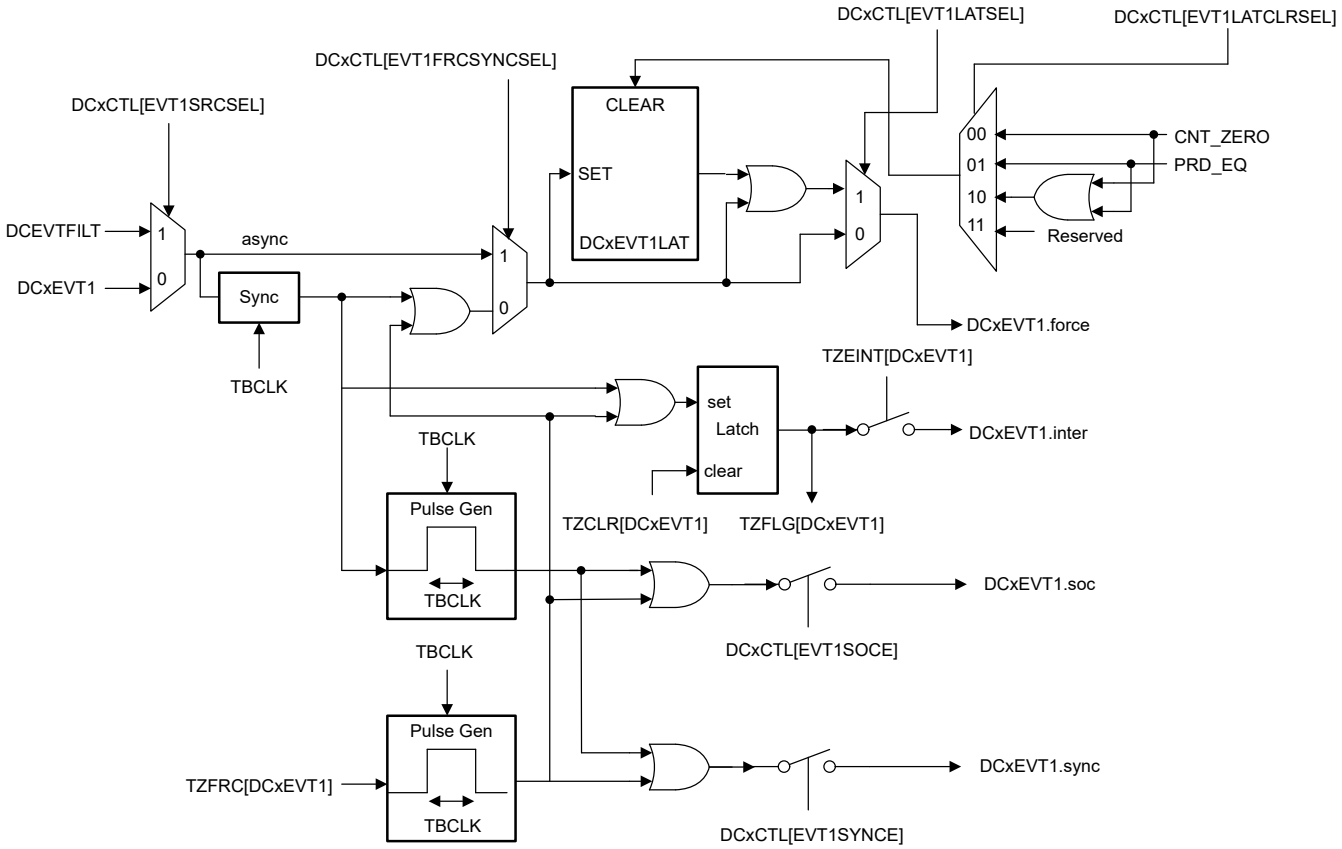


Figure 26-52. DCxEVT1 Event Triggering

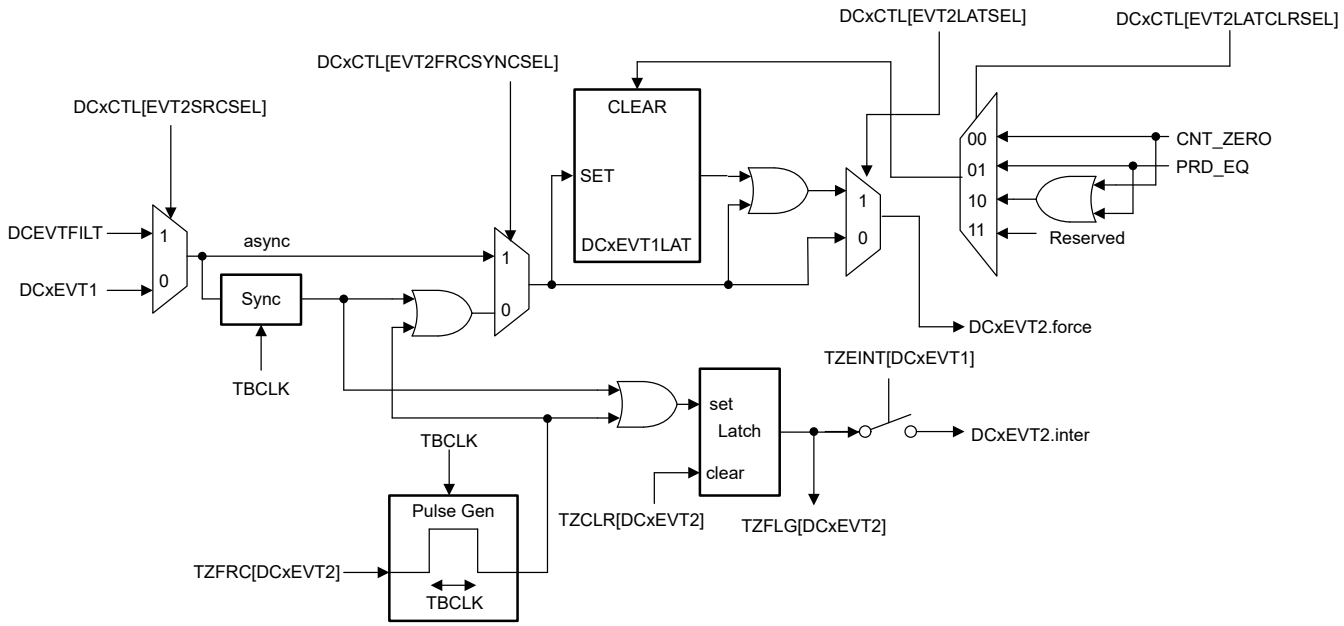


Figure 26-53. DCxEVT2 Event Triggering

26.11.4.2 Event Filtering

Blank Control Logic: The DCAEVT1/2 and DCBEVT1/2 events can be filtered using event filtering logic to remove noise by optionally blanking events for a certain period of time. This is useful for cases where the analog comparator outputs can be selected to trigger DCAEVT1/2 and DCBEVT1/2 events, and the blank control logic is used to filter out potential noise on the signal prior to tripping the PWM outputs or generating an interrupt or ADC start-of-conversion. Blank control logic is used to define a blanking window, which ignores all event occurrences on the signal while the window is active. The blanking window is configured in the DCFCTL, DCFOFFSET, and DCFWINDOW registers. The DCFCTL register enables the blanking window and aligns the blanking window to either a CTR = PRD pulse or a CTR = 0 pulse or both CTR = PRD and CTR = 0 as specified by DCFCTL[PULSESEL]. DCFCTL[SRCSEL] selects the DCxEVTy event source for the DCEVTFILT signal. An offset value in TBCLK counts is programmed into the DCFOFFSET register, which determines at what point after the CTR = PRD or CTR = 0 pulse the blanking window starts. The duration of the blanking window, in number of TBCLK counts after the offset counter expires, is written to the DCFWINDOW register by the application. Before and after the blanking window ends, events can generate soc, sync, interrupt, and force signals as before. Figure 26-54 shows the details of the event filtering logic.

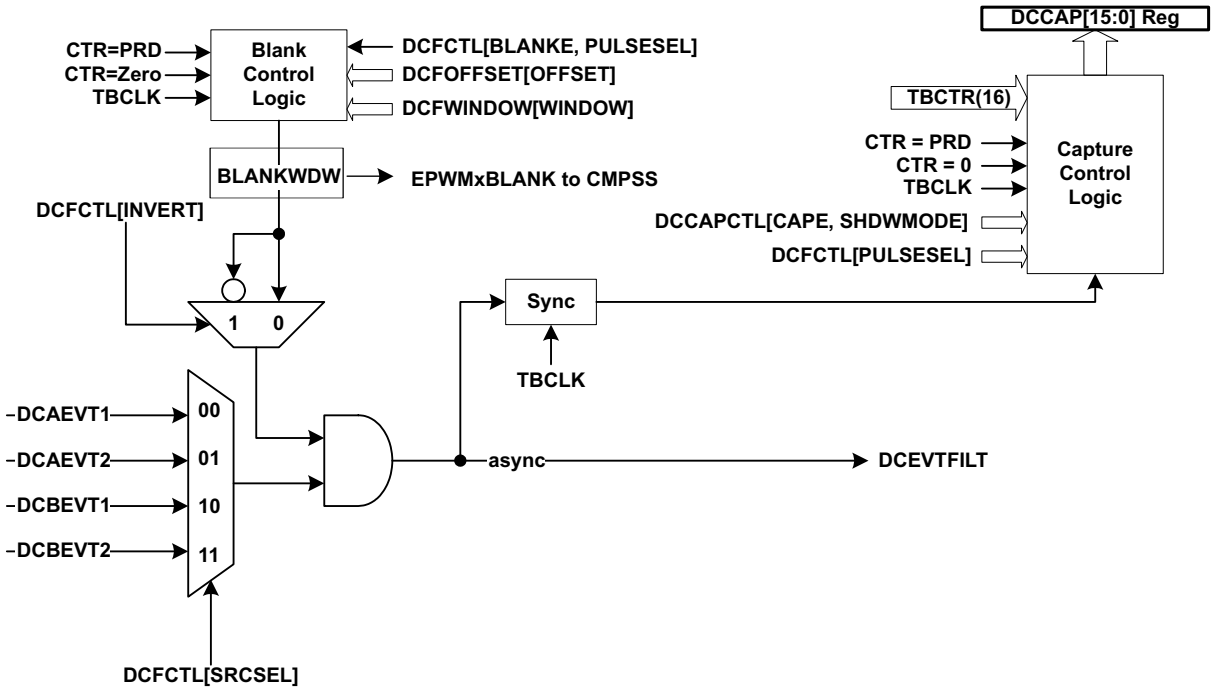


Figure 26-54. Event Filtering

Capture Control Logic: The event filtering can also capture the TBCTR value of the selected DCxEVTy event as configured in the DCCAPCTL register. When capture control logic is enabled, the selected DCxEVTy event triggers capture of the TBCTR to the active register. The CPU reads directly from the active register unless shadow mode is enabled by DCCAPCTL[SHDWMODE]. When shadow mode is enabled, the active register information is copied to shadow register on the event specified by DCFCCTL[PULSESEL], and the CPU reads from the shadow register. After the selected DCxEVTy event, no further capture events occur until the event specified by DCCAPCTL[CAPMODE]. The CAPMODE can be configured two ways: (1) no further capture events occur until the event defined by DCFCTL[PULSESEL] or (2) no further capture events occur until the compare-event flag at DCCAPCTL[CAPSTS] is cleared by DCCAPCTL[CAPCLR].

Note

You must configure the ePWM blanking window appropriately so that the Trip Input stays valid for at least 3 ePWM cycles after the blanking window has expired.

Figure 26-55 illustrates several timing conditions for the offset and blanking window within an ePWM period. Notice that if the blanking window crosses the CTR = 0 or CTR = PRD boundary, the next window still starts at the same offset value after the CTR = 0 or CTR = PRD pulse.

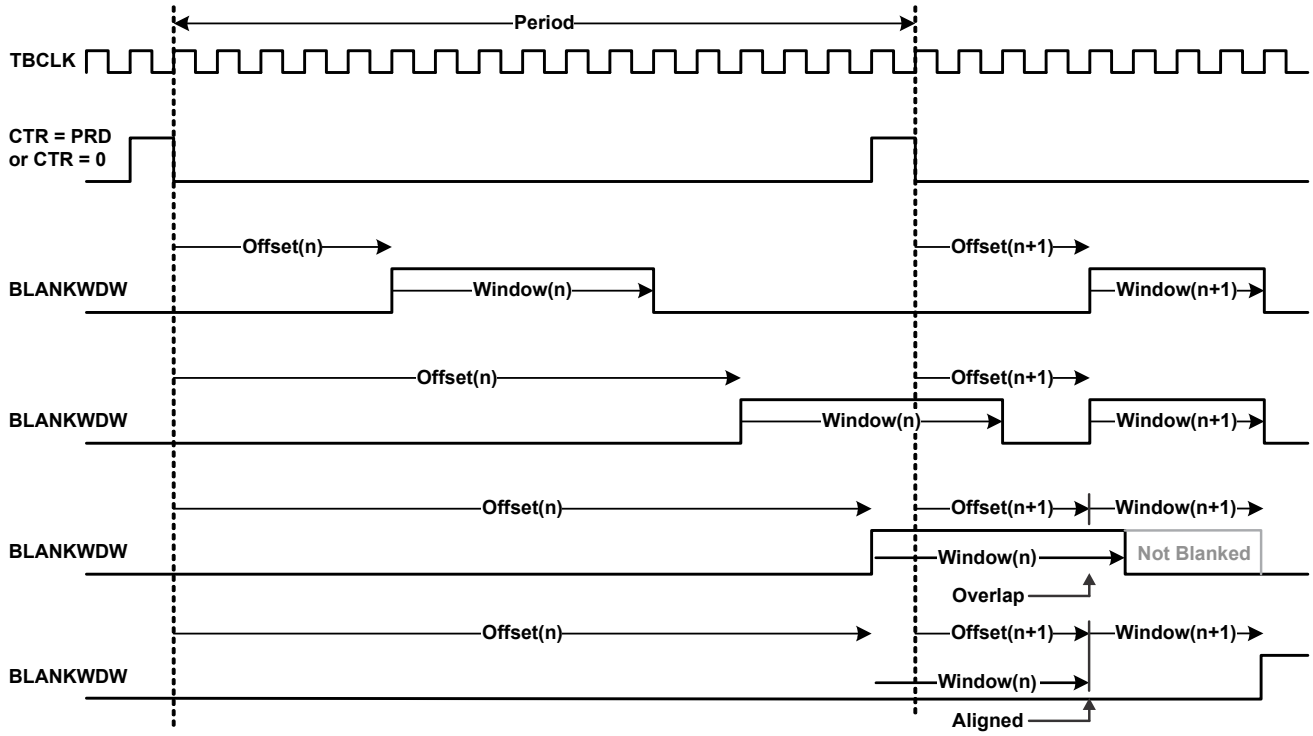


Figure 26-55. Blanking Window Timing Diagram

26.11.4.3 Valley Switching

Event filtering depicts the valley switching function along with the event filtering logic described in Section 26.11.4.2. This function can be used to achieve programmable valley switching without any additional external circuitry. This module provides an on-chip hardware mechanism that can:

- Capture the oscillation period
- Accurately delay the PWM switching instant
- Allow a programmable number of edges before the delay takes effect
- Provide multiple choices of triggers and events
- Allow easy adaptability for optimum performance under changing system/operating conditions

The DCxEVTy signal needs further processing to support valley switching. Here is a brief description of how valley switching function is enabled:

1. Select one of the DCxEVTy events as input to the valley switching block (DCFCTL[SRCSEL]) with an option to add the blanking window (Blank Control Logic). This is where the comparator output (or external input) above is selected as an input to the valley switching block.
2. Configure the edge filter to capture 'n' rising, falling or both edges through the edge selection logic (DCFCTL[EDGEMODE, EDGECOUNT]).
3. Select the correct event to reset and restart the edge filter (VCAPCTL[TRIGSEL]). Edge capturing event is triggered or armed by this selected edge.
4. Enable valley capture logic (VCAPCTL[VCAPE]).
5. Select the start edge that indicates the start of capture for oscillation period measurement (VCNTCFG[STARTEDGE]). This is where the 16-bit counter starts counting.
6. Select the stop edge (VCNTCFG[STOPEDGE]) that indicates the edge at which the 16-bit counter stops counting. The captured counter value (CNTVAL) provides oscillation period information.
   - The STOPEDGE value must always be greater than STARTEDGE value.
7. Configure and apply the captured delay (CNTVAL) to the edge filtered DCxEVTy signal. The CNTVAL value can be applied as is or applied in conjunction with a software programmed value (useful for offset adjustment) (SWVDELVAL) or only a fraction of the delay can be applied with or without SWVDELVAL. This is useful to correctly apply a delay corresponding to the valley point. (VCAPCTL[VDELAYDIV])
8. Configure VCAPCTL[EDGEFILTDLYSEL] to apply hardware delay based on the captured value above.

Once the counter is stopped, counter value is copied into CNTVAL register and counter is reset to zero. No further captures are done until the logic is triggered again by occurrence of event selected by VCAPCTL[TRIGSEL]. In this implementation, the software trigger is used as the source for VCAPCTL[TRIGSEL]. Upon occurrence of the trigger event, irrespective of the current status of the counter, the counter is reset and starts counting from zero upon occurrence of the STARTEDGE. Similarly, upon occurrence of the trigger event, the edge filter is reset and starts counting from zero upon occurrence of the STARTEDGE.

Output from the valley switching block (DCEVTFILT) is then used to synchronize the PWM time-base. The process is shown in Figure 26-56.

Note

A specific application example showcasing the usage of valley switching hardware and software is available in C2000Ware.

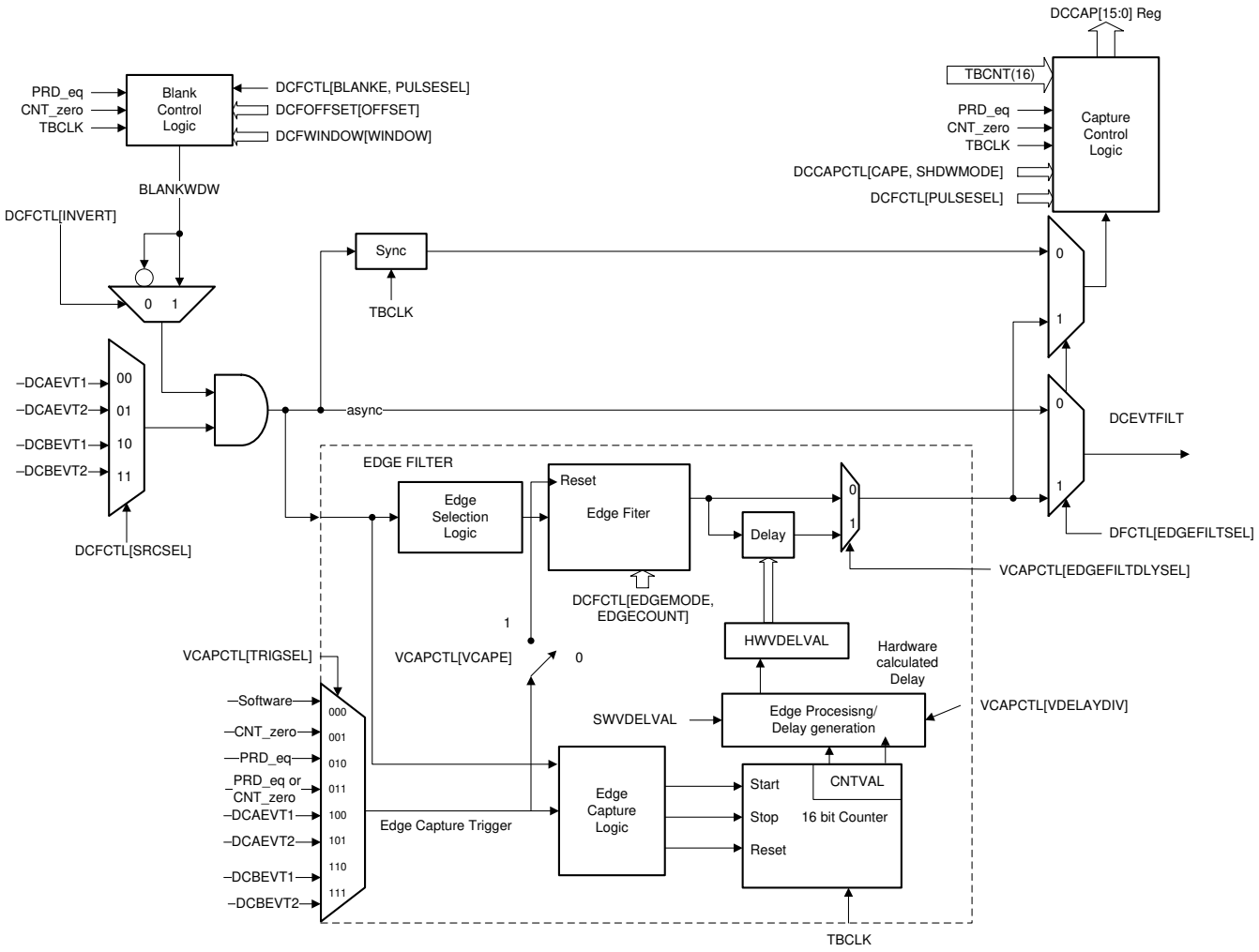


Figure 26-56. Valley Switching

26.12 ePWM Crossbar (X-BAR)

Figure 26-57 shows the architecture of the ePWM Crossbar (X-BAR). This module enables selection of various trigger sources into any of the eight dedicated EPWM trips inputs, namely the TRIP4, TRIP5, TRIP7, TRIP8, TRIP9, TRIP10, TRIP11, and TRIP12.

Note

Refer to the *Crossbar (X-BAR)* chapter for more information on the X-BAR modules, including X-BAR flags.

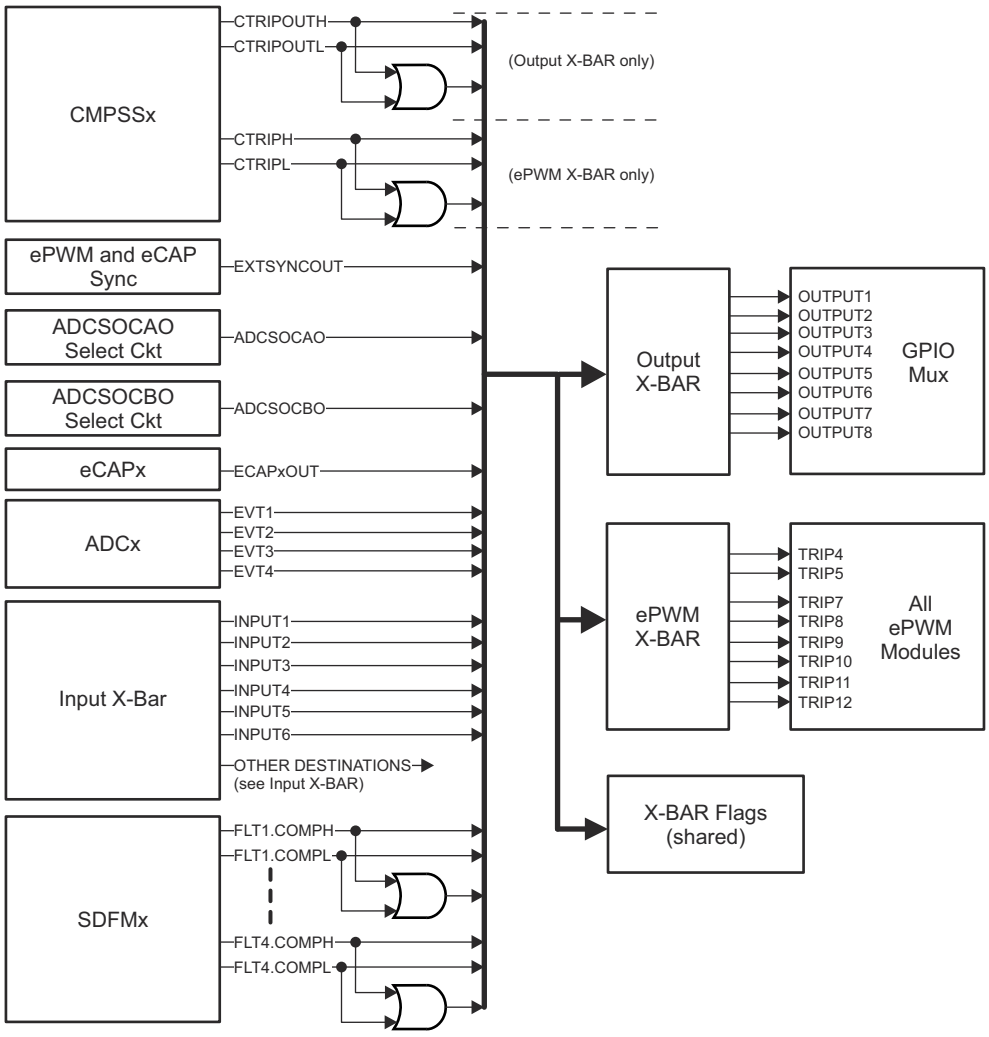


Figure 26-57. ePWM X-BAR

26.13 Applications to Power Topologies

An ePWM module has all the local resources necessary to operate completely as a standalone module or to operate in synchronization with other identical ePWM modules.

26.13.1 Overview of Multiple Modules

Previously in this chapter, all discussions have described the operation of a single module. To facilitate the understanding of multiple modules working together in a system, the ePWM module described in reference is represented by the more simplified block diagram shown in Figure 26-58. This simplified ePWM block shows only the key resources needed to explain how a multiswitch power topology is controlled with multiple ePWM modules working together.

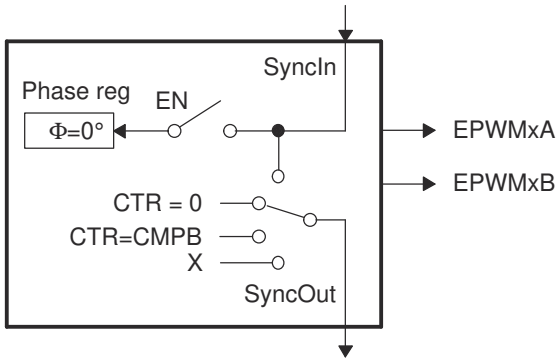


Figure 26-58. Simplified ePWM Module

26.13.2 Key Configuration Capabilities

The key configuration choices available to each module are as follows:

- Options for SyncIn
  - Load own counter with phase register on an incoming sync strobe—enable (EN) switch closed
  - Do nothing or ignore incoming sync strobe—enable switch open
  - Sync flow-through - SyncOut connected to SyncIn
  - Master mode, provides a sync at PWM boundaries—SyncOut connected to CTR = PRD
  - Master mode, provides a sync at any programmable point in time—SyncOut connected to CTR = CMPB
  - Module is in standalone mode and provides no sync to other modules—SyncOut connected to X (disabled)
- Options for SyncOut
  - Sync flow-through - SyncOut connected to SyncIn
  - Master mode, provides a sync at PWM boundaries—SyncOut connected to CTR = PRD
  - Master mode, provides a sync at any programmable point in time—SyncOut connected to CTR = CMPB
  - Module is in standalone mode and provides no sync to other modules—SyncOut connected to X (disabled)

For each choice of SyncOut, a module can also choose to load its own counter with a new phase value on a SyncIn strobe input or choose to ignore the value (that is, by the enable switch). Although various combinations are possible, the two most common—Master module and Slave module modes—are shown in Figure 26-59.

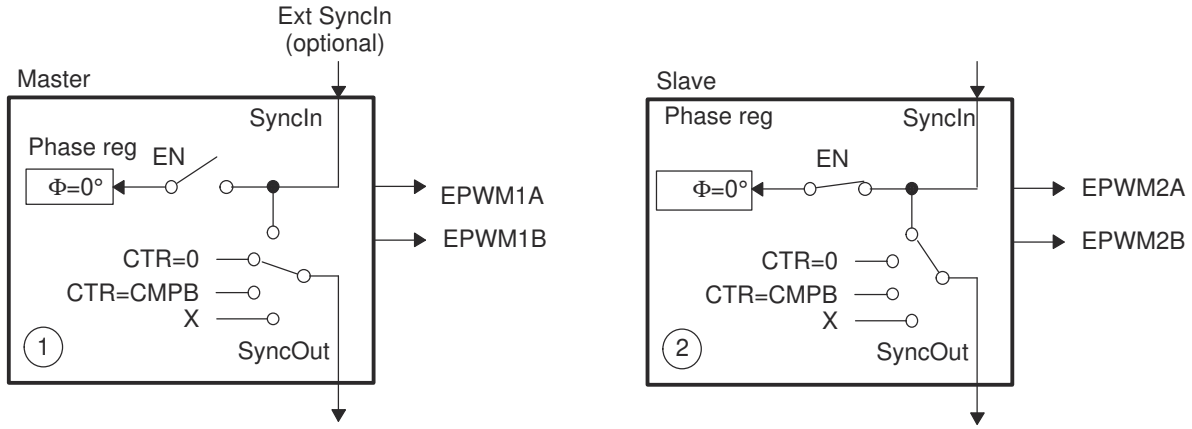
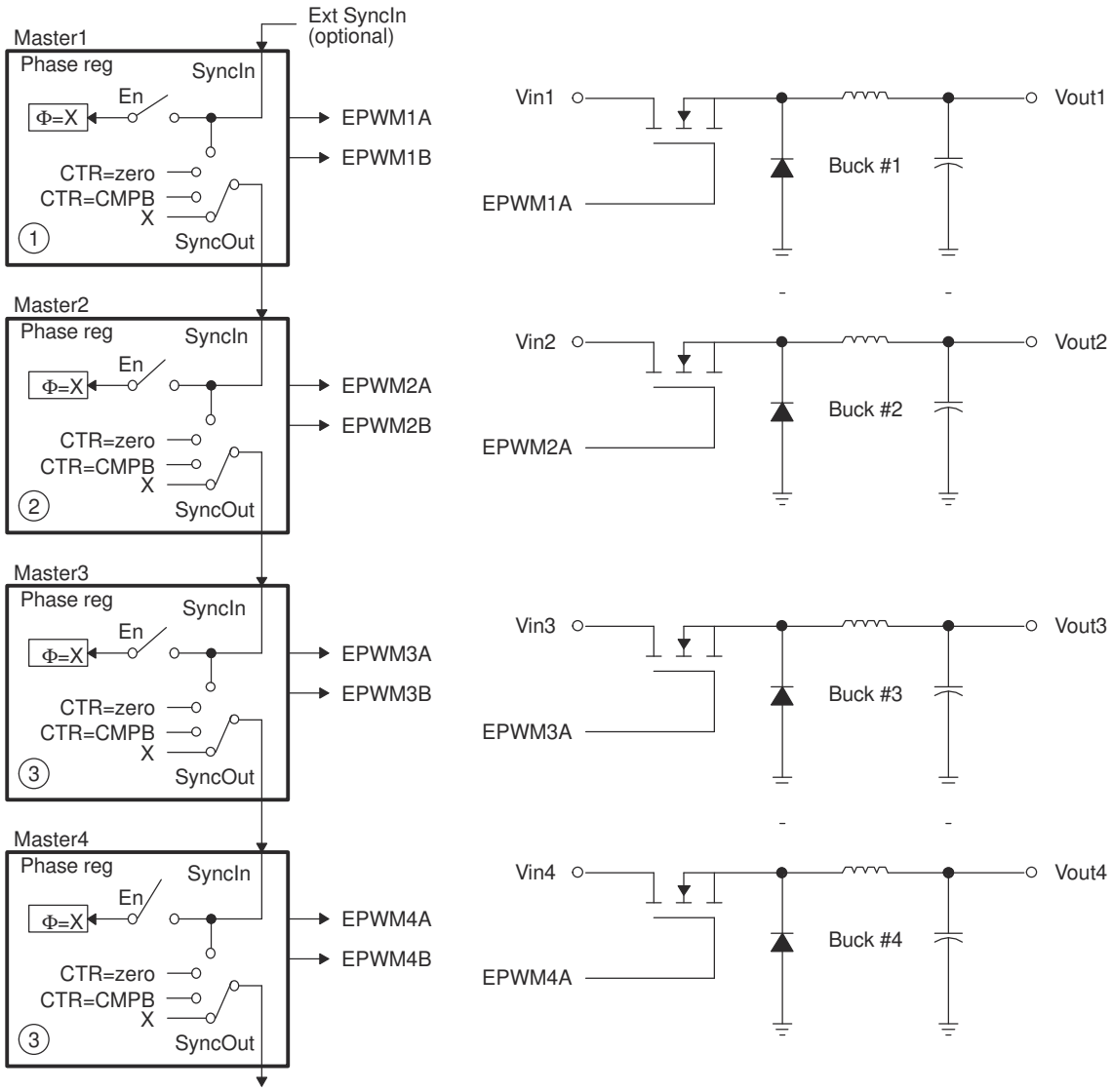


Figure 26-59. EPWM1 Configured as a Typical Master, EPWM2 Configured as a Slave

26.13.3 Controlling Multiple Buck Converters With Independent Frequencies

One of the simplest power converter topologies is the buck. A single ePWM module configured as a master can control two buck stages with the same PWM frequency. If independent frequency control is required for each buck converter, then one ePWM module must be allocated for each converter stage. Figure 26-60 shows four buck stages, each running at independent frequencies. In this case, all four ePWM modules are configured as Masters and no synchronization is used. Figure 26-61 shows the waveforms generated by the setup shown in Figure 26-60; note that only three waveforms are shown, although there are four stages.



A. φ = X indicates value in phase register is a "don't care"

Figure 26-60. Control of Four Buck Stages. Here $F_{PWM1} \neq F_{PWM2} \neq F_{PWM3} \neq F_{PWM4}$

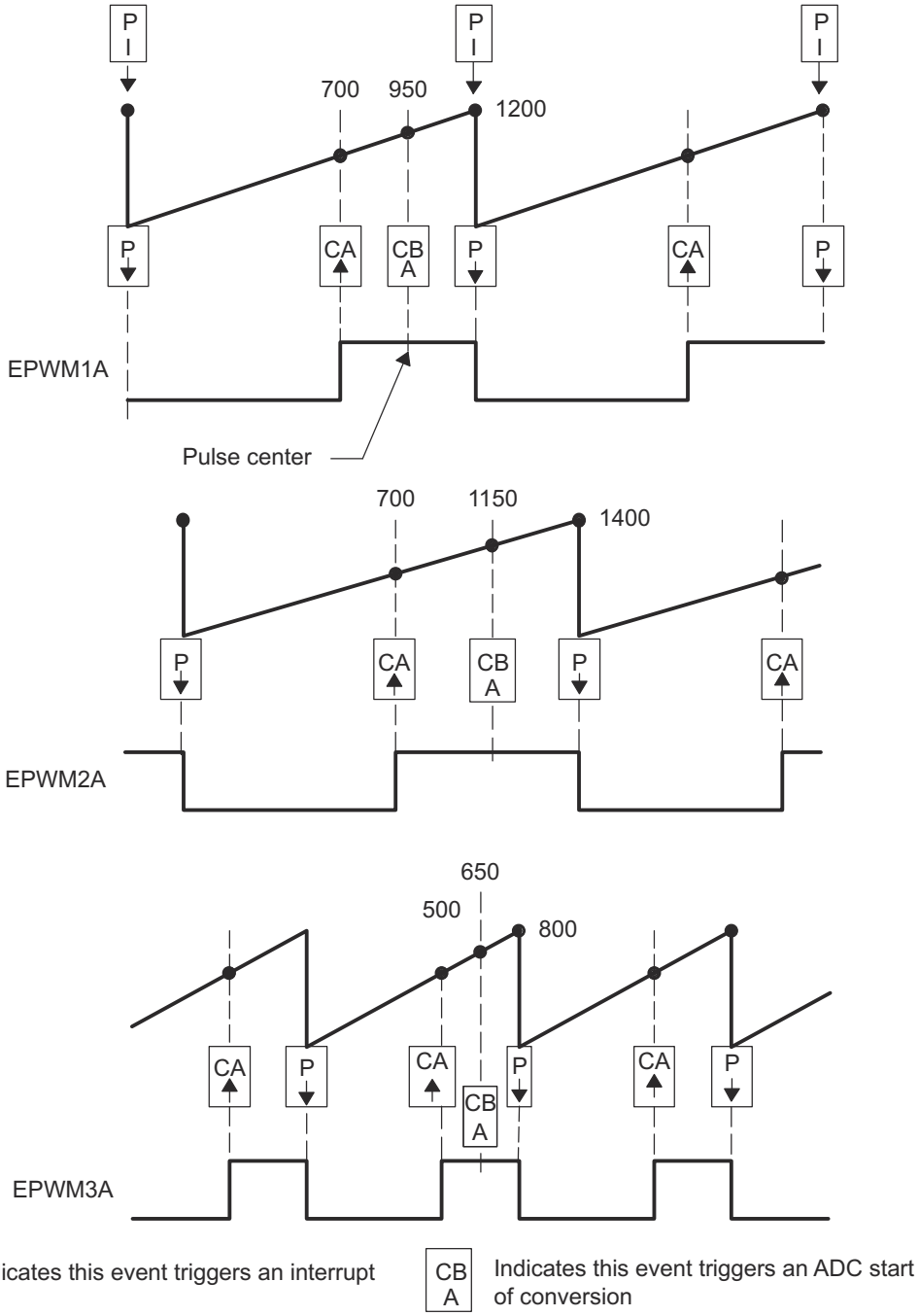
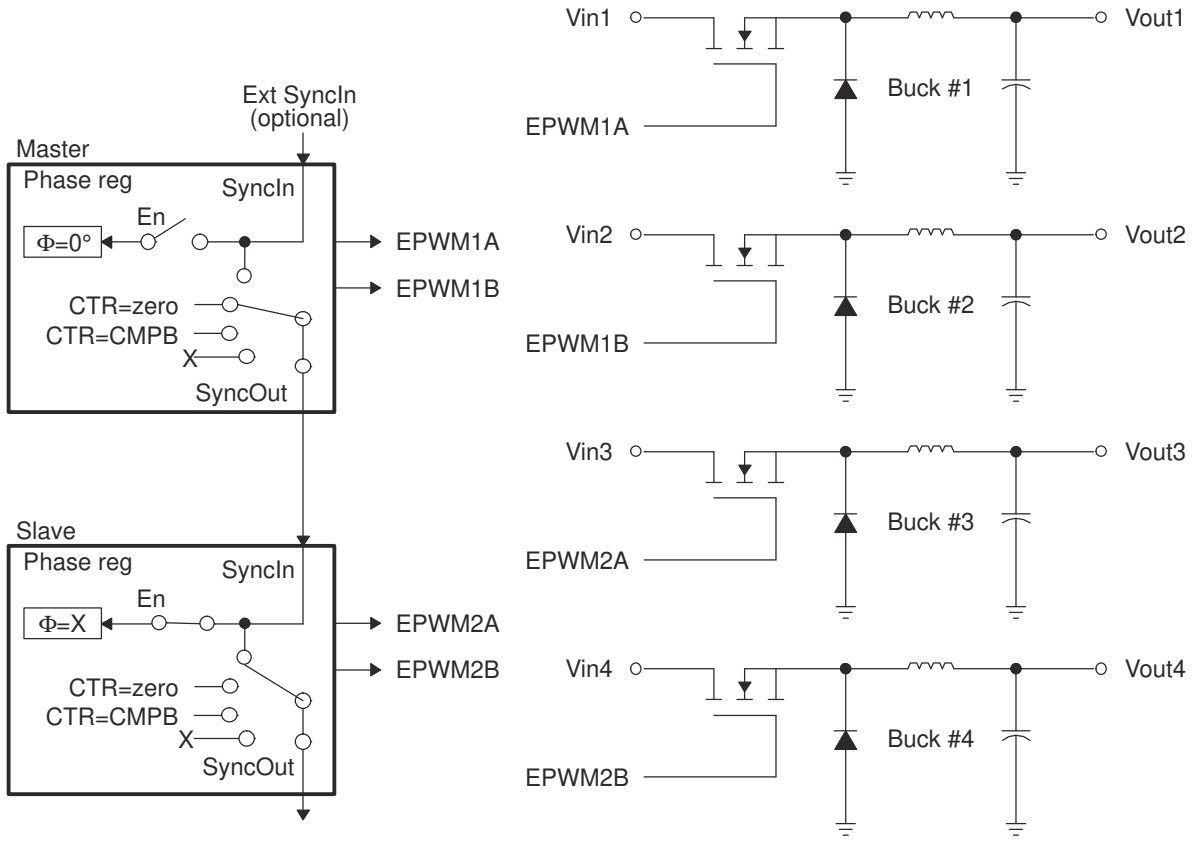


Figure 26-61. Buck Waveforms for Control of Four Buck Stages (Note: Only three bucks shown here)

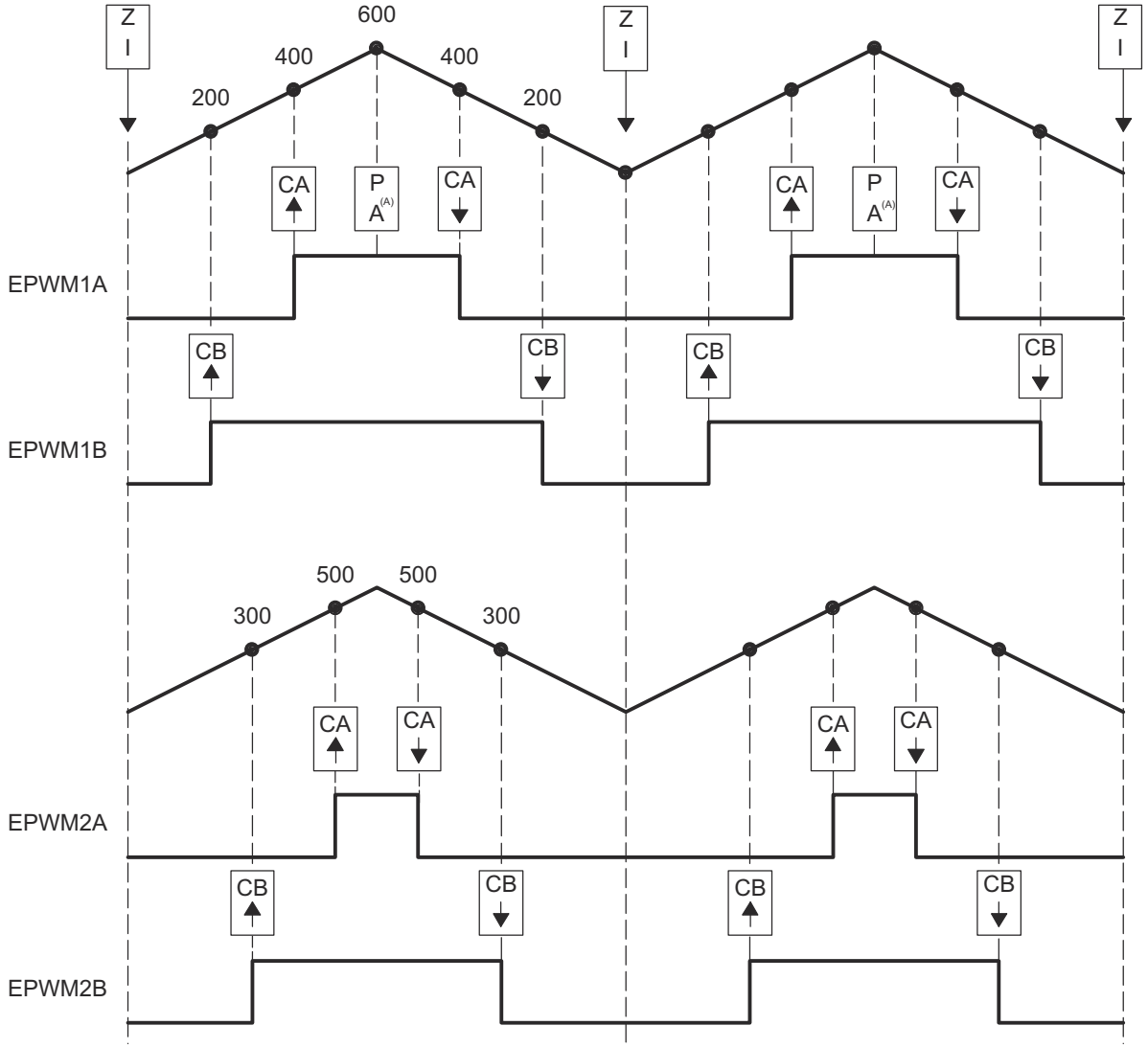
26.13.4 Controlling Multiple Buck Converters With Same Frequencies

If synchronization is a requirement, ePWM module 2 is configured as a slave and operates at integer multiple (N) frequencies of module 1. The sync signal from master to slave ensures these modules remain locked. Figure 26-62 shows such a configuration; Figure 26-63 shows the waveforms generated by the configuration.



A. φ = X indicates value in phase register is a "don't care"

Figure 26-62. Control of Four Buck Stages. (Note: F_{PWM2} = N x F_{PWM1})



A. Starts ADC conversion.

Figure 26-63. Buck Waveforms for Control of Four Buck Stages (Note: $F_{PWM2} = F_{PWM1}$)

26.13.5 Controlling Multiple Half H-Bridge (HHB) Converters

Topologies that require control of multiple switching elements can also be addressed with these same ePWM modules. It is possible to control a Half-H bridge stage with a single ePWM module. This control can be extended to multiple stages. Figure 26-64 shows control of two synchronized Half-H bridge stages where stage 2 can operate at integer multiple (N) frequencies of stage 1. Figure 26-65 shows the waveforms generated by the configuration shown in Figure 26-64.

ePWM module 2 (slave) is configured for Sync flow-through; if required, this configuration allows for a third Half-H bridge to be controlled by ePWM module 3 and also, most importantly, to remain in synchronization with master ePWM module 1.

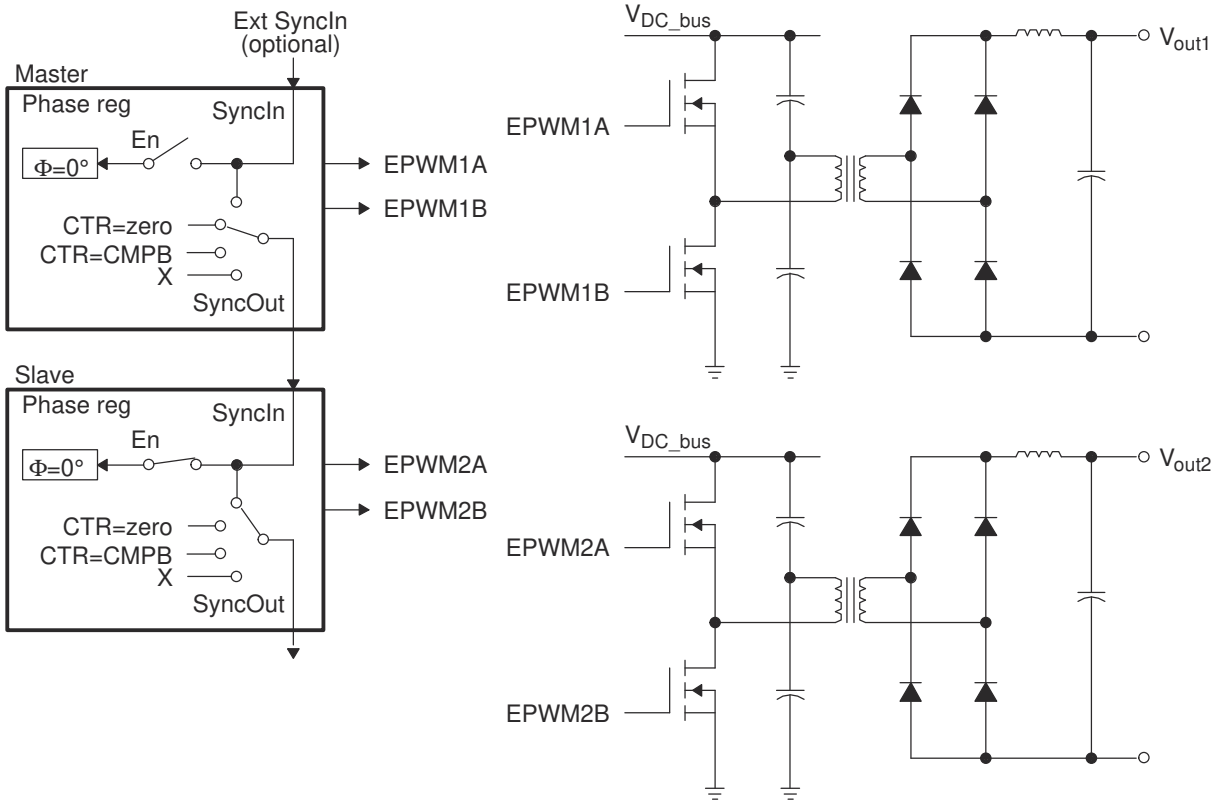


Figure 26-64. Control of Two Half-H Bridge Stages (F_{PWM2} = N x F_{PWM1})

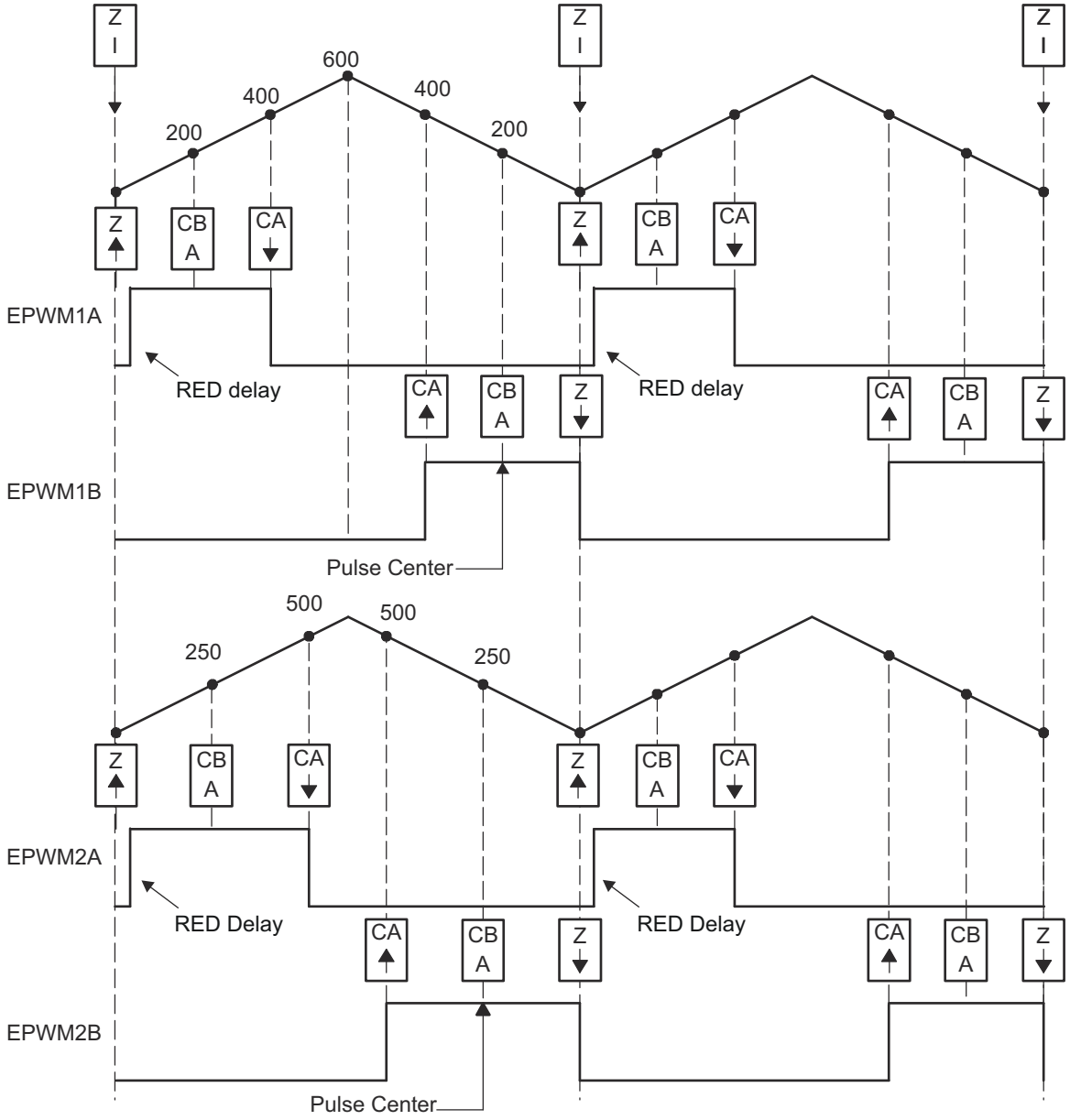


Figure 26-65. Half-H Bridge Waveforms for Control of Two Half-H Bridge Stages (Note: Here $F_{PWM2} = F_{PWM1}$)

26.13.6 Controlling Dual 3-Phase Inverters for Motors (ACI and PMSM)

The idea of multiple modules controlling a single power stage can be extended to the 3-phase inverter case. In such a case, six switching elements are controlled using three PWM modules, one for each leg of the inverter. Each leg must switch at the same frequency and all legs must be synchronized. A master slaves configuration easily addresses this requirement. Figure 26-66 shows how six PWM modules control two independent 3-phase inverters; each running a motor.

As in the cases shown in the previous sections, we have a choice of running each inverter at a different frequency (module 1 and module 4 are masters as in Figure 26-66), or both inverters can be synchronized by using one master (module 1) and five slaves. In this case, the frequency of modules 4, 5, and 6 (all equal) can be integer multiples of the frequency for modules 1, 2, and 3 (also all equal).

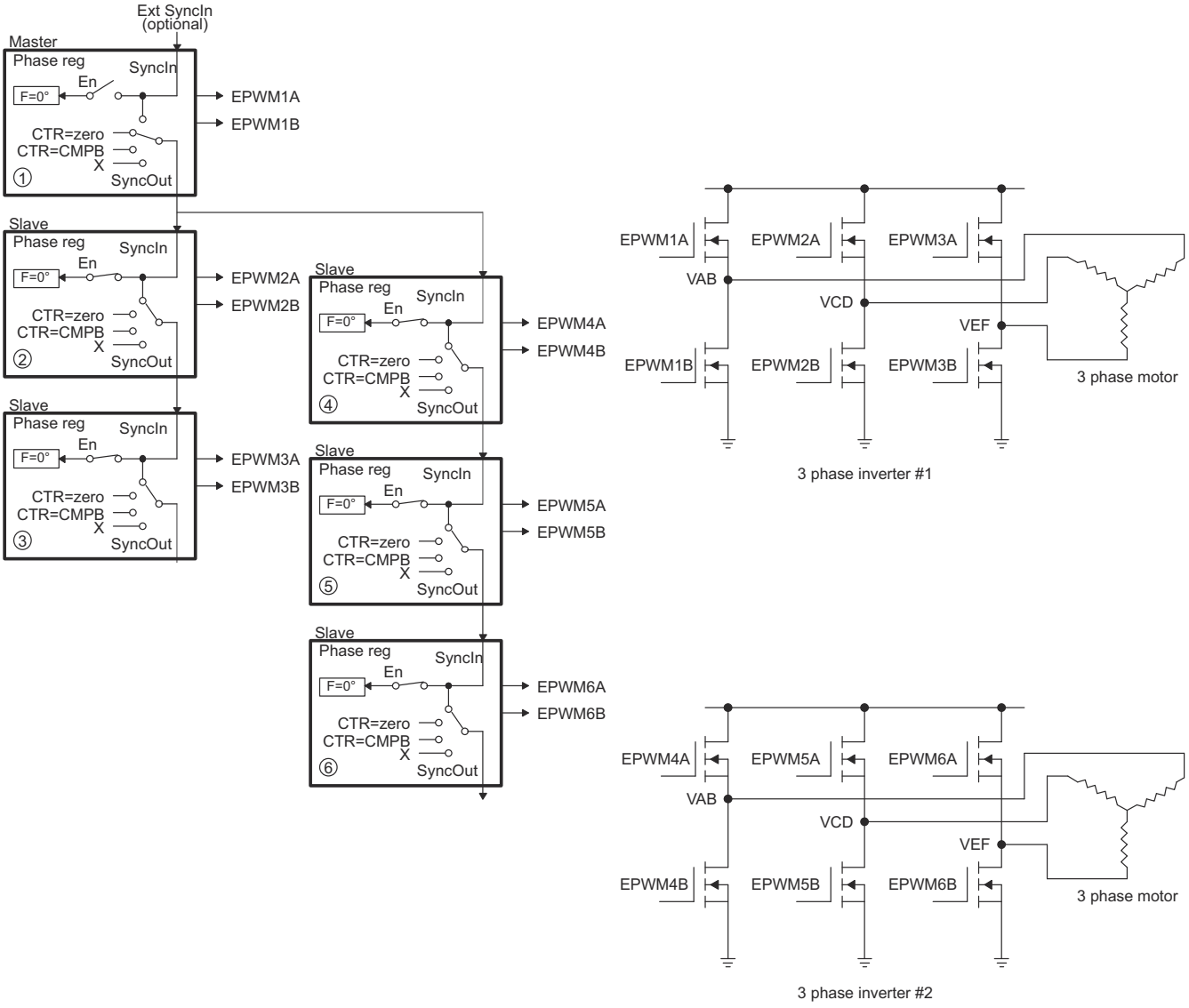


Figure 26-66. Control of Dual 3-Phase Inverter Stages as Is Commonly Used in Motor Control

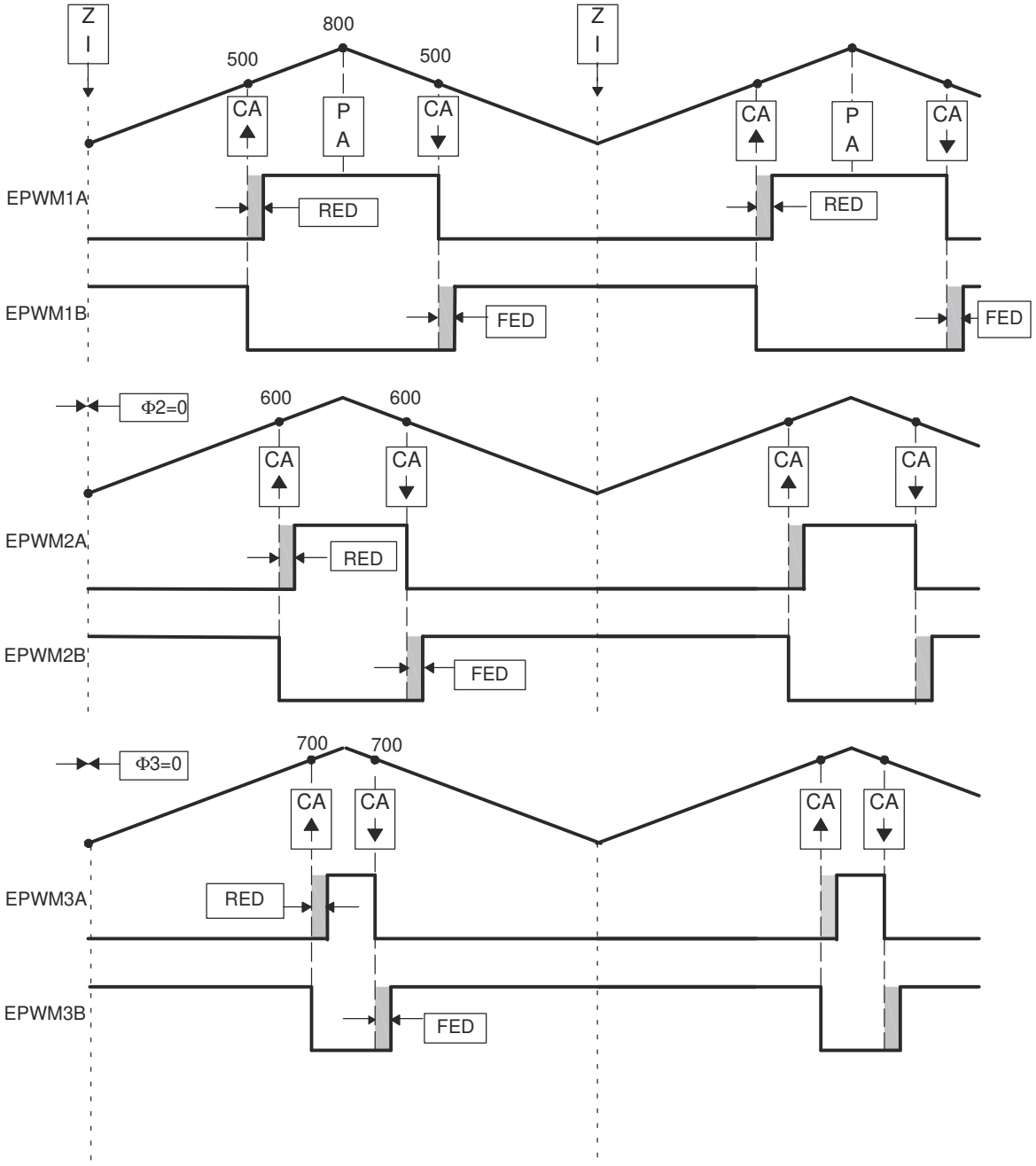


Figure 26-67. 3-Phase Inverter Waveforms for Control of Dual 3-Phase Inverter Stages (Only One Inverter Shown)

26.13.7 Practical Applications Using Phase Control Between PWM Modules

So far, none of the examples have made use of the phase register (TBPHS). It has either been set to zero or the value has been a don't care. However, by programming appropriate values into TBPHS, multiple PWM modules can address another class of power topologies that rely on phase relationship between legs (or stages) for correct operation. As described in the time-base submodule section, a PWM module can be configured to allow a SyncIn pulse to cause the TBPHS register to be loaded into the TBCTR register. To illustrate this concept, Figure 26-68 shows a master and slave module with a phase relationship of 120° (that is, the slave leads the master).

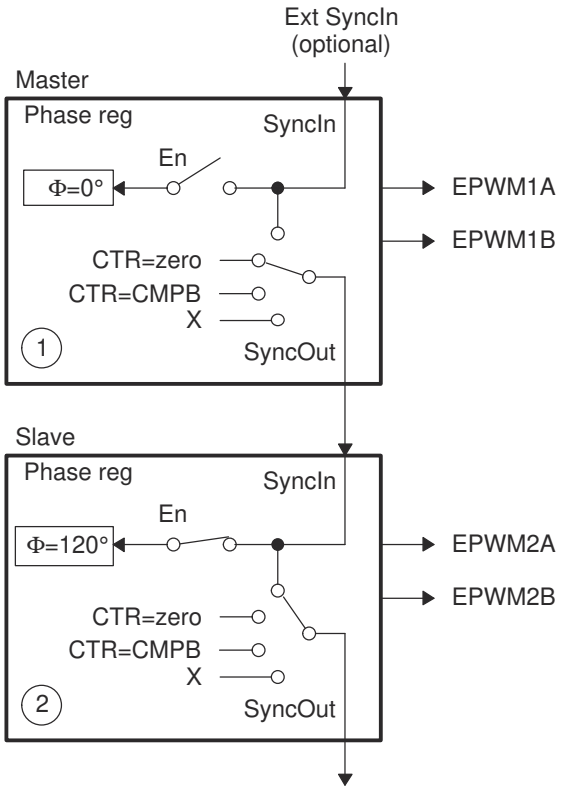


Figure 26-68. Configuring Two PWM Modules for Phase Control

Figure 26-69 shows the associated timing waveforms for this configuration. Here, TBPRD = 600 for both master and slave. For the slave, TBPHS = 200 (that is, 200/600 x 360° = 120°). Whenever the master generates a SyncIn pulse (CTR = PRD), the value of TBPHS = 200 is loaded into the slave TBCTR register so the slave time-base is always leading the master time-base by 120°.

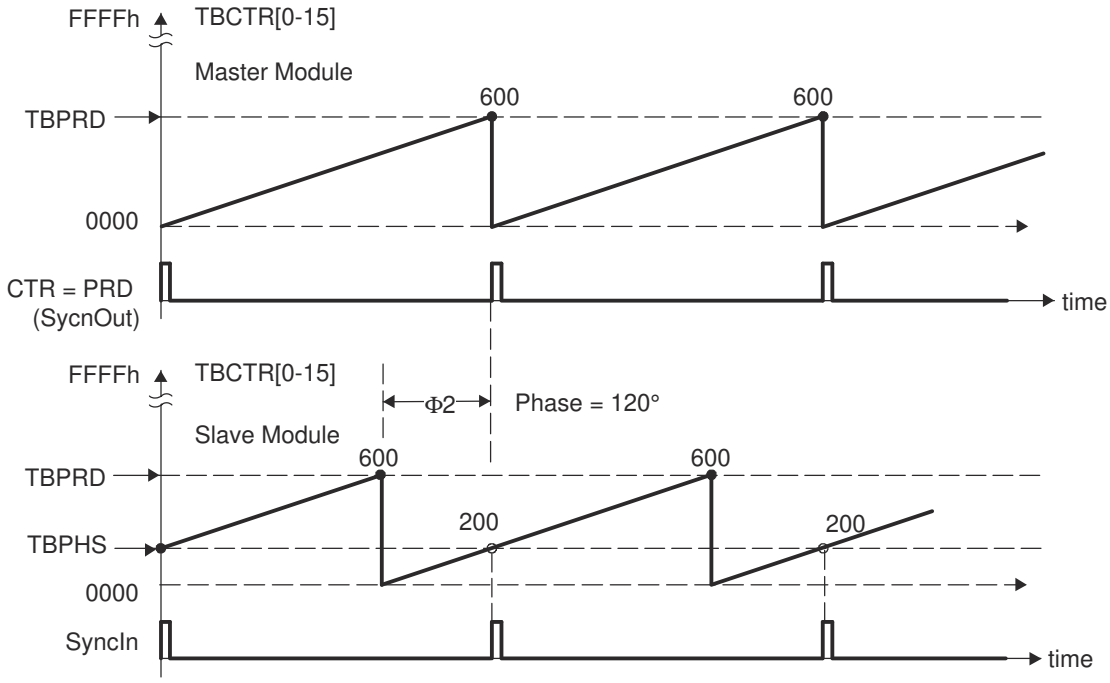


Figure 26-69. Timing Waveforms Associated with Phase Control Between Two Modules

26.13.8 Controlling a 3-Phase Interleaved DC/DC Converter

A popular power topology that makes use of phase-offset between modules is shown in Figure 26-70. This system uses three PWM modules, with module 1 configured as the master. To work, the phase relationship between adjacent modules must be F = 120°. This is achieved by setting the slave TBPHS registers 2 and 3 with values of 1/3 and 2/3 of the period value, respectively. For example, if the period register is loaded with a value of 600 counts, then TBPHS (slave 2) = 200 and TBPHS (slave 3) = 400. Both slave modules are synchronized to the master module 1.

This concept can be extended to four or more phases, by setting the TBPHS values appropriately. The following formula gives the TBPHS values for N phases:

TBPHS(N,M) = (TBPRD/N) x (M-1)

Where:

N = number of phases

M = PWM module number

For example, for the 3-phase case (N=3), TBPRD = 600,

TBPHS(3,2) = (600/3) x (2-1) = 200 (that is, Phase value for Slave module 2)

TBPHS(3,3) = 400 (that is, Phase value for Slave module 3)

Figure 26-71 shows the waveforms for the configuration in Figure 26-70.

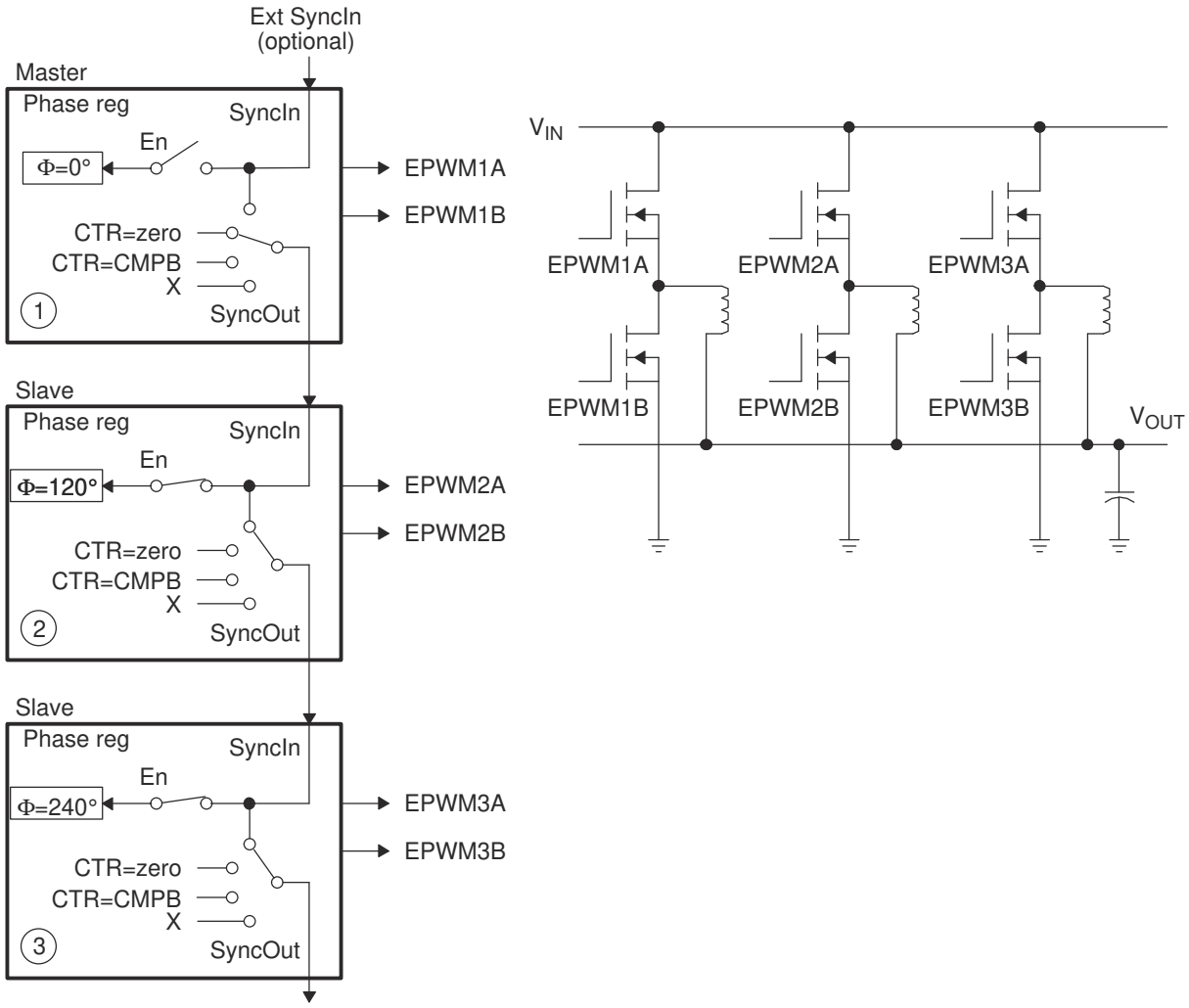


Figure 26-70. Control of 3-Phase Interleaved DC/DC Converter

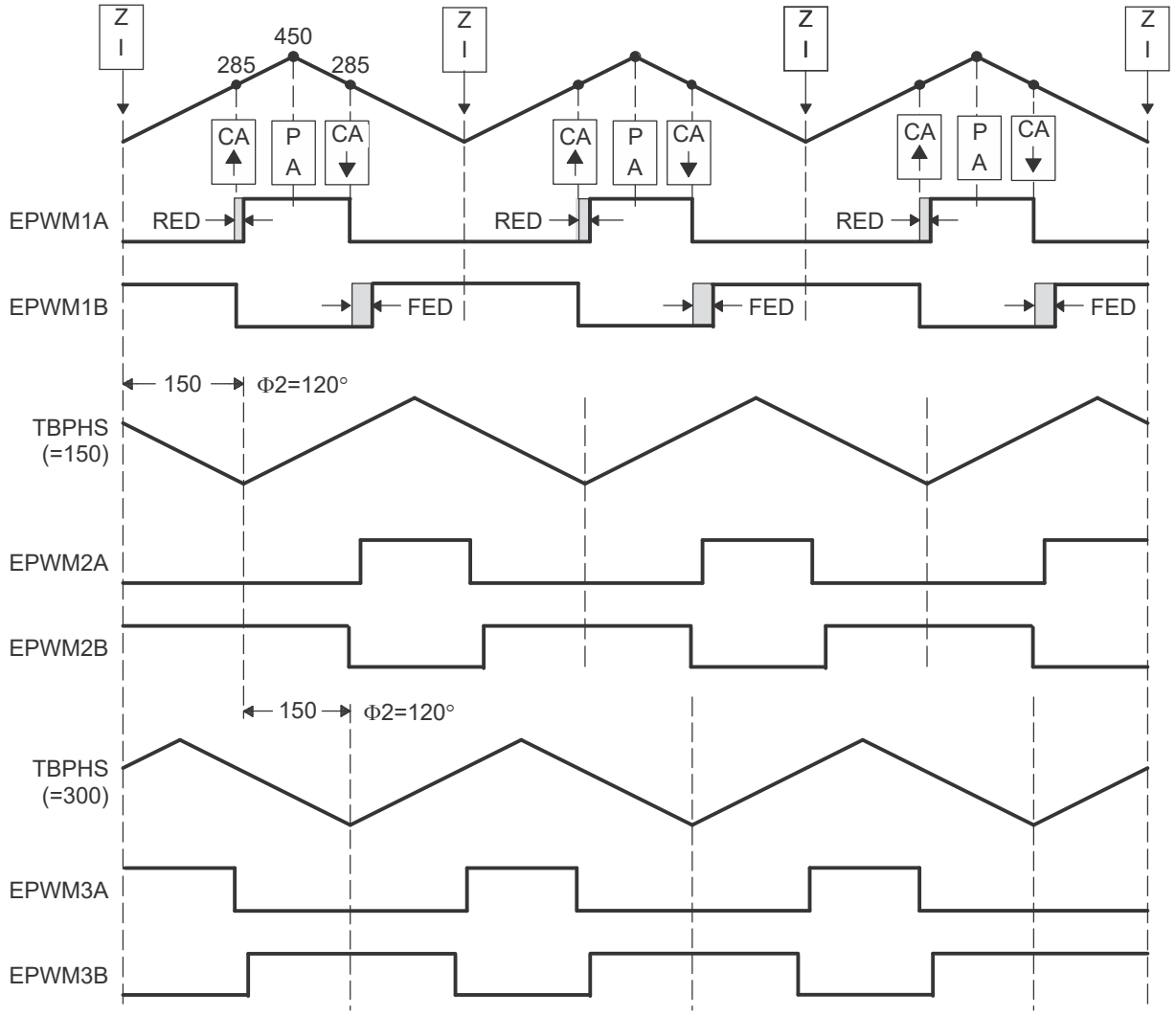


Figure 26-71. 3-Phase Interleaved DC/DC Converter Waveforms for Control of 3-Phase Interleaved DC/DC Converter

26.13.9 Controlling Zero Voltage Switched Full Bridge (ZVSFB) Converter

The example given in Figure 26-72 assumes a static or constant phase relationship between legs (modules). In such a case, control is achieved by modulating the duty cycle. It is also possible to dynamically change the phase value on a cycle-by-cycle basis. This feature lends itself to controlling a class of power topologies known as *phase-shifted full bridge*, or *zero voltage switched full bridge.* Here the controlled parameter is not duty cycle (this is kept constant at approximately 50 percent); instead it is the phase relationship between legs. Such a system can be implemented by allocating the resources of two PWM modules to control a single power stage, which in turn requires control of four switching elements. Figure 26-73 shows a master and slave module combination synchronized together to control a full H-bridge. In this case, both master and slave modules are required to switch at the same PWM frequency. The phase is controlled by using the slave phase register (TBPHS). The master phase register is not used and therefore can be initialized to zero.

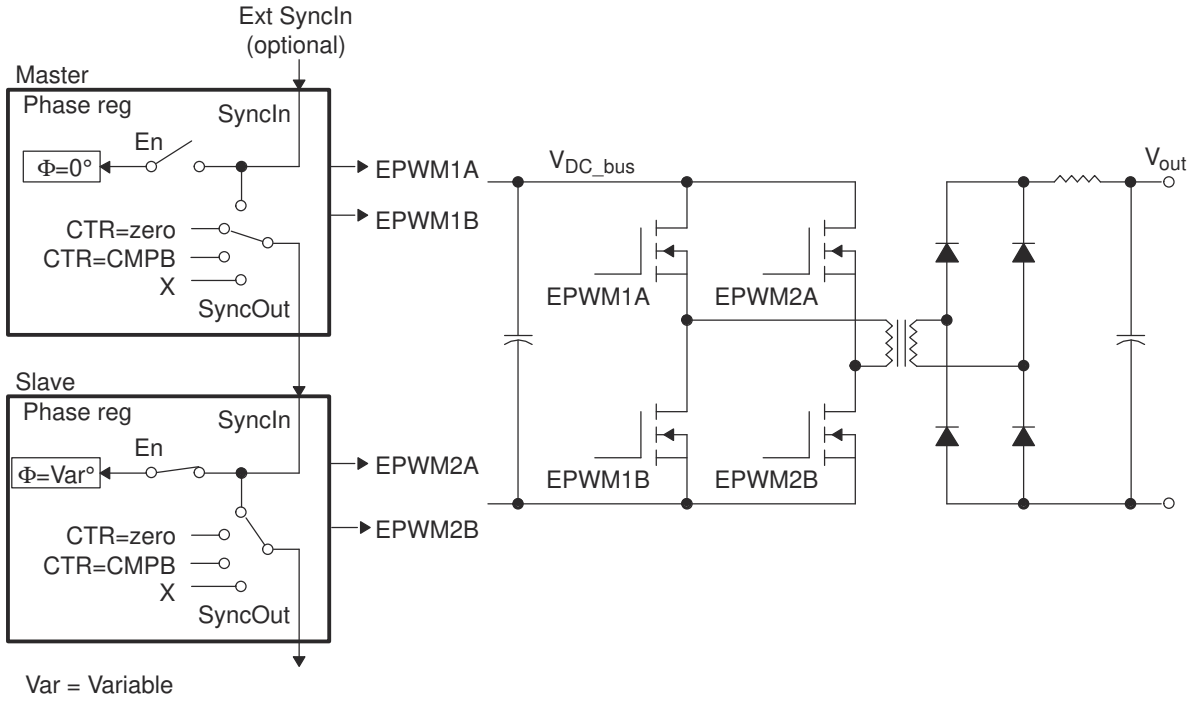


Figure 26-72. Control of Full-H Bridge Stage (F_{PWM2} = F_{PWM1})

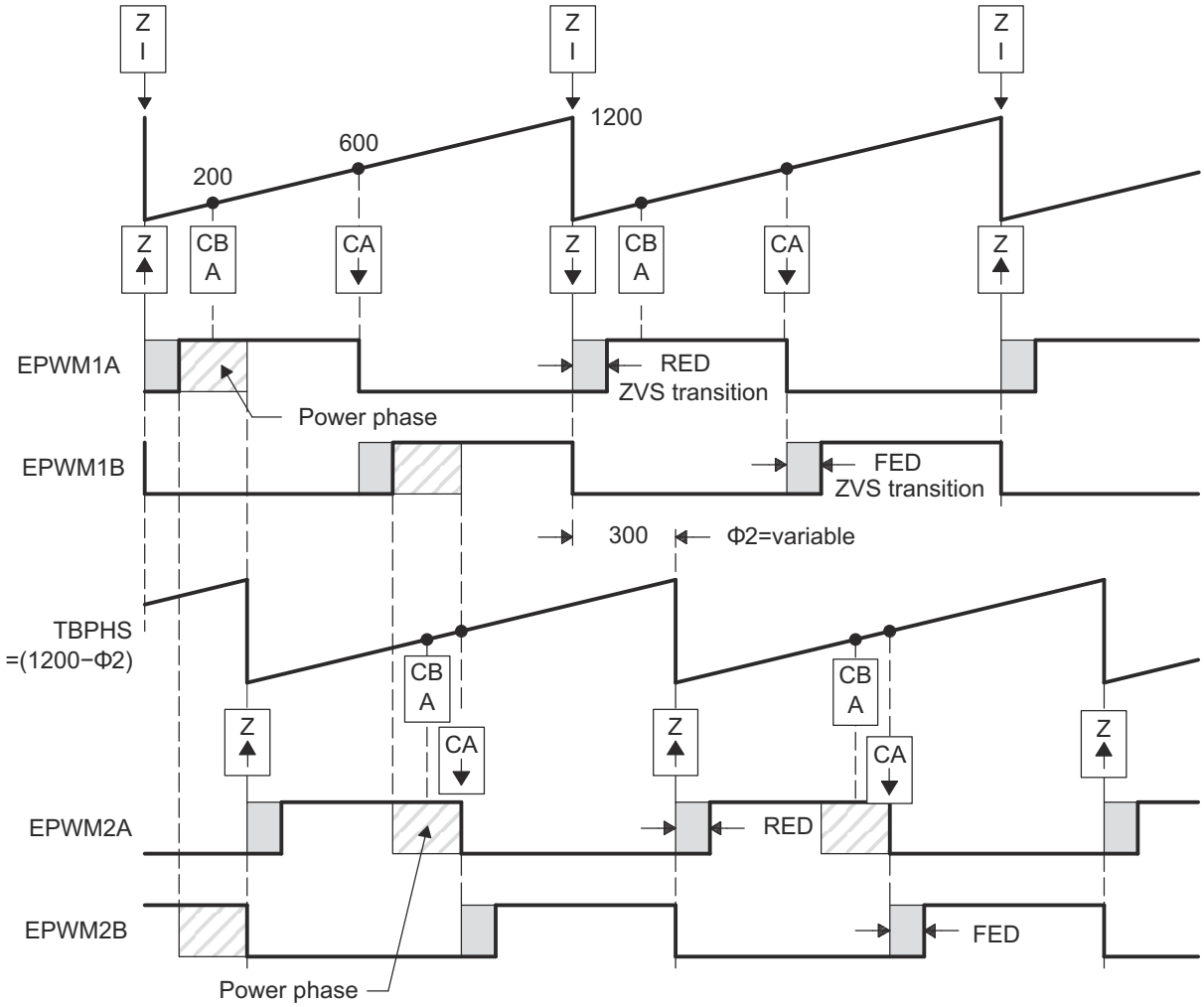


Figure 26-73. ZVS Full-H Bridge Waveforms

26.13.10 Controlling a Peak Current Mode Controlled Buck Module

Peak current control techniques offer a number of benefits like automatic over current limiting, fast correction for input voltage variations and reducing magnetic saturation. Figure 26-74 shows the use of ePWM1A along with the on-chip analog comparator for buck converter topology. The output current is sensed through a current sense resistor and fed to the positive terminal of the on-chip comparator. The internal programmable 12-bit DAC can be used to provide a reference peak current at the negative terminal of the comparator. Alternatively, an external reference could be connected at this input. The comparator output is an input to the Digital compare sub-module. The ePWM module is configured in such a way so as to trip the ePWM1A output as soon as the sensed current reaches the peak reference value. A cycle-by-cycle trip mechanism is used. Figure 26-75 shows the waveforms generated by the configuration.

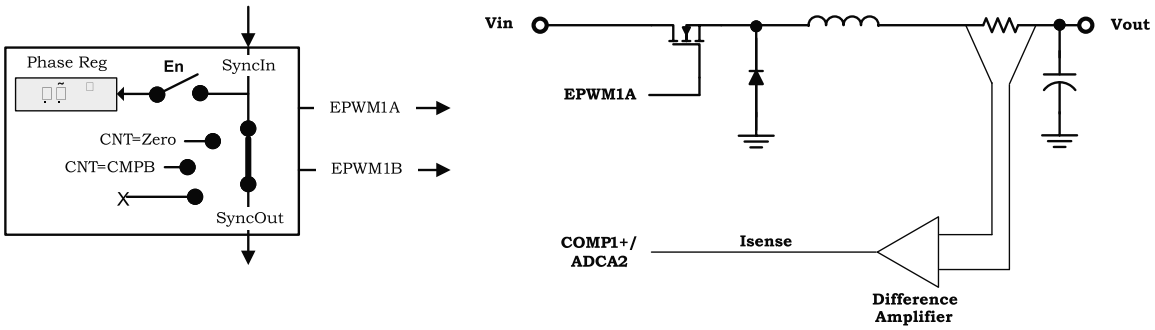


Figure 26-74. Peak Current Mode Control of Buck Converter

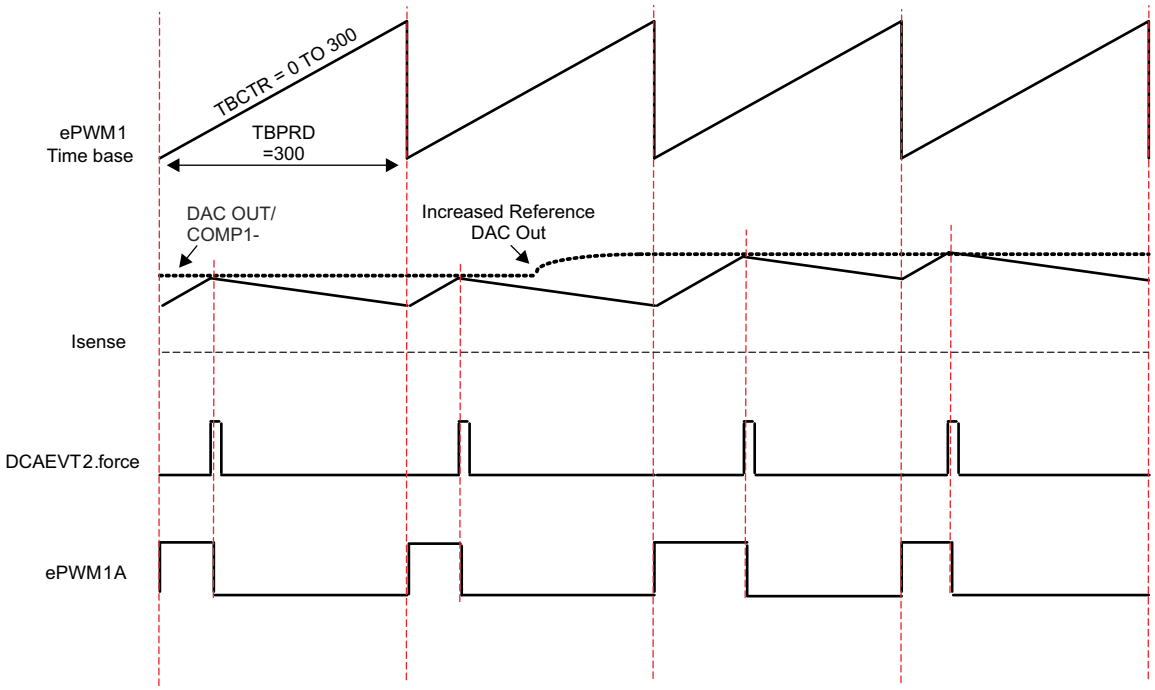
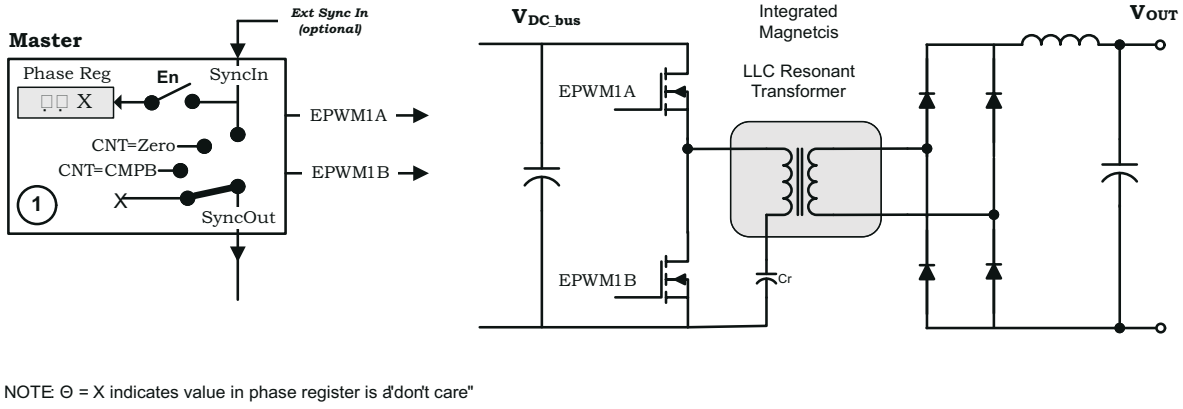


Figure 26-75. Peak Current Mode Control Waveforms for Control of Buck Converter

26.13.11 Controlling H-Bridge LLC Resonant Converter

Various topologies of resonant converters are well-known in the field of power electronics for many years. In addition to these, H-bridge LLC resonant converter topology has recently gained popularity in many consumer electronics applications where high efficiency and power density are required. In this example, single channel configuration of ePWM1 is detailed, yet the configuration can easily be extended to multichannel. Here the controlled parameter is not duty cycle (this is kept constant at approximately 50 percent); instead the parameter is frequency. Although the deadband is not controlled and kept constant as 300 ns (that is, 30 at 100 MHz TBCLK), the user can update the deadband in real time to enhance the efficiency by adjusting enough time delay for soft switching.



NOTE: Θ = X indicates value in phase register is a "don't care"

Figure 26-76. Control of Two Resonant Converter Stages

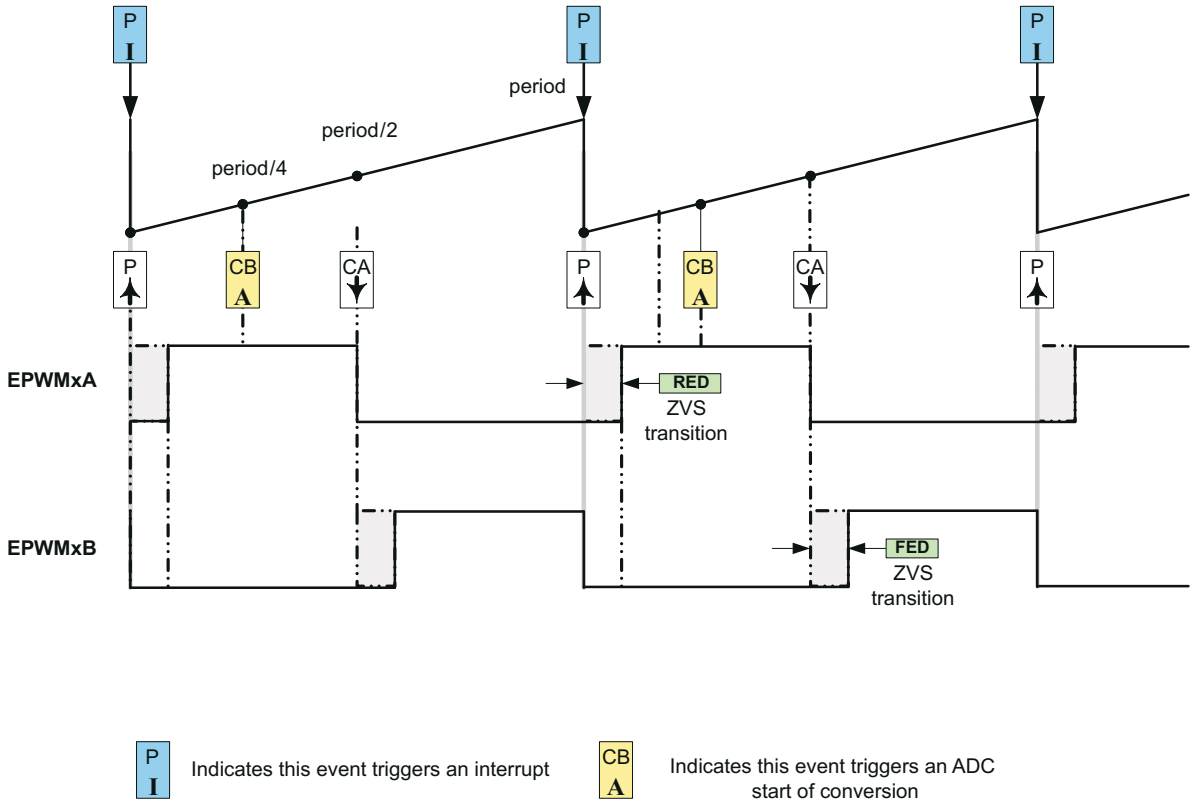


Figure 26-77. H-Bridge LLC Resonant Converter PWM Waveforms

26.14 Register Lock Protection

The register lock protection mechanism is added to protect the critical ePWM registers from being corrupted by accidental writes in case of runaway code. The register EPWMLOCK contains the definition of Lock bits (Table 26-14 shows the lock bits and the corresponding registers). This register also has a KEY field; writes to this register succeed only if the KEY field is written with a value of 0xa5a5. Refer to the register descriptions for more details.

Table 26-14. Lock Bits and Corresponding Registers

Bit Field	Definition	Registers Locked
HRLOCK	HRPWM Register Set Lock	HRCNFG, HRPWR, HRMSTEP, HRPCTL
GLLOCK	Global Load Register Set Lock	GLDCTL, GLDCFG
TZCFGLOCK	TripZone Register Set Lock	TZSEL, TZDCSEL, TZCTL, TZCTL2, TZCTLDCA, TZCTLDCB, TZEINT
TZCLRLOCK	TripZone Clear Register Set Lock	TZCLR, TZCBCCLR, TZOSTCLR, TZFRC
DCLOCK	Digital Compare Register Set Lock	DCTRIPSEL, DCACTL, DCBCTL, DCFCTL, DCCAPCTL, DCAHTRIPSEL, DCALTRIPSEL, DCBHTRIPSEL, DCBLTRIPSEL

Note

Due to the presence of the KEY field in the same register, only 32-bit writes succeed if the KEY matches. The 16-bit writes to the upper or lower half of this register are ignored.

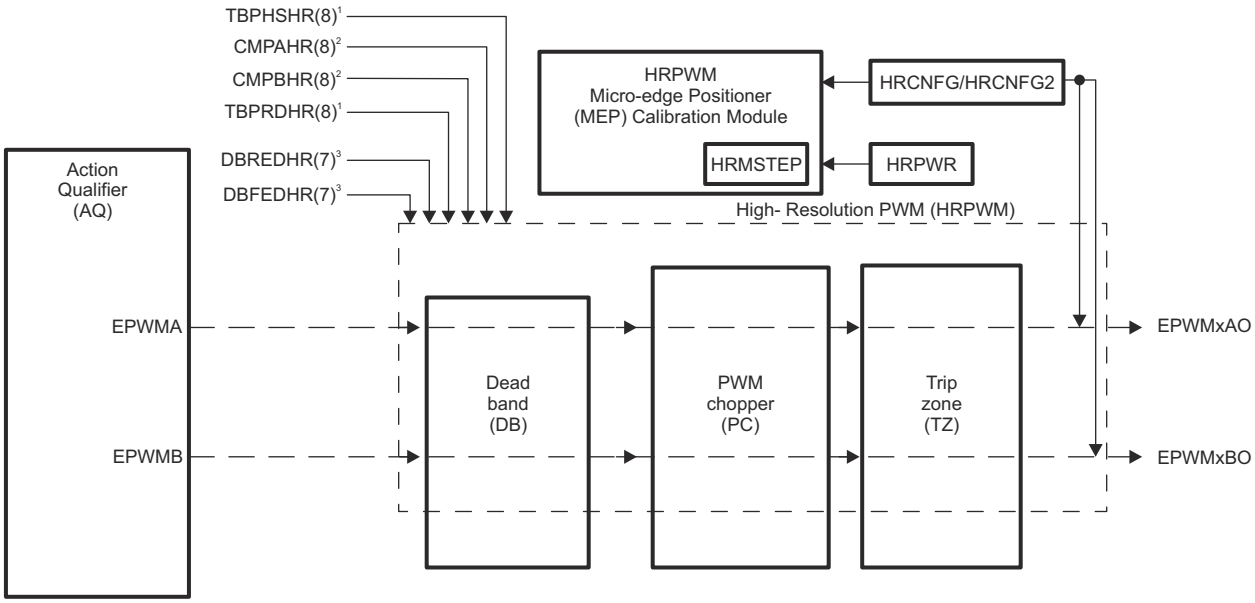
26.15 High-Resolution Pulse Width Modulator (HRPWM)

Figure 26-78 shows a block diagram of the HRPWM. This module extends the time resolution capabilities of the conventionally derived digital pulse width modulator (PWM). HRPWM is typically used when PWM resolution falls below approximately 9-10 bits. The key features of HRPWM are:

- Extended time resolution capability
- Used in both duty cycle and phase-shift control methods
- Finer time granularity control or edge positioning using extensions to the Compare A, Compare B and Phase registers
- Implemented using the A and B signal path of PWM, that is, on the EPWMxA and EPWMxB output
- Dead band high-resolution control for falling and rising edge delay in half cycle clocking operation
- Self-check diagnostics software mode to check if the micro edge positioner (MEP) logic is running how designed
- Enables high-resolution output swapping on the EPWMxA and EPWMxB output
- Enables high-resolution output on EPWMxB signal output using inversion of EPWMxA signal output
- Enables high-resolution period, duty and phase control on the EPWMxA and EPWMxB output on devices with an ePWM module

Note

See the device data sheet to determine if your device has an ePWM module with high-resolution period support.



A. From ePWM Time-base (TB) submodule
B. From ePWM counter-compare (CC) submodule
C. From ePWM Deadband (DB) submodule

Figure 26-78. HRPWM Block Diagram

The ePWM peripheral is used to perform a function mathematically equivalent to a digital-to-analog converter (DAC). As shown in Figure 26-79, the effective resolution for conventionally generated PWM is a function of PWM frequency (or period) and system clock frequency.

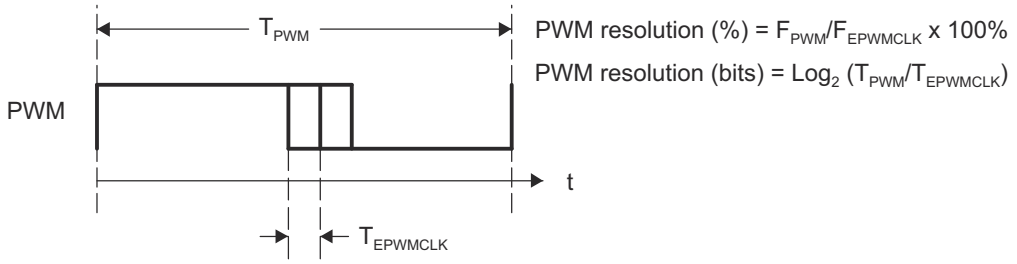


Figure 26-79. Resolution Calculations for Conventionally Generated PWM

If the required PWM operating frequency does not offer sufficient resolution in PWM mode, consider using HRPWM. As an example of improved performance offered by HRPWM, Table 26-15 shows resolution in bits for various PWM frequencies. These values assume a MEP step size of 180 ps. See the device data manual for typical and maximum performance specifications for the MEP.

Table 26-15. Resolution for PWM and HRPWM

PWM Frequency (kHz)	Regular Resolution (PWM) 100 MHz EPWMCLK		High Resolution (HRPWM)	
	Bits	%	Bits	%
20	12.3	0.02	18.1	0.000
50	11	0.05	16.8	0.001
100	10	0.1	15.8	0.002
150	9.4	0.15	15.2	0.003
200	9	0.2	14.8	0.004
250	8.6	0.25	14.4	0.005
500	7.6	0.5	13.4	0.009
1000	6.6	1	12.4	0.018
1500	6.1	1.5	11.9	0.027
2000	5.6	2	11.4	0.036

Although each application can differ, typical low-frequency PWM operation (below 250 kHz) does not require HRPWM. HRPWM capability is most useful for high-frequency PWM requirements of power conversion topologies such as:

- Single-phase buck, boost, and flyback
- Multiphase buck, boost, and flyback
- Phase-shifted full bridge
- Direct modulation of D-Class power amplifiers

26.15.1 Operational Description of HRPWM

The HRPWM is based on micro-edge positioner (MEP) technology. MEP logic is capable of positioning an edge very finely by sub-dividing one coarse system clock of a conventional PWM generator. The time step accuracy is on the order of 150 ps. See the device data manual for the typical MEP step size on a particular device. The HRPWM also has a self-check software diagnostics mode to check if the MEP logic is running as designed under all operating conditions. Details on software diagnostics and functions are in Section 26.15.1.7.

Figure 26-80 shows the relationship between one coarse system clock and edge position in terms of MEP steps, which are controlled using an 8-bit field in the Compare A extension register (CMPAHR). The same operating logic applies to CMPBHR as well.

To generate an HRPWM waveform, configure the ePWM registers to generate a conventional PWM of a given frequency and polarity. The HRPWM works together with the ePWM registers to extend edge resolution. Although many programming combinations are possible, only a few are needed and practical. These methods are described in Section 26.15.1.8.

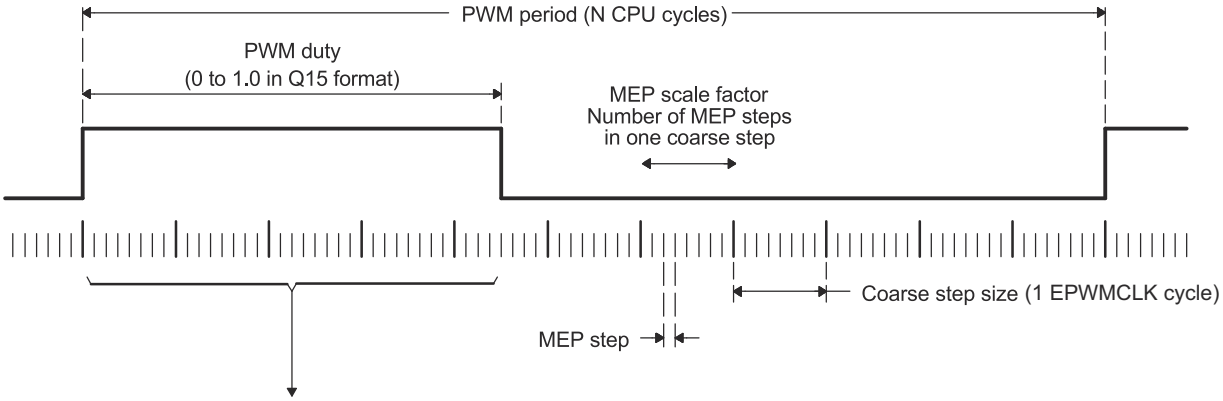
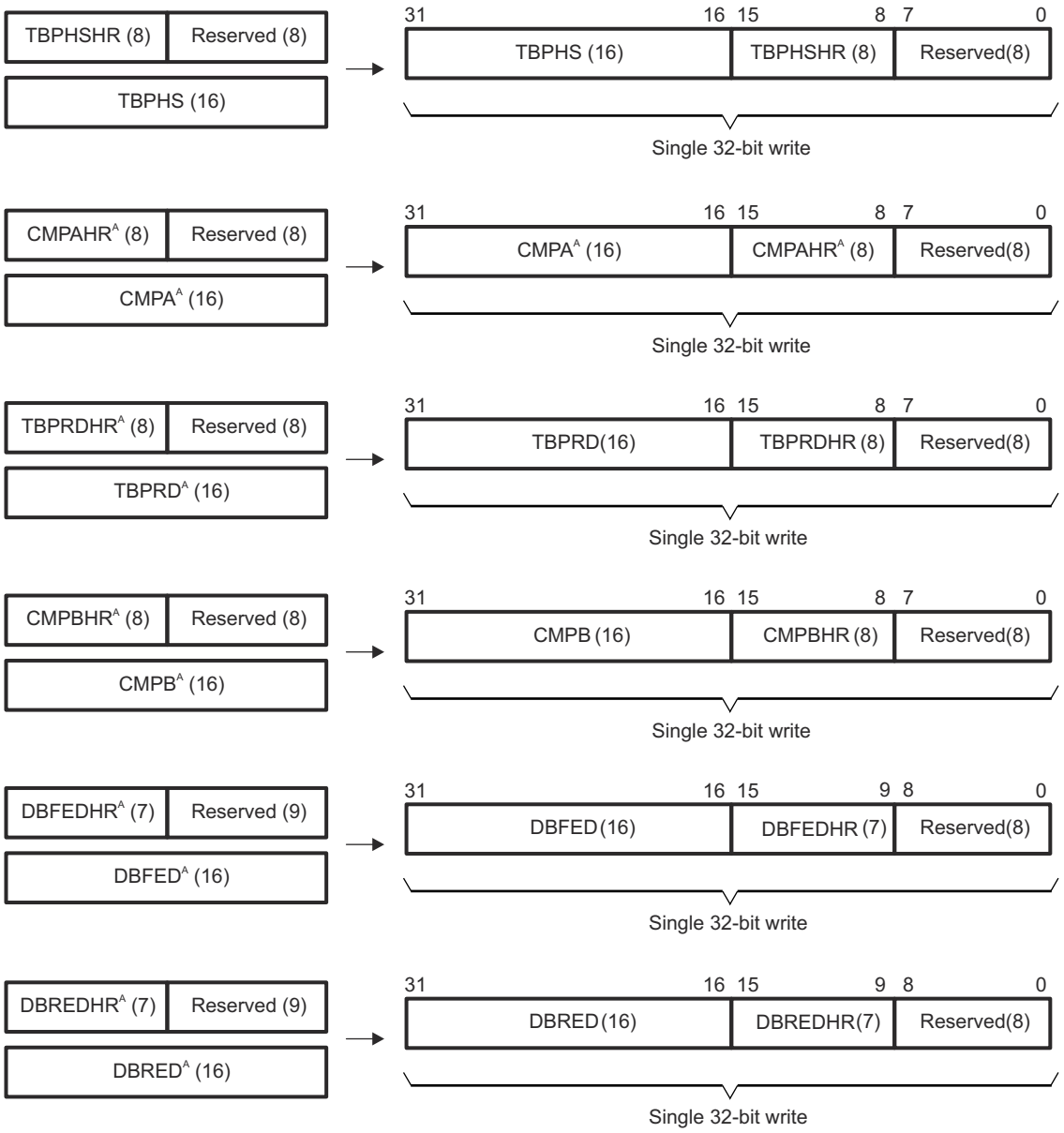


Figure 26-80. Operating Logic Using MEP

26.15.1.1 Controlling the HRPWM Capabilities

The MEP of the HRPWM is controlled by six extension registers. These HRPWM registers are concatenated with the 16-bit TBPHS, TBPRD, CMPA, CMPBM, DBREDM, and DBFEDM registers used to control PWM operation.

- TBPHSHR - Time Base Phase High Resolution Register
- CMPAHR - Counter Compare A High Resolution Register; CMPAHR is for use with the AQ output of Channel A, and is not related to CMPA
- TBPRDHR - Time Base Period High Resolution Register. (available on some devices)
- CMPBHR - Counter Compare B High Resolution Register; CMPBHR is for use with the AQ output of Channel B, and is not related to CMPB
- DBREDHR - Dead-band Generator Rising Edge Delay High Resolution Register
- DBFEDHR - Dead-band Generator Falling Edge Delay High Resolution Register



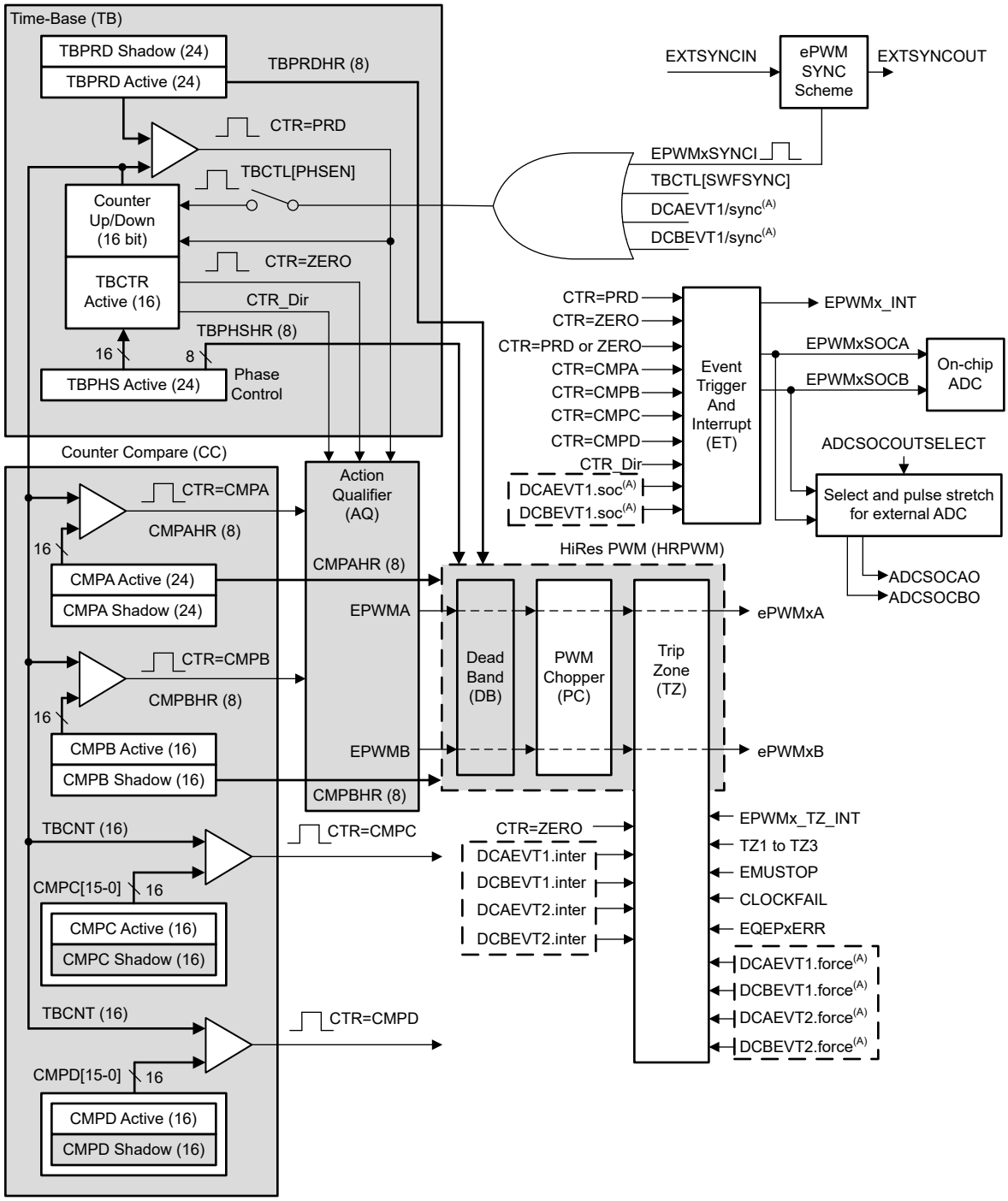
A. Dependent upon your device, these registers can be mirrored and can be written to at two different memory locations.

Figure 26-81. HRPWM Extension Registers and Memory Configuration

Note

HRPWM capabilities on Deadband Rising Edge Delay and Falling Edge Delay is applicable only during dead band half cycle clocking Operation. The number of MEP steps is half in size [bits 15:9] than duty and phase high-resolution registers for the same reason.

HRPWM capabilities are controlled using the Channel A and B PWM signal path. HRPWM support on the Dead band signal path is available by properly configuring the HRCNFG2 register. Figure 26-82 shows how the HRPWM interfaces with the 8-bit extension registers.



A. These events are generated by the ePWM Digital Compare (DC) submodule based on the levels of the TRIPIN inputs.

Figure 26-82. HRPWM System Interface

26.15.1.2 HRPWM Source Clock

Each HRPWM module is clocked from the respective EPWMxCLK. HRCAL has a separate clock. For example, HRPWM1 is sourced from EPWM1CLK while HRPWM2 is clocked from the EPWM2CLK. Figure 26-83 shows the HRCAL and HRPWM modules are sourced from their respective ePWM clock source.

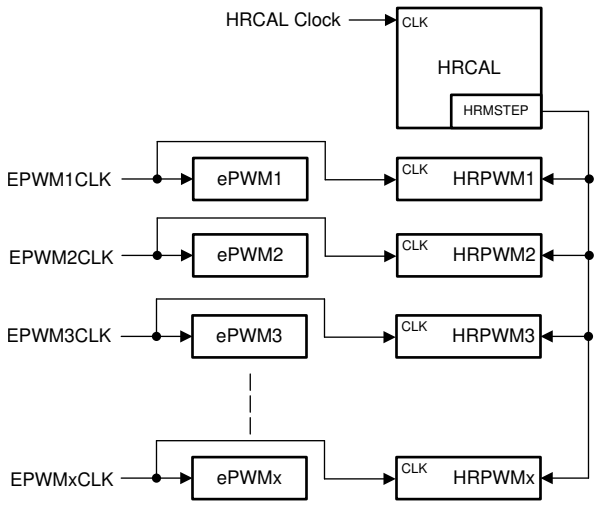


Figure 26-83. HRPWM and HRCAL Source Clock

Note

HRCAL registers can not be configured using CPU2 through the registers available on ePWM2 to ePWM8 modules.

26.15.1.3 Configuring the HRPWM

Once the ePWM has been configured to provide conventional PWM of a given frequency and polarity, the HRPWM is configured by programming the HRCNFG register in that particular ePWM module's register space. This register provides the following configuration options:

Edge Mode	The MEP can be programmed to provide precise position control on the rising edge (RE), falling edge (FE) or both edges (BE) at the same time. FE and RE are used for power topologies requiring duty cycle control (CMPA or CMPB high-resolution control), while BE is used for topologies requiring phase shifting, for example, phase shifted full bridge (TBPHS or TBPRD high-resolution control).
Control Mode	The MEP is programmed to be controlled either from the CMPAHR/CMPBHR register in case of duty cycle control or the TBPHSHR register (phase control). RE or FE control mode can be used with the CMPAHR or CMPBHR register. BE control mode can be used with the TBPHSHR register. When the MEP is controlled from the TBPRDHR register (period control), the duty cycle and phase can also be controlled using their respective high-resolution registers.
Shadow Mode	This mode provides the same shadowing (double buffering) option as in regular PWM mode. This option is valid only when operating from the CMPAHR, CMPBHR, and TBPRDHR registers and can be chosen to be the same as the regular load option for the CMPA/CMPB register. If TBPHSHR is used, then this option has no effect.
High-Resolution B Signal Control	The B signal path of an ePWM channel can generate a high-resolution output by outputting an inverted version of the high-resolution ePWMxA signal on the ePWMxB pin. A Type 2 or Type 4 HRPWM module can also enable high-resolution features on the B signal path independently of the A signal path as well.

Swap ePWMxA and ePWMxB Outputs	This mode enables the swapping of the high-resolution A and B outputs. The mode selection allows either A and B Outputs Unchanged or A Output Comes Out On B and B Output Comes Out On A.
Auto-conversion Mode	This mode is used in conjunction with the scale factor optimization (SFO) software only. For a type 4 HRPWM module, below is a description of the Auto-conversion Mode taking CMPAHR as an example. If auto-conversion is enabled, CMPAHR = fraction(PWMduty*PWMperiod)<<8. The scale factor optimization software calculates the MEP scale factor in the background code and automatically updates the HRMSTEP register with the calculated number of MEP steps per coarse step. The MEP Calibration Module then uses the values in the HRMSTEP and CMPAHR registers to automatically calculate the appropriate number of MEP steps represented by the fractional duty cycle and moves the high-resolution ePWM signal edge accordingly. If auto-conversion is disabled, the CMPAHR register behaves like a type 0 HRPWM module and CMPAHR = (fraction(PWMduty * PWMperiod) * MEP Scale Factor + 0.5)<<8). All calculations need to be performed by your code in this mode, and the HRMSTEP register is ignored. Auto-conversion for high-resolution period has the same behavior as auto-conversion for high-resolution duty cycle. Auto-conversion must always be enabled for high-resolution period mode.

Note

If the HRPWM module is configured in UP-DOWN counter mode, the shadow mode for the HRPWM registers must be set to load on both ZERO AND PERIOD. New values from the user are loaded to the shadow registers only at CTR=ZERO, but the shadow mode of for the registers must be set to both ZERO AND PERIOD. The CTR=PRD event is used for specific internal logic inside the HRPWM module.

Auto-conversion Mode performs the calculation for CMPBHR , DBREDHR, and DBFEDHR. The scale factor optimization software calculates the MEP scale factor in the background code and automatically updates the HRMSTEP register with the calculated number of MEP steps per coarse step. The MEP Calibration Module then uses the values in the HRMSTEP and CMPBHR or DBREDHR/DBFEDHR register to automatically calculate the appropriate number of MEP steps represented by the fractional components and moves the high-resolution ePWM signal edge accordingly. If auto-conversion is disabled, CMPBHR behaves the same as CMPAHR. CMPBHR = (fraction(PWMduty * PWMperiod) * MEP Scale Factor + 0.5)<<8).

You are expected to disable protection for both ePWM1 and HRPWM when the application requires access to either of these modules.

26.15.1.4 Configuring High-Resolution in Deadband Rising-Edge and Falling-Edge Delay

Once the ePWM has been configured to provide conventional PWM of a given frequency, polarity, and dead band enabled in half-cycle clocking mode, the high-resolution operation on dead band RED and FED lines are enabled by programming the HRCNFG2 register in that particular ePWM module register space. This register provides the following configuration options:

Edge Mode The MEP can be programmed to provide precise position control on the dead band rising edge (RED), dead band falling edge (FED), or both edges (rising edge of DBRED signal and falling edge of DBFED signal) at the same time.

Control Mode Selects the time event that loads the shadow value in the active register for DBRED and DBFED in high-resolution mode. Select the pulse to match the selection in the ePWM DBCTL[LOADREDMODE] and DBCTL[LOADFEDMODE] bits.

26.15.1.5 Principle of Operation

The MEP logic is capable of placing an edge in one of 255 (8 bits) discrete time steps (see the device data manual for typical MEP step size). The MEP works with the TBM and CCM registers to be certain that time steps are applied and that edge placement accuracy is maintained over a wide range of PWM frequencies, system clock frequencies, and other operating conditions. Table 26-16 shows the typical range of operating frequencies supported by the HRPWM.

Table 26-16. Relationship Between MEP Steps, PWM Frequency, and Resolution

System (MHz)	MEP Steps Per EPWMCLK (1) (2) (3)	PWM Minimum (Hz) (4)	PWM Maximum (MHz)	Resolution at Maximum (Bits) (5)
60.0	93	916	3.00	10.9
70.0	79	1068	3.50	10.6
80.0	69	1221	4.00	10.4
90.0	62	1373	4.50	10.3
100.0	56	1526	5.00	10.1

(1) TBCLK = EPWMCLK.
(2) Table data based on a MEP time resolution of 180 ps (this is an example value. See the device data sheet for MEP limits)
(3) MEP steps applied = $T_{EPWMCLK}$/180 ps in this example.
(4) PWM minimum frequency is based on a maximum period value,(TBPRD = 65535). PWM mode is asymmetrical up-count.
(5) Resolution in bits is given for the maximum PWM frequency stated.

26.15.1.5.1 Edge Positioning

Note

The following example is presented using the [CMPA:CMPAHR] register combination. The theory of operation and equations are the same, if intending to use the [CMPBM:CMPBHRM] for duty cycle control.

In a typical power control loop, a digital controller issues a duty command, usually expressed in a per unit or percentage terms. Assume that for a particular operating point, the demanded duty cycle is 0.405 or 40.5% on time and the required converter PWM frequency is 1.25 MHz. In conventional PWM generation with a system clock of 100 MHz, the duty cycle choices are in the vicinity of 40.5%. As shown in Figure 26-84, a compare value of 32 counts (duty = 40%) is the closest to 40.5% that can be attained. This is equivalent to an edge position of 320 ns instead of the desired 324 ns. This data is shown in Table 26-17.

By utilizing the MEP, an edge position much closer to the desired point of 324 ns can be achieved. Table 26-17 shows that in addition to the CMPA value, 22 steps of the MEP (CMPAHR register) positions the edge at 323.96 ns, resulting in almost zero error. In this example, it is assumed that the MEP has a step resolution of 180 ps.

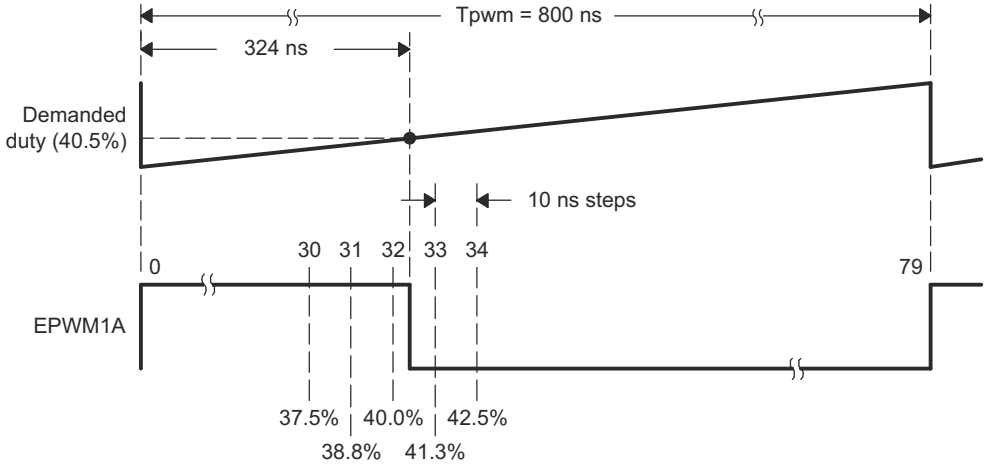


Figure 26-84. Required PWM Waveform for a Requested Duty = 40.5%

Table 26-17. CMPA versus Duty (left), and [CMPA:CMPAHR] versus Duty (right)

CMPA (count)(1) (2) (3)	Duty (%)	High Time (ns)	CMPA (count)	CMPAHR (count)	Duty (%)	High Time (ns)
28	***35.0***	280	32	18	***40.405***	323.24
29	***36.3***	290	32	19	***40.428***	323.42
30	***37.5***	300	32	20	***40.450***	323.60
31	***38.8***	310	32	21	***40.473***	323.78
32	***40.0***	320	32	22	***40.495***	323.96
33	***41.3***	330	32	23	***40.518***	324.14
34	***42.5***	340	32	24	***40.540***	324.32
			32	25	***40.563***	324.50
Required			32	26	***40.585***	324.68
32.40	***40.5***	324	32	27	***40.608***	324.86

(1) Assumed MEP step size for the above example = 180 ps. See the device-specific data sheet for typical and maximum MEP values.
(2) TBCLK = 100 MHz, 10 ns
(3) For a PWM Period register value of 80 counts, PWM Period = 80 × 10 ns = 800 ns, PWM frequency = 1/800 ns = 1.25 MHz

26.15.1.5.2 Scaling Considerations

The mechanics of how to position an edge precisely in time has been demonstrated using the resources of the standard CMPA and MEP (CMPAHR) registers. In a practical application, however, it is necessary to seamlessly provide the CPU a mapping function from a per-unit (fractional) duty cycle to a final integer (non-fractional) representation that is written to the [CMPA:CMPAHR] register combination.

To do this, first examine the scaling or mapping steps involved. It is common in control software to express duty cycle in a per-unit or percentage basis. This has the advantage of performing all needed math calculations without concern for the final absolute duty cycle, expressed in clock counts or high time in nanoseconds (ns). Furthermore, it makes the code more transportable across multiple converter types running different PWM frequencies.

To implement the mapping scheme, a two-step scaling procedure is required.

Assumptions for this example:

TBCLK	=	10 ns (100 MHz)
PWM frequency	=	1.25 MHz (1/800 ns)
Required PWM duty cycle, **PWMDuty**	=	0.405 (40.5%)
PWM period in terms of coarse steps, **PWMPeriod** (800 ns/10 ns)	=	80
Number of MEP steps per coarse step at 180 ps (10 n/180 ps), **MEP_ScaleFactor**	=	55
Value to keep CMPAHR within the range of 1-255 and fractional rounding constant (default value)	=	0.5 (0080h in Q8 format)

Step 1: Percentage Integer Duty value conversion for CMPA register

CMPA register value	=	int(**PWMDuty*****PWMPeriod**); int means integer part
	=	int(0.405 * 80)
	=	int(32.4)
CMPA register value	=	32 (20h)

Step 2: Fractional value conversion for CMPAHR register

CMPAHR	=	(frac(**PWMDuty*****PWMPeriod**)***MEP_ScaleFactor** + 0.5) <<8); frac means fractional part
	=	(frac(32.4) * 55 + 0.5) <<8; Shifting is to move the value to the high byte of CMPAHR.
	=	(0.4 * 55 + 0.5) <<8
	=	(22 + 0.5) <<8
	=	22.5 * 256; Shifting left by 8 is the same as multiplying by 256.
	=	5760 (1680h)
CMPAHR	=	1680h CMPAHR value = 1600h (lower 8 bits are ignored by hardware).

Note

If the AUTOCONV bit (HRCNFG.6) is set and the MEP_ScaleFactor is in the HRMSTEP register, then CMPAHR / CMPBHR register value = frac (PWMDuty*PWMperiod<<8). The rest of the conversion calculations are performed automatically in hardware, and the correct MEP-scaled signal edge appears on the ePWM channel output. If AUTOCONV is not set, the above calculations must be performed by software.

The MEP scale factor (MEP_ScaleFactor) varies with the system clock and DSP operating conditions. TI provides an MEP scale factor optimizing (SFO) software C function, which uses the built in diagnostics in each HRPWM and returns the best scale factor for a given operating point.

The scale factor varies slowly over a limited range so the optimizing C function can be run very slowly in a background loop.

The CMPA, CMPB, CMPAHR and CMPBHR registers are configured in memory so that the 32-bit data capability of the CPU can write this as a single concatenated value, that is, [CMPA:CMPAHR], [CMPB:CMPBHR], and so on.

The mapping scheme has been implemented in both C and assembly, as shown in Section 26.15.1.8. The actual implementation takes advantage of the 32-bit CPU architecture and is somewhat different from the steps shown in Section 26.15.1.5.2.

For time-critical control loops where every cycle counts, the assembly version is recommended. This is a cycle optimized function (11 EPWMCLK cycles) that takes a Q15 duty value as input and writes a single [CMPA:CMPAHR] value.

26.15.1.5.3 Duty Cycle Range Limitation

In high-resolution mode, the MEP is not active for 100% of the PWM period and becomes operational:

- Three EPWMCLK cycles after the period starts when high-resolution period (TBPRDHR) control is not enabled.
- When high-resolution period (TBPRDHR) control is enabled using the HRPCTL register:
  - In up-count mode: three EPWMCLK cycles after the period starts until three EPWMCLK cycles before the period ends.
  - In up-down count mode: when counting up, three cycles after CTR = 0 until three cycles before CTR = PRD, and when counting down, three cycles after CTR = PRD until three cycles before CTR = 0.
- When using DBREDHR or DBFEDHR, DBRED or DBFED (the register corresponding to the edge with high-resolution displacement) must be greater than or equal to 7.

Duty cycle range limitations are illustrated in Figure 26-85 to Figure 26-88. This limitation imposes a duty cycle limit on the MEP. For example, precision edge control is not available all the way down to 0% duty cycle. When high-resolution period control is disabled, regular PWM duty control is fully operational down to 0% duty cycle despite the unavailability of HRPWM features in the first three cycles. In most applications, this cannot be an issue as the controller regulation point is usually not designed to be close to 0% duty cycle. To better understand the useable duty cycle range, see Table 26-18. When high-resolution period control is enabled (HRPCTL[HRPE]=1), the duty cycle must not fall within the restricted range; otherwise, there can be undefined behavior on the ePWMxA output.

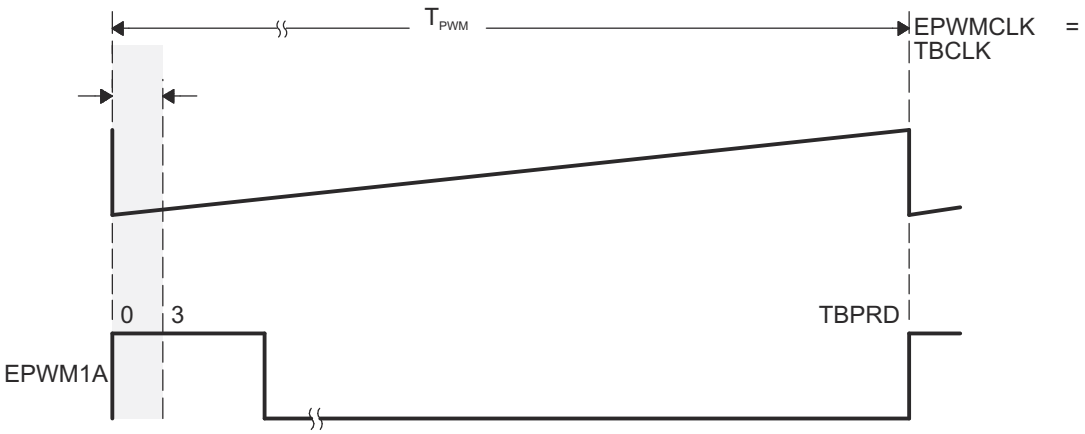


Figure 26-85. Low % Duty Cycle Range Limitation Example (HRPCTL[HRPE] = 0)

Table 26-18. Duty Cycle Range Limitation for Three EPWMCLK/TBCLK Cycles

PWM Frequency(1) (kHz)	3 Cycles Minimum Duty	3 Cycles Maximum Duty(2)
200	0.6%	99.4%
400	1.2%	98.8%
600	1.8%	98.2%
800	2.4%	97.6%
1000	3%	97%
1200	3.6%	96.4%
1400	4.2%	95.8%
1600	4.8%	95.2%
1800	5.4%	94.6%
2000	6%	94%

(1) EPWMCLK = TBCLK = 100 MHz
(2) This limitation applies only if high-resolution period (TBPRDHR) control is enabled.

If the application demands HRPWM operation below the minimum duty cycle limitation, then the HRPWM can be configured to operate in count-down mode with the rising edge position (REP) controlled by the MEP when high-resolution period is disabled (HRPCTL[HRPE] = 0). This is illustrated in Figure 26-86. In this configuration, the minimum duty cycle limitation is no longer an issue. However, there is a maximum duty limitation with same percent numbers as given in Table 26-18.

CAUTION

If the application has enabled high-resolution period control (HRPCTL[HRPE]=1), the duty cycle must not fall within the restricted range; otherwise, there can be undefined behavior on the ePWM output.

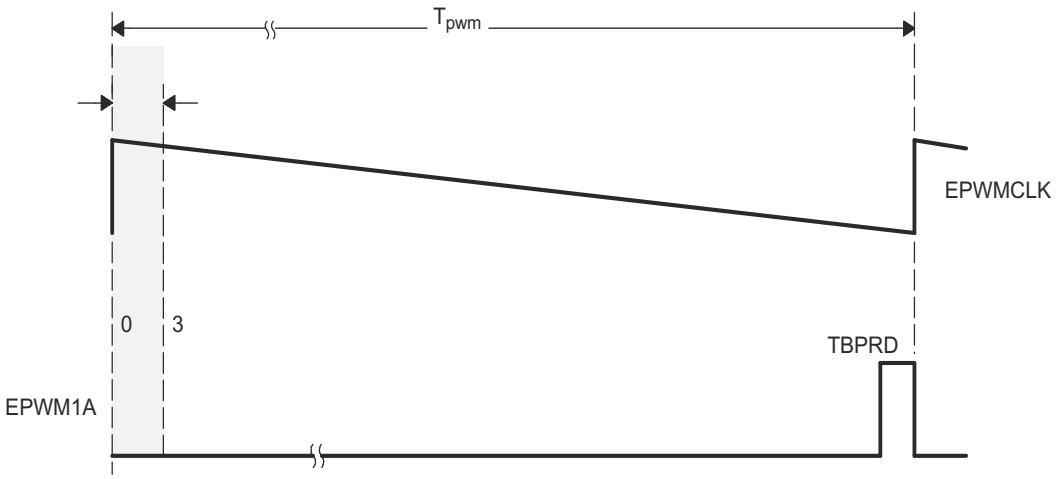


Figure 26-86. High % Duty Cycle Range Limitation Example (HRPCTL[HRPE] = 0)

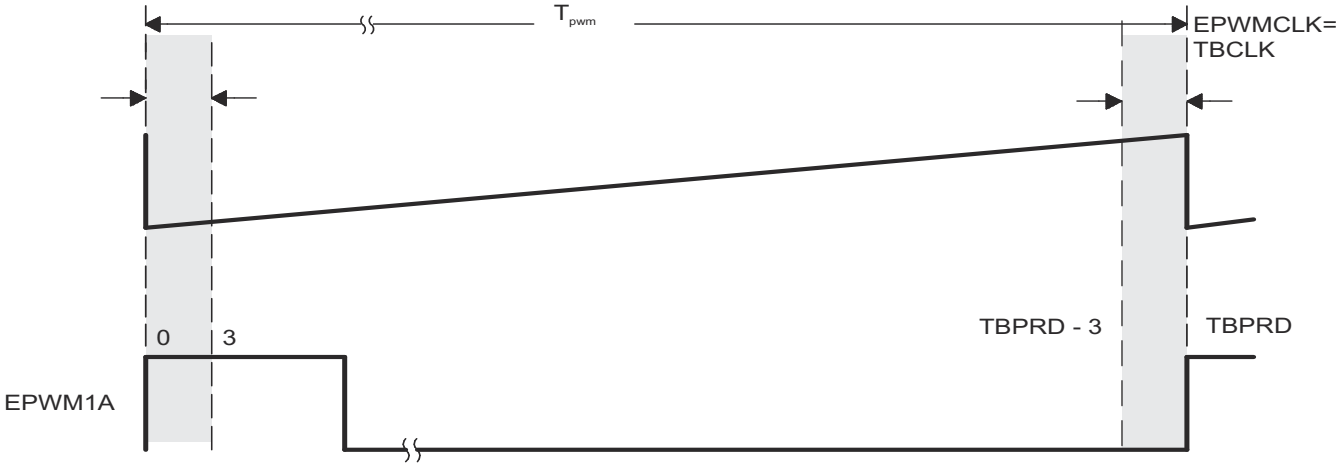


Figure 26-87. Up-Count Duty Cycle Range Limitation Example (HRPCTL[HRPE]=1)

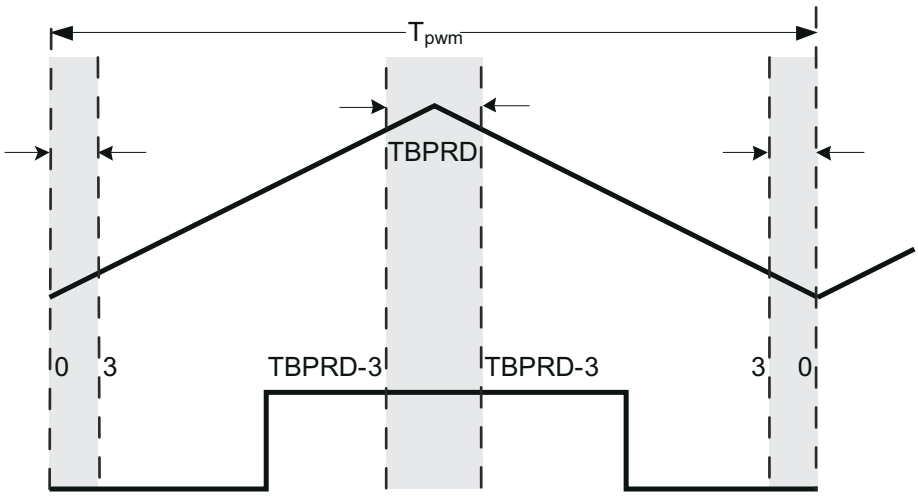


Figure 26-88. Up-Down Count Duty Cycle Range Limitation Example (HRPCTL[HRPE]=1)

26.15.1.5.4 High-Resolution Period

High-resolution period control using the MEP logic is supported on devices with a Type 1 ePWM module or greater.

Note

When high-resolution period control is enabled, on ePWMxA only, and not ePWMxB output and conversely, the non high-resolution output has ±1 TBCLK cycle jitter in up-count mode and ±2 TBCLK cycle jitter in up-down count mode.

The scaling procedure described for duty cycle in Section 26.15.1.5.2 applies for high-resolution period as well:

Assumptions for this example:

TBCLK	= 10 ns (100 MHz)
Required PWM frequency	= 175 kHz (period of 571.428)
Number of MEP steps per coarse step at 180 ps (MEP_ScaleFactor)	= 55 (10 ns / 180 ps)
Value to keep TBPRDHR within range of 1-255 and fractional rounding constant (default value)	= 0.5 (0080h in Q8 format)

Problem:

In up-count mode:

- If TBPRD = 571, then PWM frequency = 174.82 kHz (period = (571+1)* T_{TBCLK}).
- If TBPRD = 570, then PWM frequency = 175.13 kHz (period = (570+1)* T_{TBCLK}).

In up-down count mode:

- If TBPRD = 286, then PWM frequency = 174.82 kHz (period = (286*2)* T_{TBCLK}).
- If TBPRD = 285, then PWM frequency = 175.44 kHz (period = (285*2)* T_{TBCLK}).

Solution:

With 55 MEP steps per coarse step at 180 ps each:

Step 1: Percentage Integer Period value conversion for TBPRD register

Integer period value	= 571 * T_{TBCLK}
	= int (571.428) * T_{TBCLK}
	= int (PWMperiod) * T_{TBCLK}
In up-count mode:	
TBPRD	= 570 (TBPRD = period value - 1)
	= 023Ah
In up-down count mode:	= 285 (TBPRD = period value / 2)
TBPRD	= 011Dh

Step 2: Fractional value conversion for TBPRDHR register

In up-count mode:

TBPRDHR register value	= (frac(PWMperiod) * MEP_ScaleFactor + 0.5)
If auto-conversion enabled and HRMSTEP =	
MEP_ScaleFactor value (55):	=frac (PWMperiod) << 8 (Shifting is to move the value to the high byte of TBPRDHR)
TBPRDHR register value	=frac (571.428) << 8
	=0.428 × 256
	=6D00h
The auto-conversion then automatically performs the calculation, such that TBPRDHR MEP delay is scaled by hardware to:	=((TBPRDHR(15:0) >> 8) × HRMSTEP + 80h) << 8
	= (006Dh × 55 + 80h) >> 8
	=(17EBh) >> 8
Period MEP delay	=0017h MEP Steps

In up-down count mode:

TBPRDHR register value	= (frac(PWMperiod) * MEP_ScaleFactor + 0.5)
If auto-conversion enabled and HRMSTEP =	
MEP_ScaleFactor value (55):	=frac (PWMperiod / 2) << 8 (Shifting is to move the value to the high byte of TBPRDHR)
TBPRDHR register value	=frac (285.714) << 8
	=0.714 × 256
	=B600h
The auto-conversion then automatically performs the calculation, such that TBPRDHR MEP delay is scaled by hardware to:	=((TBPRDHR(15:0) >> 8) × HRMSTEP + 80h) << 8
	= (00B6h × 55 + 80h) >> 8
	=(279Ah) >> 8
Period MEP delay	=0027h MEP Steps

26.15.1.5.4.1 High-Resolution Period Configuration

To use high-resolution period, the ePWMx module must be initialized in the exact order presented.

The following steps use CMPA with shadow registers and the corresponding HRCNFG bits for high-resolution operation on EPWMxA. For high-resolution operation on EPWMxB, make the appropriate substitutions with the B channel fields.

1. Enable ePWMx clock
2. Enable HRPWM clock
3. Disable TBCLKSYNC
4. Configure ePWMx registers - AQ, TBPRD, CC, and so on.
   - ePWMx can only be configured for up-count or up-down count modes. High-resolution period is not compatible with down-count mode.
   - TBPRD and CC registers must be configured for shadow loads.
   - CMPCTL[LOADAMODE]
     - In up-count mode: CMPCTL[LOADAMODE] = 1 (load on CTR = PRD)
     - In up-down count mode: CMPCTL[LOADAMODE] = 2 (load on CTR=0 or CTR=PRD)
5. Configure the HRCNFG register such that:
   - HRCNFG[HRLOAD] = 2 (load on either CTR = 0 or CTR = PRD)
   - HRCNFG[AUTOCONV] = 1 (Enable auto-conversion)
   - HRCNFG[EDGMODE] = 3 (MEP control on both edges)
6. For TBPHS:TBPHSHR synchronization with high-resolution period, set both HRPCTL[TBPSHRLOADE] = 1 and TBCTL[PHSEN] = 1. In up-down count mode these bits must be set to 1 regardless of the contents of TBPHSHR.
7. Enable high-resolution period control (HRPCTL[HRPE] = 1)
8. Enable TBCLKSYNC
9. TBCTL[SWFSYNC] = 1
10. HRMSTEP must contain an accurate MEP scale factor (# of MEP steps per EPWMCLK coarse step) because auto-conversion is enabled. The MEP scale factor can be acquired using the SFO() function described in Section 26.15.2.
11. To control high-resolution period, write to the TBPRDHR(M) registers.

Note

When high-resolution period mode is enabled, an EPWMxSYNC pulse introduces ±1-2 cycle jitter to the PWM (±1 cycle in up-count mode and ±2 cycle in up-down count mode). For this reason, EPWMxSYNCO source cannot be set to CTR = 0 or CTR = CMPB. Otherwise, the jitter occurs on every PWM cycle with the synchronization pulse.

When EPWMxSYNCI is EPWMxSYNCO source, a software synchronization pulse can be issued only once during high-resolution period initialization. If a software sync pulse is applied while the PWM is running, the jitter appears on the PWM output at the time of the sync pulse.

26.15.1.6 Deadband High-Resolution Operation

Note

In up-count mode, the dead-band module is not available when any high-resolution mode is enabled.

Assumptions for this example:

System clock	= 10 ns (100 MHz)
Deadband enabled in half-cycle mode, TBCLK = EPWMCLK	
Required PWM frequency	1.33 MHz (1 / 750 ns)
Required PWM duty cycle	0.5 (50%)
Required Deadband Rising Edge Delay	5% over duty
Required Deadband Rising Edge Delay in ns	(0.05 * 375 ns) = 18.75 ns

Note

Similar to the duty cycle restrictions when using HRPWM, the DBRED and DBFED values must be greater than 3 to use high-resolution deadband.

Deadband delay values as a function of DBFED and DBRED:

When half-cycle clocking is enabled, the formula to calculate the falling-edge-delay and rising-edge-delay becomes:

FED = DBFED * TBCLK / 2

RED = DBRED * TBCLK / 2

DBRED and DBFED calculated values:

Required Dead band Rising Edge Delay in ns = 18.75 ns

DBRED = RED / (TBCLK / 2)

DBRED = 18.75 ns / 5 ns

DBRED Required = 3.75 ns

With 55 MEP steps per coarse step at 180 ps each:

Step 1: Integer Deadband value conversion for DBREDM register

Integer DBRED value	= int (RED / (TBCLK / 2))
	= int (3.75)
DBRED	= 3

Step 2: Fractional value conversion for Deadband high-resolution register DBREDHR

DBREDHR register value	= (frac(DBRED Required) * MEP_ScaleFactor + 0.5) << 8 (Shifting is to move the value to the high byte of DBREDHR)
	= (frac (3.75) * 55 + 0.5) << 8
	= (0.75 * 55 + 0.5) << 8
	= (41.75) * 256 Shifting left by 8 is the same as multiplying by 256.
DBREDHR value	= 29C0h MEP Steps
	Hardware ignores lower 9 bits in the above calculated DBREDHR value

Note

If the AUTOCONV bit (HRCNFG.6) is set and the MEP_ScaleFactor is in the HRMSTEP register, then DBREDHR:DBRED = frac((required DB value) < <8). The rest of the conversion calculations are performed automatically in hardware, and the correct MEP-scaled signal edge appears on the ePWM channel output. If AUTOCONV is not set, the above calculations must be performed by software.

26.15.1.7 Scale Factor Optimizing Software (SFO)

The micro edge positioner (MEP) logic is capable of placing an edge in one of 255 discrete time steps. As previously mentioned, the size of these steps is on the order of 150 ps (see the device data manual for typical MEP step size on your device). The MEP step size varies based on worst-case process parameters, operating temperature, and voltage. MEP step size increases with decreasing voltage and increasing temperature and decreases with increasing voltage and decreasing temperature. Applications that use the HRPWM feature can use the TI-supplied MEP scale factor optimization (SFO) software function. The SFO function helps to dynamically determine the number of MEP steps per EPWMCLK period while the HRPWM is in operation.

To utilize the MEP capabilities effectively, the correct value for the MEP scaling factor needs to be known by the software. To accomplish this, the HRPWM module has built in self-check and diagnostic capabilities that can be used to determine the optimum MEP scale factor value for any operating condition. TI provides a C-callable library containing one SFO function that utilizes this hardware and determines the optimum MEP scale factor. As such, MEP control and diagnostics registers are reserved for TI use.

A detailed description of the SFO library - SFO_TI_Build_V8.lib software can be found in SFO Library Software - SFO TI_Build_V8.lib.

26.15.1.8 HRPWM Examples Using Optimized Assembly Code

The best way to understand how to use the HRPWM capabilities is through two real examples:

1. Simple buck converter using asymmetrical PWM (count-up) with active high polarity.
2. DAC function using simple R+C reconstruction filter.

The following examples all have initialization and configuration code written in C. To make these easier to understand, the #defines shown below are used.

Example 26-2 assumes MEP step size of 150 ps and does not use the SFO library.

Example 26-2. #Defines for HRPWM Header Files

```
// HRPWM (High Resolution PWM) //
================================
// HRCNFG
#define HR_Disable 0x0
#define HR_REP 0x1          // Rising Edge position
#define HR_FEP 0x2          // Falling Edge position
#define HR_BEP 0x3          // Both Edge position #define HR_CMP 0x0 // CMPAHR controlled
#define HR_PHS 0x1          // TBPHSHR controlled #define HR_CTR_ZERO 0x0 // CTR = Zero event
#define HR_CTR_PRD 0x1      // CTR = Period event
#define HR_CTR_ZERO_PRD 0x2 // CTR = ZERO or Period event
#define HR_NORM_B  0x0      // Normal ePWMxB output
#define HR_INVERT_B 0x1     // ePWMxB is inverted ePWMxA output
```

26.15.1.8.1 Implementing a Simple Buck Converter

In this example, the PWM requirements are:

- PWM frequency = 1 MHz (that is, TBPRD = 100)
- PWM mode = asymmetrical, up-count
- Resolution = 12.7 bits (with a MEP step size of 150 ps)

Figure 26-89 and Figure 26-90 show the required PWM waveform. As explained previously, configuration for the ePWM1 module is almost identical to the normal case except that the appropriate MEP options need to be enabled/selected.

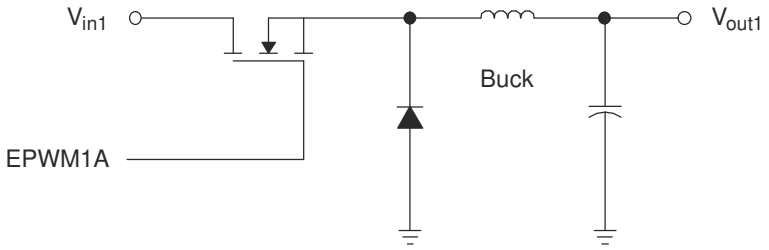


Figure 26-89. Simple Buck Controlled Converter Using a Single PWM

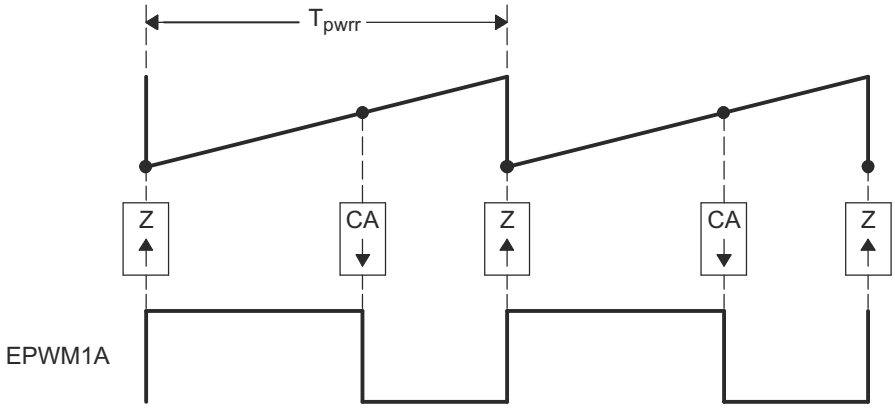


Figure 26-90. PWM Waveform Generated for Simple Buck Controlled Converter

The example code shown consists of two main parts:

- Initialization code (executed once)
- Run time code (typically executed within an ISR)

Example 26-3 shows the Initialization code. The first part is configured for conventional PWM. The second part sets up the HRPWM resources.

This example assumes MEP step size of 150 ps and does not use the SFO library.

Example 26-4 shows an assembly example of run-time code for the HRPWM buck converter.

Example 26-3. HRPWM Buck Converter Initialization Code

```
void HrBuckDrvCnf(void)
 {
// Config for conventional PWM first
EPwm1Regs.TBCTL.bit.PRDLD = TB_IMMEDIATE;         // set Immediate load
EPwm1Regs.TBPRD = 100;                            // Period set for 1000 kHz PWM
hrbuck_period = 200;                              // Used for Q15 to Q0 scaling
EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP;
EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;           // EPWM1 is the Master

EPwm1Regs.EPWMSYNCOUTEN.all = TB_SYNC_DISABLE;
EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;
EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;
// Note: ChB is initialized here only for comparison purposes, it is not required

EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;
EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;     // optional
EPwm1Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;       // optional
EPwm1Regs.AQCTLA.bit.ZRO = AQ_SET;
EPwm1Regs.AQCTLA.bit.CAU = AQ_CLEAR;
EPwm1Regs.AQCTLB.bit.ZRO = AQ_SET;                // optional
EPwm1Regs.AQCTLB.bit.CBU = AQ_CLEAR;              // optional
// Now configure the HRPWM resources
EALLOW;                                           // Note these registers are protected
                                                  // and act only on ChA
EPwm1Regs.HRCNFG.all = 0x0;                       // clear all bits first
EPwm1Regs.HRCNFG.bit.EDGMODE = HR_FEP;            // Control Falling Edge Position
EPwm1Regs.HRCNFG.bit.CTLMODE = HR_CMP;            // CMPAHR controls the MEP
EPwm1Regs.HRCNFG.bit.HRLOAD = HR_CTR_ZERO;        // Shadow load on CTR=Zero
EDIS;
MEP_ScaleFactor = 66*256;                         // Start with typical Scale Factor
                                                  // value for 100 MHz
                                                  // Note: Use SFO functions to update
                                                     MEP_ScaleFactor dynamically
}
```

Example 26-4. HRPWM Buck Converter Run-Time Code

```
EPWM1_BASE .set 0x6800
CMPAHR1 .set EPWM1_BASE+0x8
;=============================================
HRBUCK_DRV; (can execute within an ISR or loop)
;=============================================
     MOVW DP, #_HRBUCK_In
     MOVL XAR2,@_HRBUCK_In       ; Pointer to Input Q15 Duty (XAR2)
     MOVL XAR3,#CMPAHR1          ; Pointer to HRPWM CMPA reg (XAR3)

; Output for EPWM1A (HRPWM)
     MOV T,*XAR2 ; T <= Duty
     MPYU ACC,T,@_hrbuck_period  ; Q15 to Q0 scaling based on Period
     MOV T,@_MEP_ScaleFactor     ; MEP scale factor (from optimizer s/w)
     MPYU P,T,@AL                ; P <= T * AL, Optimizer scaling
     MOVH @AL,P                  ; AL <= P, move result back to ACC
     ADD ACC, #0x080             ; MEP range and rounding adjustment
     MOVL *XAR3,ACC              ; CMPA:CMPAHR(31:8) <= ACC

; Output for EPWM1B (Regular Res) Optional - for comparison purpose only
     MOV *+XAR3[2],AH            ; Store ACCH to regular CMPB
```

26.15.1.8.2 Implementing a DAC Function Using an R+C Reconstruction Filter

In this example, the PWM requirements are:

- PWM frequency = 400 kHz (that is, TBPRD = 250)
- PWM mode = Asymmetrical, Up-count
- Resolution = 14 bits (MEP step size = 150 ps)

Figure 26-91 and Figure 26-92 show the DAC function and the required PWM waveform. As explained previously, configuration for the ePWM1 module is almost identical to the normal case except that the appropriate MEP options need to be enabled/selected.

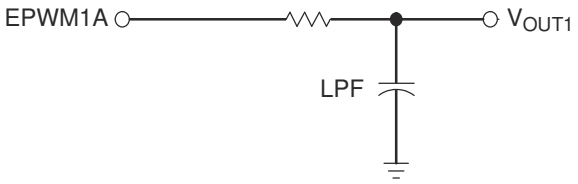


Figure 26-91. Simple Reconstruction Filter for a PWM-based DAC

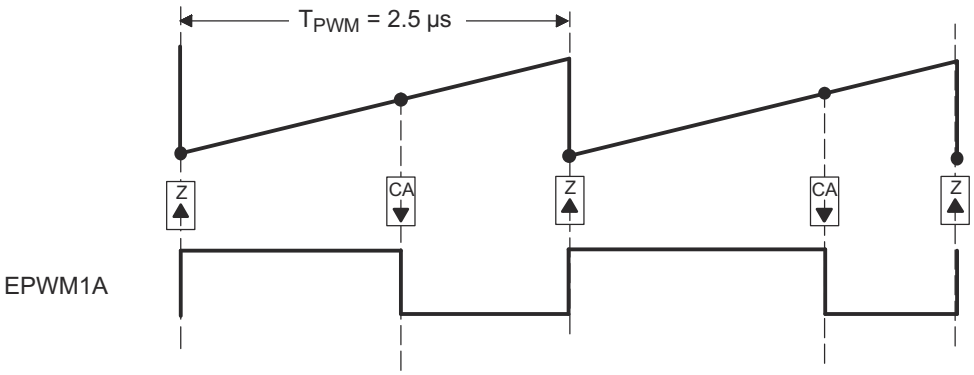


Figure 26-92. PWM Waveform Generated for the PWM DAC Function

The example code shown consists of two main parts:

- Initialization code (executed once)
- Run time code (typically executed within an ISR)

This example assumes a typical MEP_SP and does not use the SFO library.

Example 26-5 shows the Initialization code. The first part is configured for conventional PWM. The second part sets up the HRPWM resources.

Example 26-6 shows an assembly example of run-time code that can execute in a high-speed ISR loop.

Example 26-5. PWM DAC Function Initialization Code

```
void HrPwmDacDrvCnf(void)
{
// Config for conventional PWM first
EPwm1Regs.TBCTL.bit.PRDLD = TB_IMMEDIATE;      // Set Immediate load
EPwm1Regs.TBPRD = 250;                         // Period set for 400 kHz PWM
hrDAC_period = 250;                            // Used for Q15 to Q0 scaling
EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP;
EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;        // EPWM1 is the Master

EPwm1Regs.EPWMSYNCOUTEN.all = TB_SYNC_DISABLE;
EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;
EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;
// Note: ChB is initialized here only for comparison purposes, it is not required

EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;
EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;  // optional
EPwm1Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;    // optional

EPwm1Regs.AQCTLA.bit.ZRO = AQ_SET;
EPwm1Regs.AQCTLA.bit.CAU = AQ_CLEAR;
EPwm1Regs.AQCTLB.bit.ZRO = AQ_SET;             // optional
EPwm1Regs.AQCTLB.bit.CBU = AQ_CLEAR;           // optional
// Now configure the HRPWM resources
EALLOW;                                        // Note these registers are protected
                                               // and act only on ChA.
EPwm1Regs.HRCNFG.all = 0x0;                    // Clear all bits first
EPwm1Regs.HRCNFG.bit.EDGMODE = HR_FEP;         // Control falling edge position
EPwm1Regs.HRCNFG.bit.CTLMODE = HR_CMP;         // CMPAHR controls the MEP.
EPwm1Regs.HRCNFG.bit.HRLOAD = HR_CTR_ZERO;     // Shadow load on CTR=Zero.
EDIS;
MEP_ScaleFactor = 66*256;                      // Start with typical Scale Factor
                                               // value for 100 MHz.
                                               // Use SFO functions to update MEP_ScaleFactor
                                               // dynamically.
}
```

Example 26-6. PWM DAC Function Run-Time Code

```
EPWM1_BASE .set 0x6800
CMPAHR1 .set EPWM1_BASE+0x8
;================================================
HRPWM_DAC_DRV; (can execute within an ISR or loop)
;================================================
      MOVW DP, #_HRDAC_In
      MOVL XAR2,@_HRDAC_In             ; Pointer to input Q15 duty (XAR2)
      MOVL XAR3,#CMPAHR1               ; Pointer to HRPWM CMPA reg (XAR3)

; Output for EPWM1A (HRPWM
      MOV T,*XAR2                      ; T <= duty
      MPY ACC,T,@_hrDAC_period         ; Q15 to Q0 scaling based on period
      ADD ACC,@_HrDAC_period<<15       ; Offset for bipolar operation
      MOV T,@_MEP_ScaleFactor          ; MEP scale factor (from optimizer s/w)
      MPYU P,T,@AL                     ; P <= T * AL, optimizer scaling
      MOVH @AL,P                       ; AL <= P, move result back to ACC
      ADD ACC, #0x080                  ; MEP range and rounding adjustment
      MOVL *XAR3,ACC                   ; CMPA:CMPAHR(31:8) <= ACC

; Output for EPWM1B (Regular Res) Optional - for comparison purpose only
      MOV *+XAR3[2],AH                 ; Store ACCH to regular CMPB
```

26.15.2 SFO Library Software - SFO_TI_Build_V8.lib

Table 26-19 lists several features of the SFO_TI_Build_V8.lib library.

Table 26-19. SFO Library Features

	SFO_TI_Build_V8.lib	Unit
Completion-checking?	Yes	Function return value
Typical cycles required for SFO() to update MEP_ScaleFactor if called repetitively without interrupts	130,000	EPWMCLK cycles

26.15.2.1 Scale Factor Optimizer Function - int SFO()

This routine drives the micro-edge positioner (MEP) calibration module to run SFO diagnostics and determine the appropriate MEP scale factor (number of MEP steps per coarse EPWMCLK step) for a device at any given time.

If EPWMCLK = TBCLK = 100 MHz and assuming the MEP step size is 150 ps, the typical scale factor value at 100 MHz = 66 MEP steps per TBCLK unit (10 ns)

The function returns a MEP scale factor value:

MEP_ScaleFactor = Number of MEP steps per EPWMCLK

Constraints when using this function:

- SFO() can be used with a minimum EPWMCLK = TBCLK = 50 MHz. MEP diagnostics logic uses EPWMCLK and not TBCLK, so the EPWMCLK restriction is an important constraint. Below 50 MHz with device process variation, the MEP step size can decrease under cold temperature and high core voltage conditions to such a point that 255 MEP steps do not span an entire EPWMCLK cycle.
- At any time, SFO() can be called to run SFO diagnostics on the MEP calibration module.

Usage:

- SFO() can be called at any time in the background while the ePWM channels are running in HRPWM mode. The scale factor result obtained can be applied to all ePWM channels running in HRPWM mode because the function makes use of the diagnostics logic in the MEP calibration module (which runs independently of ePWM channels).
- This routine returns a 1 when calibration is finished and a new scale factor has been calculated or returns a 0 if calibration is still running. The routine returns a 2 if there is an error, and the MEP_ScaleFactor is greater than the maximum 255 fine steps per coarse EPWMCLK cycle. In this case, the HRMSTEP register maintains the last MEP scale factor value less than 256 for auto conversion.
- All ePWM modules operating in HRPWM incur only a 3 EPWMCLK cycle minimum duty cycle limitation when high-resolution period control is not used. If high-resolution period control is enabled, there is an additional duty cycle limitation 3-EPWMCLK cycles before the end of the PWM period (see Section 26.15.1.5.3).
- The SFO() function also updates the HRMSTEP register with the scale factor result. If the HRCNFG[AUTOCONV] bit is set, the application software is responsible only for setting CMPAHR = fraction(PWMduty*PWMperiod) << 8 or CMPBHR = fraction(PWMduty*PWMperiod) << 8 or TBPRDHR = fraction (PWMperiod) while running SFO() in the background. The MEP Calibration Module then uses the values in the HRMSTEP and CMPAHR/CMPBHR/TBPRDHR register to automatically calculate the appropriate number of MEP steps represented by the fractional duty cycle or period and move the high-resolution ePWM signal edge accordingly.
- If the HRCNFG[AUTOCONV] bit is clear, the HRMSTEP register is ignored. The application software needs to perform the necessary calculations manually so that:
  - CMPAHR = (fraction(PWMduty * PWMperiod) * MEP Scale Factor) << 8 + 0x080.
  - Similar behavior applies for TBPHSHR, CMPBHR, DBREDHR, and DBFEDHR. Auto-conversion must be enabled when using TBPRDHR.

The following code snippet shows how to use the HRPWM DUTY using driverlib functions.

```
float32_t dutyFine = 85.62;
float32_t count = (dutyFine * (float32_t)(EPWM_TIMER_TBPRD << 8))/100;
uint32_t compCount = (count);
HRPWM_setCounterCompareValue(EPWM1_BASE, HRPWM_COUNTER_COMPARE_A, compCount);
HRPWM_setCounterCompareValue(EPWM1_BASE, HRPWM_COUNTER_COMPARE_B, compCount);
```

The routine can be run as a background task in a slow loop requiring negligible CPU cycles. The repetition rate at which an SFO function needs to be executed depends on the application's operating environment. As with all digital CMOS devices, temperature and supply voltage variations have an effect on MEP operation. However, in most applications these parameters vary slowly and therefore is often sufficient to execute the SFO function once every 5 to 10 seconds. If more rapid variations are expected, then execution can be performed more frequently to match the application. Note there is no high limit restriction on the SFO function repetition rate; hence, the SFO function can execute as quickly as the background loop is capable.

While using the HRPWM feature, HRPWM logic is not active for the first 3 EPWMCLK cycles of the PWM period (and the last 3 EPWMCLK cycles of the PWM period if TBPRDHR is used). While running the application in this configuration, if high-resolution period control is disabled (HRPCTL[HRPE=0]) and the CMPA/CMPB register value is less than three cycles, then the CMPAHR/CMPBHR register must be cleared to zero. If high-resolution period control is enabled (HRPCTL[HRPE=1]), the CMPA register value must not fall below 3 or above TBPRD-3.This can avoid any unexpected transitions on the PWM signal.

26.15.2.2 Software Usage

The software library function SFO(), calculates the MEP scale factor for the HRPWM-supported ePWM modules. The scale factor is an integer value in the range 1-255, and represents the number of micro step edge positions available for a system clock period. The scale factor value is returned in an integer variable called MEP_ScaleFactor. For example, see Table 26-20.

Table 26-20. Factor Values

Software Function call	Functional Description	Updated Variables
SFO()	Returns MEP scale factor in the HRMSTEP register	MEP_ScaleFactor and HRMSTEP register.

To use the HRPWM feature of the ePWMs, it is recommended that the SFO function be used as described here.

Step 1. Add "Include" Files

The SFO_V8.h file needs to be included as follows. This include file is mandatory while using the SFO library function. For the SFO() to operate, the appropriate (Device)_Device.h and (Device)_Epwm_defines.h must be included in the project. These include files are optional if customized header files are used in the end applications.

Example 26-7. A Sample of How to Add "Include" Files

```
#include "F28x7x_Device.h"         // F28x7x Headerfile
#include "F28x7x_EPwm_defines.h"   // init defines
#include "SFO_V8.h"                // SFO lib functions (needed for HRPWM)
```

Declare an integer variable for the scale factor value as shown below.

Example 26-8. Declaring an Element

```
int MEP_ScaleFactor = 0;   //scale factor value
volatile struct EPWM_REGS *ePWM[] = {0, &EPwm1Regs, &EPwm2Regs, &EPwm3Regs,
&EPwm4Regs};
```

Step 3. MEP_ScaleFactor Initialization

The SFO() function does not require a starting scale factor value in MEP_ScaleFactor. Prior to using the MEP_ScaleFactor variable in application code, SFO() can be called to drive the MEP calibration module to calculate an MEP_ScaleFactor value.

As part of the one-time initialization code prior to using MEP_ScaleFactor, include the following:

Example 26-9. Initializing With a Scale Factor Value

```
MEP_ScaleFactor initialized using function SFO ()
while (SFO() == 0) {} // MEP_ScaleFactor calculated by MEP Cal Module
```

Step 4. Application Code

While the application is running, fluctuations in both device temperature and supply voltage can be expected. To be sure that good Scale Factors are used for each ePWM module, the SFO function can be re-run periodically as part of a slower back-ground loop. Some examples of this are shown here.

Note

See the HRPWM_SFO example in the device-specific C/C++ header files and peripheral examples available from the TI website.

Example 26-10. SFO Function Calls

```
main ()
 {
 int status;
    // User code
    // ePWM1, 2, 3, 4 are running in HRPWM mode
    // The status variable returns 1 once a new MEP_ScaleFactor has been
    // calculated by the MEP Calibration Module running SFO
    // diagnostics.
status = SFO();
if(status==2) {ESTOP0;}   // The function returns a 2 if MEP_ScaleFactor is greater
                          // than the maximum 255 allowed (error condition)
 }
```

26.16 Software

26.16.1 EPWM Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/epwm

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

26.16.1.1 ePWM Trip Zone

FILE: epwm_ex1_trip_zone.c

This example configures ePWM1 and ePWM2 as follows
- ePWM1 has TZ1 as one shot trip source
- ePWM2 has TZ1 as cycle by cycle trip source

Initially tie TZ1 high. During the test, monitor ePWM1 or ePWM2 outputs on a scope. Pull TZ1 low to see the effect.

External Connections
- ePWM1A is on GPIO0
- ePWM2A is on GPIO2
- TZ1 is on GPIO12

This example also makes use of the Input X-BAR. GPIO12 (the external trigger) is routed to the input X-BAR, from which it is routed to TZ1.

The TZ-Event is defined such that ePWM1A will undergo a One-Shot Trip and ePWM2A will undergo a Cycle-By-Cycle Trip.

26.16.1.2 ePWM Up Down Count Action Qualifier

FILE: epwm_ex2_updown_aq.c

This example configures ePWM1, ePWM2, ePWM3 to produce a waveform with independent modulation on ePWMxA and ePWMxB.

The compare values CMPA and CMPB are modified within the ePWM's ISR.

The TB counter is in up/down count mode for this example.

View the ePWM1A/B(GPIO0 & GPIO1), ePWM2A/B(GPIO2 &GPIO3) and ePWM3A/B(GPIO4 & GPIO5) waveforms on oscilloscope.

26.16.1.3 ePWM Synchronization

FILE: epwm_ex3_synchronization.c

This example configures ePWM1, ePWM2, ePWM3 and ePWM4 as follows
- ePWM1 without phase shift as sync source
- ePWM2 with phase shift of 300 TBCLKs
- ePWM3 with phase shift of 600 TBCLKs
- ePWM4 with phase shift of 900 TBCLKs

External Connections
- GPIO0 EPWM1A
- GPIO1 EPWM1B
- GPIO2 EPWM2A
- GPIO3 EPWM2B
- GPIO4 EPWM3A
- GPIO5 EPWM3B
- GPIO6 EPWM4A
- GPIO7 EPWM4B

Watch Variables

- None.

26.16.1.4 ePWM Digital Compare

FILE: epwm_ex4_digital_compare.c

This example configures ePWM1 as follows

- ePWM1 with DCAEVT1 forcing the ePWM output LOW
- GPIO25 is used as the input to the INPUT XBAR INPUT1
- INPUT1 (from INPUT XBAR) is used as the source for DCAEVT1
- GPIO25's PULL-UP resistor is enabled, in order to test the trip, PULL this pin to GND

External Connections

- GPIO0 EPWM1A
- GPIO1 EPWM1B
- GPIO25 TZ1, pull this pin low to trip the ePWM

Watch Variables

- None.

26.16.1.5 ePWM Digital Compare Event Filter Blanking Window

FILE: epwm_ex5_digital_compare_event_filter.c

This example configures ePWM1 as follows

- ePWM1 with DCAEVT1 forcing the ePWM output LOW
- GPIO25 is used as the input to the INPUT XBAR INPUT1
- INPUT1 (from INPUT XBAR) is used as the source for DCAEVT1
- GPIO25's PULL-UP resistor is enabled, in order to test the trip, PULL this pin to GND
- ePWM1 with DCBEVT1 forcing the ePWM output LOW
- GPIO25 is used as the input to the INPUT XBAR INPUT1
- INPUT1 (from INPUT XBAR) is used as the source for DCAEVT1
- GPIO25's PULL-UP resistor is enabled, in order to test the trip, PULL this pin to GND
- DCBEVT1 uses the filtered version of DCBEVT1
- The DCFILT signal uses the blanking window to ignore the DCBEVT1 for the duration of DC Blanking window

External Connections

- GPIO0 EPWM1A
- GPIO1 EPWM1B
- GPIO25 TRIPIN1, pull this pin low to trip the ePWM

Watch Variables

- None.

26.16.1.6 ePWM Valley Switching

FILE: epwm_ex6_valley_switching.c

This example configures ePWM1 as follows

- ePWM1 with DCAEVT1 forcing the ePWM output LOW
- GPIO25 is used as the input to the INPUT XBAR INPUT1
- INPUT1 (from INPUT XBAR) is used as the source for DCAEVT1
- GPIO25 is set to output and toggled in the main loop to trip the PWM
- ePWM1 with DCBEVT1 forcing the ePWM output LOW
- GPIO25 is used as the input to the INPUT XBAR INPUT1
- INPUT1 (from INPUT XBAR) is used as the source for DCAEVT1
- GPIO25 is set to output and toggled in the main loop to trip the PWM
- DCBEVT1 uses the filtered version of DCBEVT1
- The DCFILT signal uses the valley switching module to delay the

- DCFILT signal by a software defined DELAY value.

External Connections
- GPIO0 EPWM1A
- GPIO1 EPWM1B
- GPIO25 TRIPIN1 (Output Pin, toggled through software)

Watch Variables
- None.

26.16.1.7 ePWM Digital Compare Edge Filter

FILE: epwm_ex7_edge_filter.c

This example configures ePWM1 as follows
- ePWM1 with DCBEVT2 forcing the ePWM output LOW as a CBC source
- GPIO25 is used as the input to the INPUT XBAR INPUT1
- INPUT1 (from INPUT XBAR) is used as the source for DCBEVT2
- GPIO25 is set to output and toggled in the main loop to trip the PWM
- The DCBEVT2 is the source for DCFILT
- The DCFILT will count edges of the DCBEVT2 and generate a signal to to trip the ePWM on the 4th edge of DCBEVT2

External Connections
- GPIO0 EPWM1A
- GPIO1 EPWM1B
- GPIO25 TRIPIN1 (Output Pin, toggled through software)

Watch Variables
- None.

26.16.1.8 ePWM Deadband

FILE: epwm_ex8_deadband.c

This example configures ePWM1 through ePWM6 as follows
- ePWM1 with Deadband disabled (Reference)
- ePWM2 with Deadband Active High
- ePWM3 with Deadband Active Low
- ePWM4 with Deadband Active High Complimentary
- ePWM5 with Deadband Active Low Complimentary
- ePWM6 with Deadband Output Swap (switch A and B outputs)

External Connections
- GPIO0 EPWM1A
- GPIO1 EPWM1B
- GPIO2 EPWM2A
- GPIO3 EPWM2B
- GPIO4 EPWM3A
- GPIO5 EPWM3B
- GPIO6 EPWM4A
- GPIO7 EPWM4B
- GPIO8 EPWM5A
- GPIO9 EPWM5B
- GPIO10 EPWM6A
- GPIO11 EPWM6B

Watch Variables
- None.

26.16.1.9 ePWM DMA

FILE: epwm_ex9_dma.c

This example configures ePWM1 and DMA as follows:

- ePWM1 is set up to generate PWM waveforms
- DMA5 is set up to update the CMPAHR, CMPA, CMPBHR and CMPB every period with the next value in the configuration array. This allows the user to create a DMA enabled fifo for all the CMPx and CMPxHR registers to generate unconventional PWM waveforms.
- DMA6 is set up to update the TBPHSHR, TBPHS, TBPRDHR and TBPRD every period with the next value in the configuration array.
- Other registers such as AQCTL can be controlled through the DMA as well by following the same procedure. (Not used in this example)

External Connections

- GPIO0 EPWM1A
- GPIO1 EPWM1B

Watch Variables

- None.

26.16.1.10 ePWM Chopper

FILE: epwm_ex10_chopper.c

This example configures ePWM1, ePWM2, ePWM3 and ePWM4 as follows

- ePWM1 with Chopper disabled (Reference)
- ePWM2 with chopper enabled at 1/8 duty cycle
- ePWM3 with chopper enabled at 6/8 duty cycle
- ePWM4 with chopper enabled at 1/2 duty cycle with One-Shot Pulse enabled

External Connections

- GPIO0 EPWM1A
- GPIO1 EPWM1B
- GPIO2 EPWM2A
- GPIO3 EPWM2B
- GPIO4 EPWM3A
- GPIO5 EPWM3B
- GPIO6 EPWM4A
- GPIO7 EPWM4B

Watch Variables

- None.

26.16.1.11 EPWM Configure Signal

FILE: epwm_ex11_configure_signal.c

This example configures ePWM1, ePWM2, ePWM3 to produce signal of desired frequency and duty. It also configures phase between the configured modules.

Signal of 10kHz with duty of 0.5 is configured on ePWMxA & ePWMxB with ePWMxB inverted. Also, phase of 120 degree is configured between ePWM1 to ePWM3 signals.

During the test, monitor ePWM1, ePWM2, and/or ePWM3 outputs on an oscilloscope.

- ePWM1A is on GPIO0
- ePWM1B is on GPIO1
- ePWM2A is on GPIO2
- ePWM2B is on GPIO3
- ePWM3A is on GPIO4

- ePWM3B is on GPIO5

26.16.1.12 Realization of Monoshot mode

FILE: epwm_ex12_monoshot_mode.c

This example showcases how to generate monoshot PWM output based on external trigger, that is, generating just a single pulse output on receipt of an external trigger. And the next pulse is generated only when the next trigger comes. The example utilizes external synchronization and T1 action qualifier event features to achieve the desired output.

ePWM1 is used to generate the monoshot output and ePWM2 is used an external trigger for that. No external connections are required as ePWM2A is fed as the trigger using Input X-BAR automatically.

ePWM1 is configured to generated a single pulse of 0.5 µs when received an external trigger. This is achieved by enabling the phase synchronization feature and configuring EPWMxSYNCI as EXTSYNCIN1. And this EPWMxSYNCI is also configured as T1 event of action qualifier to set output HIGH while CTR = PRD action is used to set output LOW.

ePWM2 is configured to generate a 100 KHz signal with a duty of 1% (to simulate a rising edge trigger) which is routed to EXTSYNCIN1 using Input XBAR.

Observe GPIO0 (EPWM1A : Monoshot Output) and GPIO2(EPWM2 : External Trigger) on oscilloscope.

NOTE : In the following example, the ePWM timer is still running in a continuous mode rather than a one-shot mode thus for more reliable implementation, refer to CLB based one shot PWM implementation demonstrated in "clb_ex17_one_shot_pwm" example

26.16.1.13 EPWM Action Qualifier (epwm_up_aq)

FILE: epwm_ex13_up_aq.c

This example configures ePWM1, ePWM2, ePWM3 to produce an waveform with independent modulation on EPWMxA and EPWMxB.

The compare values CMPA and CMPB are modified within the ePWM's ISR.

The TB counter is in up count mode for this example.

View the EPWM1A/B(GPIO0 & GPIO1), EPWM2A/B(GPIO2 & GPIO3) and EPWM3A/B(GPIO4 & GPIO5) waveforms via an oscilloscope.

26.16.2 HRPWM Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/hrpwm

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

26.16.2.1 HRPWM Duty Control with SFO

FILE: hrpwm_ex1_duty_sfo.c

This example modifies the MEP control registers to show edge displacement for high-resolution period with ePWM in Up count mode due to the HRPWM control extension of the respective ePWM module.

This example calls the following TI's MEP Scale Factor Optimizer (SFO) software library V8 functions:

int SFO();

- updates MEP_ScaleFactor dynamically when HRPWM is in use
- updates HRMSTEP register (exists only in EPwm1Regs register space) with MEP_ScaleFactor value
- returns 2 if error: MEP_ScaleFactor is greater than maximum value of 255 (Auto-conversion may not function properly under this condition)
- returns 1 when complete for the specified channel
- returns 0 if not complete for the specified channel

This example is intended to explain the HRPWM capabilities. The code can be optimized for code efficiency. Refer to TI's Digital power application examples and TI Digital Power Supply software libraries for details.

External Connections

- Monitor ePWM1/2/3/4 A/B pins on an oscilloscope.

26.16.2.2 HRPWM Slider

FILE: hrpwm_ex2_slider.c

This example modifies the MEP control registers to show edge displacement due to HRPWM. Control blocks of the respective ePWM module channel A and B have fine edge movement due to HRPWM logic.

Monitor ePWM1 A/B pins on an oscilloscope.

26.16.2.3 HRPWM Period Control

FILE: hrpwm_ex3_prd_updown_sfo.c

This example modifies the MEP control registers to show edge displacement for high-resolution period with ePWM in Up-Down count mode due to the HRPWM control extension of the respective ePWM module.

This example calls the following TI's MEP Scale Factor Optimizer (SFO) software library V8 functions:

int SFO();

- updates MEP_ScaleFactor dynamically when HRPWM is in use
- updates HRMSTEP register (exists only in EPwm1Regs register space) with MEP_ScaleFactor value
- returns 2 if error: MEP_ScaleFactor is greater than maximum value of 255 (Auto-conversion may not function properly under this condition)
- returns 1 when complete for the specified channel
- returns 0 if not complete for the specified channel

This example is intended to explain the HRPWM capabilities. The code can be optimized for code efficiency. Refer to TI's Digital power application examples and TI Digital Power Supply software libraries for details.

External Connections

- Monitor ePWM1/2/3/4 A/B pins on an oscilloscope.

26.16.2.4 HRPWM Duty Control with UPDOWN Mode

FILE: hrpwm_ex4_duty_updown_sfo.c

This example calls the following TI's MEP Scale Factor Optimizer (SFO) software library V8 functions:

int SFO();

- updates MEP_ScaleFactor dynamically when HRPWM is in use
- updates HRMSTEP register (exists only in EPwm1Regs register space) with MEP_ScaleFactor value
- returns 2 if error: MEP_ScaleFactor is greater than maximum value of 255 (Auto-conversion may not function properly under this condition)
- returns 1 when complete for the specified channel
- returns 0 if not complete for the specified channel

This example is intended to explain the HRPWM capabilities. The code can be optimized for code efficiency. Refer to TI's Digital power application examples and TI Digital Power Supply software libraries for details.

External Connections

- Monitor ePWM1/2/3/4 A/B pins on an oscilloscope.

26.16.2.5 HRPWM Slider Test

FILE: hrpwm_ex5_slider_qformat.c

This example modifies the MEP control registers to show edge displacement due to HRPWM. Control blocks of the respective ePWM module channel A and B will have fine edge movement due to HRPWM logic. Load the hrpwm_slider.gel file. Select the HRPWM_eval from the GEL menu. A FineDuty slider graphics will show up in

CCS. Load the program and run. Use the Slider to and observe the EPWM edge displacement for each slider step change. This explains the MEP control on the EPwmxA channels.

Monitor ePWM1 & ePWM2 A/B pins on an oscilloscope.

26.16.2.6 HRPWM Duty Up Count

FILE: hrpwm_ex6_duty_sfo_qformat.c

This example modifies the MEP control registers to show edge displacement for high-resolution period with ePWM in Up count mode due to the HRPWM control extension of the respective ePWM module.

This example calls the following TI's MEP Scale Factor Optimizer (SFO) software library V8 functions:

int SFO();
- updates MEP_ScaleFactor dynamically when HRPWM is in use
- updates HRMSTEP register (exists only in EPwm1Regs register space) with MEP_ScaleFactor value
- returns 2 if error: MEP_ScaleFactor is greater than maximum value of 255 (Auto-conversion may not function properly under this condition)
- returns 1 when complete for the specified channel
- returns 0 if not complete for the specified channel

This example is intended to explain the HRPWM capabilities. The code can be optimized for code efficiency. Refer to TI's Digital power application examples and TI Digital Power Supply software libraries for details.

To run this example:
1. Run this example at maximum SYSCLKOUT
2. Activate Real time mode
3. Run the code

External Connections
- Monitor ePWM1/2 A/B pins on an oscilloscope.

Watch Variables
- status - Example run status
- updateFine - Set to 1 use HRPWM capabilities and observe in fine MEP steps(default) Set to 0 to disable HRPWM capabilities and observe in coarse SYSCLKOUT cycle steps

26.16.2.7 HRPWM Period Up-Down Count

FILE: hrpwm_ex7_prd_updown_sfo_qformat.c

This example modifies the MEP control registers to show edge displacement for high-resolution period with ePWM in Up-Down count mode due to the HRPWM control extension of the respective ePWM module.

This example calls the following TI's MEP Scale Factor Optimizer (SFO) software library V8 functions:

int SFO();
- updates MEP_ScaleFactor dynamically when HRPWM is in use
- updates HRMSTEP register (exists only in EPwm1Regs register space) with MEP_ScaleFactor value
- returns 2 if error: MEP_ScaleFactor is greater than maximum value of 255 (Auto-conversion may not function properly under this condition)
- returns 1 when complete for the specified channel
- returns 0 if not complete for the specified channel

This example is intended to explain the HRPWM capabilities. The code can be optimized for code efficiency. Refer to TI's Digital power application examples and TI Digital Power Supply software libraries for details.

To run this example:
1. Run this example at maximum SYSCLKOUT
2. Activate Real time mode
3. Run the code

External Connections

- Monitor ePWM1/2 A/B pins on an oscilloscope.

Watch Variables

- updateFine - Set to 1 use HRPWM capabilities and observe in fine MEP steps(default) Set to 0 to disable HRPWM capabilities and observe in coarse SYSCLKOUT cycle steps

26.17 ePWM Registers

This section describes the Enhanced Pulse Width Modulator registers.

26.17.1 EPWM Base Address Table (C28)

Table 26-21. EPWM Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
EPwm1Regs	EPWM_REGS	EPWM1_BASE	0x0000_4000	YES	YES	YES	YES	YES
EPwm2Regs	EPWM_REGS	EPWM2_BASE	0x0000_4100	YES	YES	YES	YES	YES
EPwm3Regs	EPWM_REGS	EPWM3_BASE	0x0000_4200	YES	YES	YES	YES	YES
EPwm4Regs	EPWM_REGS	EPWM4_BASE	0x0000_4300	YES	YES	YES	YES	YES
EPwm5Regs	EPWM_REGS	EPWM5_BASE	0x0000_4400	YES	YES	YES	YES	YES
EPwm6Regs	EPWM_REGS	EPWM6_BASE	0x0000_4500	YES	YES	YES	YES	YES
EPwm7Regs	EPWM_REGS	EPWM7_BASE	0x0000_4600	YES	YES	YES	YES	YES
EPwm8Regs	EPWM_REGS	EPWM8_BASE	0x0000_4700	YES	YES	YES	YES	YES
EPwm9Regs	EPWM_REGS	EPWM9_BASE	0x0000_4800	YES	YES	YES	YES	YES
EPwm10Regs	EPWM_REGS	EPWM10_BASE	0x0000_4900	YES	YES	YES	YES	YES
EPwm11Regs	EPWM_REGS	EPWM11_BASE	0x0000_4A00	YES	YES	YES	YES	YES
EPwm12Regs	EPWM_REGS	EPWM12_BASE	0x0000_4B00	YES	YES	YES	YES	YES
EPwm13Regs	EPWM_REGS	EPWM13_BASE	0x0000_4C00	YES	YES	YES	YES	YES
EPwm14Regs	EPWM_REGS	EPWM14_BASE	0x0000_4D00	YES	YES	YES	YES	YES
EPwm15Regs	EPWM_REGS	EPWM15_BASE	0x0000_4E00	YES	YES	YES	YES	YES
EPwm16Regs	EPWM_REGS	EPWM16_BASE	0x0000_4F00	YES	YES	YES	YES	YES

26.17.2 EPWM_REGS Registers

Table 26-22 lists the memory-mapped registers for the EPWM_REGS registers. All register offset addresses not listed in Table 26-22 should be considered as reserved locations and the register contents should not be modified.

Table 26-22. EPWM_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	TBCTL	Time Base Control Register		Go
1h	TBCTL2	Time Base Control Register 2		Go
3h	EPWMSYNCINSEL	EPWMxSYNCIN Source Select Register		Go
4h	TBCTR	Time Base Counter Register		Go
5h	TBSTS	Time Base Status Register		Go
6h	EPWMSYNCOUTEN	EPWMxSYNCOUT Source Enable Register		Go
7h	TBCTL3	Time Base Control Register 3		Go
8h	CMPCTL	Counter Compare Control Register		Go
9h	CMPCTL2	Counter Compare Control Register 2		Go
Ch	DBCTL	Dead-Band Generator Control Register		Go
Dh	DBCTL2	Dead-Band Generator Control Register 2		Go
10h	AQCTL	Action Qualifier Control Register		Go
11h	AQTSRCSEL	Action Qualifier Trigger Event Source Select Register		Go
14h	PCCTL	PWM Chopper Control Register		Go
18h	VCAPCTL	Valley Capture Control Register		Go
19h	VCNTCFG	Valley Counter Config Register		Go
20h	HRCNFG	HRPWM Configuration Register	EALLOW	Go
21h	HRPWR	HRPWM Power Register	EALLOW	Go
26h	HRMSTEP	HRPWM MEP Step Register	EALLOW	Go
27h	HRCNFG2	HRPWM Configuration 2 Register	EALLOW	Go
2Dh	HRPCTL	High Resolution Period Control Register	EALLOW	Go
2Eh	TRREM	HRPWM High Resolution Remainder Register	EALLOW	Go
34h	GLDCTL	Global PWM Load Control Register	EALLOW	Go
35h	GLDCFG	Global PWM Load Config Register	EALLOW	Go
38h	EPWMXLINK	EPWMx Link Register		Go
40h	AQCTLA	Action Qualifier Control Register For Output A		Go
41h	AQCTLA2	Additional Action Qualifier Control Register For Output A		Go
42h	AQCTLB	Action Qualifier Control Register For Output B		Go
43h	AQCTLB2	Additional Action Qualifier Control Register For Output B		Go
47h	AQSFRC	Action Qualifier Software Force Register		Go
49h	AQCSFRC	Action Qualifier Continuous S/W Force Register		Go
50h	DBREDHR	Dead-Band Generator Rising Edge Delay High Resolution Mirror Register		Go
51h	DBRED	Dead-Band Generator Rising Edge Delay High Resolution Mirror Register		Go
52h	DBFEDHR	Dead-Band Generator Falling Edge Delay High Resolution Register		Go
53h	DBFED	Dead-Band Generator Falling Edge Delay Count Register		Go
60h	TBPHS	Time Base Phase High		Go

Table 26-22. EPWM_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
62h	TBPRDHR	Time Base Period High Resolution Register		Go
63h	TBPRD	Time Base Period Register		Go
6Ah	CMPA	Counter Compare A Register		Go
6Ch	CMPB	Compare B Register		Go
6Fh	CMPC	Counter Compare C Register		Go
71h	CMPD	Counter Compare D Register		Go
74h	GLDCTL2	Global PWM Load Control Register 2		Go
77h	SWVDELVAL	Software Valley Mode Delay Register		Go
80h	TZSEL	Trip Zone Select Register	EALLOW	Go
82h	TZDCSEL	Trip Zone Digital Comparator Select Register	EALLOW	Go
84h	TZCTL	Trip Zone Control Register	EALLOW	Go
85h	TZCTL2	Additional Trip Zone Control Register	EALLOW	Go
86h	TZCTLDCA	Trip Zone Control Register Digital Compare A	EALLOW	Go
87h	TZCTLDCB	Trip Zone Control Register Digital Compare B	EALLOW	Go
8Dh	TZEINT	Trip Zone Enable Interrupt Register	EALLOW	Go
93h	TZFLG	Trip Zone Flag Register		Go
94h	TZCBCFLG	Trip Zone CBC Flag Register		Go
95h	TZOSTFLG	Trip Zone OST Flag Register		Go
97h	TZCLR	Trip Zone Clear Register	EALLOW	Go
98h	TZCBCCLR	Trip Zone CBC Clear Register	EALLOW	Go
99h	TZOSTCLR	Trip Zone OST Clear Register	EALLOW	Go
9Bh	TZFRC	Trip Zone Force Register	EALLOW	Go
A4h	ETSEL	Event Trigger Selection Register		Go
A6h	ETPS	Event Trigger Pre-Scale Register		Go
A8h	ETFLG	Event Trigger Flag Register		Go
AAh	ETCLR	Event Trigger Clear Register		Go
ACh	ETFRC	Event Trigger Force Register		Go
AEh	ETINTPS	Event-Trigger Interrupt Pre-Scale Register		Go
B0h	ETSOCPS	Event-Trigger SOC Pre-Scale Register		Go
B2h	ETCNTINITCTL	Event-Trigger Counter Initialization Control Register		Go
B4h	ETCNTINIT	Event-Trigger Counter Initialization Register		Go
C0h	DCTRIPSEL	Digital Compare Trip Select Register	EALLOW	Go
C3h	DCACTL	Digital Compare A Control Register	EALLOW	Go
C4h	DCBCTL	Digital Compare B Control Register	EALLOW	Go
C7h	DCFCTL	Digital Compare Filter Control Register	EALLOW	Go
C8h	DCCAPCTL	Digital Compare Capture Control Register	EALLOW	Go
C9h	DCFOFFSET	Digital Compare Filter Offset Register		Go
CAh	DCFOFFSETCNT	Digital Compare Filter Offset Counter Register		Go
CBh	DCFWINDOW	Digital Compare Filter Window Register		Go
CCh	DCFWINDOWCNT	Digital Compare Filter Window Counter Register		Go
CFh	DCCAP	Digital Compare Counter Capture Register		Go
D2h	DCAHTRIPSEL	Digital Compare AH Trip Select	EALLOW	Go
D3h	DCALTRIPSEL	Digital Compare AL Trip Select	EALLOW	Go
D4h	DCBHTRIPSEL	Digital Compare BH Trip Select	EALLOW	Go

Table 26-22. EPWM_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
D5h	DCBLTRIPSEL	Digital Compare BL Trip Select	EALLOW	Go
FAh	EPWMLOCK	EPWM Lock Register		Go
FDh	HWVDELVAL	Hardware Valley Mode Delay Register		Go
FEh	VCNTVAL	Hardware Valley Counter Register		Go

Complex bit access types are encoded to fit into small table cells. Table 26-23 shows the codes that are used for access types in this section.

Table 26-23. EPWM_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
WOnce	W Once	Write Write once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

26.17.2.1 TBCTL Register (Offset = 0h) [Reset = 83h]

TBCTL is shown in Figure 26-93 and described in Table 26-24.

Return to the Summary Table.

Time Base Control Register

Figure 26-93. TBCTL Register

15	14	13	12	11	10	9	8
FREE_SOFT		PHSDIR	CLKDIV			HSPCLKDIV	
R/W-0h		R/W-0h	R/W-0h			R/W-1h	
7	6	5	4	3	2	1	0
HSPCLKDIV	SWFSYNC	RESERVED		PRDLD	PHSEN	CTRMODE	
R/W-1h	R-0/W1S-0h	R-0h		R/W-0h	R/W-0h	R/W-3h	

Table 26-24. TBCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	FREE_SOFT	R/W	0h	Emulation Mode Bits. These bits select the behavior of the ePWM time-base counter during emulation events 00: Stop after the next time-base counter increment or decrement 01: Stop when counter completes a whole cycle: - Up-count mode: stop when the time-base counter = period (TBCTR = TBPRD) - Down-count mode: stop when the time-base counter = 0x00 (TBCTR = 0x00) - Up-down-count mode: stop when the time-base counter = 0x00 (TBCTR = 0x00) 1x: Free run Reset type: SYSRSn
13	PHSDIR	R/W	0h	Phase Direction Bit This bit is only used when the time-base counter is configured in the up-down-count mode. The PHSDIR bit indicates the direction the time-base counter (TBCTR) will count after a synchronization event occurs and a new phase value is loaded from the phase (TBPHS) register. This is irrespective of the direction of the counter before the synchronization event.. In the up-count and down-count modes this bit is ignored. 0: Count down after the synchronization event. 1: Count up after the synchronization event. Reset type: SYSRSn
12-10	CLKDIV	R/W	0h	Time Base Clock Pre-Scale Bits These bits select the time base clock pre-scale value (TBCLK = EPWMCLK/(HSPCLKDIV * CLKDIV): 000: /1 (default on reset) 001: /2 010: /4 011: /8 100: /16 101: /32 110: /64 111: /128 Reset type: SYSRSn

Table 26-24. TBCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9-7	HSPCLKDIV	R/W	1h	High Speed Time Base Clock Pre-Scale Bits These bits determine part of the time-base clock prescale value. TBCLK = EPWMCLK / (HSPCLKDIV x CLKDIV). This divisor emulates the HSPCLK in the TMS320x281x system as used on the Event Manager (EV) peripheral. 000: /1 001: /2 (default on reset) 010: /4 011: /6 100: /8 101: /10 110: /12 111: /14 Reset type: SYSRSn
6	SWFSYNC	R-0/W1S	0h	Software Forced Sync Pulse 0: Writing a 0 has no effect and reads always return a 0. 1: Writing a 1 forces a one-time synchronization pulse to be generated. SWFSYNC can be enabled to affect EPWMxSYNCO by setting the EPWMSYNCOUTEN.SWEN bit. Reset type: SYSRSn
5-4	RESERVED	R	0h	Reserved
3	PRDLD	R/W	0h	Active Period Reg Load from Shadow Select 0: The period register (TBPRD) is loaded from its shadow register when the time-base counter, TBCTR, is equal to zero and/or a sync event as determined by the TBCTL2[PRDLDSYNC] bit. A write/read to the TBPRD register accesses the shadow register. 1: Immediate Mode (Shadow register bypassed): A write or read to the TBPRD register accesses the active register. Reset type: SYSRSn
2	PHSEN	R/W	0h	Counter Reg Load from Phase Reg Enable 0: Do not load the time-base counter (TBCTR) from the time-base phase register (TBPHS). 1: Allow Counter to be loaded from the Phase register (TBPHS) and shadow to active load events when an EPWMxSYNCI input signal occurs or a software-forced sync signal, see bit 6. Reset type: SYSRSn
1-0	CTRMODE	R/W	3h	Counter Mode The time-base counter mode is normally configured once and not changed during normal operation. If you change the mode of the counter, the change will take effect at the next TBCLK edge and the current counter value shall increment or decrement from the value before the mode change. These bits set the time-base counter mode of operation as follows: 00: Up-count mode 01: Down-count mode 10: Up-down count mode 11: Freeze counter operation (default on reset) Reset type: SYSRSn

26.17.2.2 TBCTL2 Register (Offset = 1h) [Reset = 0h]

TBCTL2 is shown in Figure 26-94 and described in Table 26-25.

Return to the Summary Table.

Time Base Control Register 2

Figure 26-94. TBCTL2 Register

15	14	13	12	11	10	9	8
PRDLDSYNC		RESERVED		RESERVED			
R/W-0h		R-0h		R-0-0h			
7	6	5	4	3	2	1	0
OSHTSYNC	OSHTSYNCMODE	RESERVED		RESERVED			
R-0/W1S-0h	R/W-0h	R/W-0h		R-0-0h			

Table 26-25. TBCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	PRDLDSYNC	R/W	0h	Shadow to Active Period Register Load on SYNC event 00: Shadow to Active Load of TBPRD occurs only when TBCTR = 0 (same as legacy). 01: Shadow to Active Load of TBPRD occurs both when TBCTR = 0 and when SYNC occurs. 10: Shadow to Active Load of TBPRD occurs only when a SYNC is received. 11: Reserved Note: This bit selection is valid only if TBCTL[PRDLD]=0. Reset type: SYSRSn
13-12	RESERVED	R	0h	Reserved
11-8	RESERVED	R-0	0h	Reserved
7	OSHTSYNC	R-0/W1S	0h	Oneshot sync bit 0: Writing a '0' has no effect. 1: Allow one sync pulse to propogate. Reset type: SYSRSn
6	OSHTSYNCMODE	R/W	0h	Oneshot sync enable bit 0: Oneshot sync mode disabled 1: Oneshot sync mode enabled Reset type: SYSRSn
5	RESERVED	R/W	0h	Reserved
4-0	RESERVED	R-0	0h	Reserved

26.17.2.3 EPWMSYNCINSEL Register (Offset = 3h) [Reset = 1h]

EPWMSYNCINSEL is shown in Figure 26-95 and described in Table 26-26.

Return to the Summary Table.

EPWMxSYNCIN Source Select Register

Figure 26-95. EPWMSYNCINSEL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			SEL				
R-0h			R/W-1h				

Table 26-26. EPWMSYNCINSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4-0	SEL	R/W	1h	These bits determine the source of the EPWMxSYNCI signal. 0x00 Disabled Other Values defined in the "ePWM SYNC Selection" table Reset type: SYSRSn

26.17.2.4 TBCTR Register (Offset = 4h) [Reset = 0h]

TBCTR is shown in Figure 26-96 and described in Table 26-27.

Return to the Summary Table.

Time Base Counter Register

Figure 26-96. TBCTR Register

15	14	13	12	11	10	9	8
TBCTR							
R/W-0h							
7	6	5	4	3	2	1	0
TBCTR							
R/W-0h							

Table 26-27. TBCTR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TBCTR	R/W	0h	Time Base Counter Register Reset type: SYSRSn

26.17.2.5 TBSTS Register (Offset = 5h) [Reset = 1h]

TBSTS is shown in Figure 26-97 and described in Table 26-28.

Return to the Summary Table.

Time Base Status Register

Figure 26-97. TBSTS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED					CTRMAX	SYNCI	CTRDIR
R-0-0h					R/W1C-0h	R/W1C-0h	R-1h

Table 26-28. TBSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R-0	0h	Reserved
2	CTRMAX	R/W1C	0h	Time-Base Counter Max Latched Status Bit 0: Reading a 0 indicates the time-base counter never reached its maximum value. Writing a 0 will have no effect. 1: Reading a 1 on this bit indicates that the time-base counter reached the max value 0xFFFF. Writing a 1 to this bit will clear the latched event. Reset type: SYSRSn
1	SYNCI	R/W1C	0h	Input Synchronization Latched Status Bit 0: Writing a 0 will have no effect. Reading a 0 indicates no external synchronization event has occurred. 1: Reading a 1 on this bit indicates that an external synchronization event has occurred (EPWMxSYNCI). Writing a 1 to this bit will clear the latched event. Reset type: SYSRSn
0	CTRDIR	R	1h	Time Base Counter Direction Status Bit 0: Time-Base Counter is currently counting down. 1: Time-Base Counter is currently counting up. Note: This bit is only valid when the counter is not frozen. Reset type: SYSRSn

26.17.2.6 EPWMSYNCOUTEN Register (Offset = 6h) [Reset = 1h]

EPWMSYNCOUTEN is shown in Figure 26-98 and described in Table 26-29.

Return to the Summary Table.

EPWMxSYNCOUT Source Enable Register

Figure 26-98. EPWMSYNCOUTEN Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	DCBEVT1EN	DCAEVT1EN	CMPDEN	CMPCEN	CMPBEN	ZEROEN	SWEN
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h

Table 26-29. EPWMSYNCOUTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	RESERVED	R	0h	Reserved
6	DCBEVT1EN	R/W	0h	This bit enables the DCBEVT1.sync event to set the EPWMxSYNCO signal. 0 Disabled 1 The EPWMxSYNCO signal is pulsed for one PWM clock period upon a DCBEVT1.sync event Reset type: SYSRSn
5	DCAEVT1EN	R/W	0h	This bit enables the DCAEVT1.sync event to set the EPWMxSYNCOUT signal. 0 Disabled 1 The EPWMxSYNCOUT signal is pulsed for one PWM clock period upon a DCAEVT1.sync event Reset type: SYSRSn
4	CMPDEN	R/W	0h	This bit enables the TBCTR = CMPD event to set the EPWMxSYNCO signal. 0 Disabled 1 The EPWMxSYNCO signal is pulsed for one PWM clock period upon a time-base counter equal to counter compare D event (TBCTR = CMPD) Reset type: SYSRSn
3	CMPCEN	R/W	0h	This bit enables the TBCTR = CMPC event to set the EPWMxSYNCO signal. 0 Disabled 1 The EPWMxSYNCO signal is pulsed for one PWM clock period upon a time-base counter equal to counter compare C event (TBCTR = CMPC) Reset type: SYSRSn
2	CMPBEN	R/W	0h	This bit enables the TBCTR = CMPB event to set the EPWMxSYNCO signal. 0 Disabled 1 The EPWMxSYNCO signal is pulsed for one PWM clock period upon a time-base counter equal to counter compare B event (TBCTR = CMPB) Reset type: SYSRSn
1	ZEROEN	R/W	0h	This bit enables the TBCTR = 0x0000 event to set the EPWMxSYNCOUT signal. 0 Disabled 1 The EPWMxSYNCOUT signal is pulsed for one PWM clock period upon the value of TBCTR changing to 0x0000 Reset type: SYSRSn

Table 26-29. EPWMSYNCOUTEN Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	SWEN	R/W	1h	This bit enables the TBCTL.SWFSYNC bit to set the EPWMxSYNCO signal. 0 Disabled 1 The EPWMxSYNCO signal is pulsed for one PWM clock period when the TBCTL.SWFSYNC bit is set Reset type: SYSRSn

26.17.2.7 TBCTL3 Register (Offset = 7h) [Reset = 0h]

TBCTL3 is shown in Figure 26-99 and described in Table 26-30.

Return to the Summary Table.

Time Base Control Register 3

Figure 26-99. TBCTL3 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							OSSFRCEN
R-0h							R/W-0h

Table 26-30. TBCTL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	OSSFRCEN	R/W	0h	This bit determines which bit sets the EPWMxSYNCOUT One Shot Latch. 0 TBCTL2[OSHTSYNC] sets the One Shot Latch 1 GLDCTL2[OSHTLD] sets the One Shot Latch Reset type: SYSRSn

26.17.2.8 CMPCTL Register (Offset = 8h) [Reset = 0h]

CMPCTL is shown in Figure 26-100 and described in Table 26-31.

Return to the Summary Table.

Counter Compare Control Register

Figure 26-100. CMPCTL Register

15	14	13	12	11	10	9	8
RESERVED		LOADBSYNC		LOADASYNC		SHDWBFULL	SHDWAFULL
R-0-0h		R/W-0h		R/W-0h		R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	SHDWBMODE	RESERVED	SHDWAMODE	LOADBMODE		LOADAMODE	
R-0-0h	R/W-0h	R-0-0h	R/W-0h	R/W-0h		R/W-0h	

Table 26-31. CMPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	RESERVED	R-0	0h	Reserved
13-12	LOADBSYNC	R/W	0h	Shadow to Active CMPB Register Load on SYNC event 00: Shadow to Active Load of CMPB:CMPBHR occurs according to LOADBMODE (bits 1,0) (same as legacy) 01: Shadow to Active Load of CMPB:CMPBHR occurs both according to LOADBMODE bits and when SYNC occurs 10: Shadow to Active Load of CMPB:CMPBHR occurs only when a SYNC is received 11: Reserved Note: This bit is valid only if CMPCTL[SHDWBMODE] = 0. Reset type: SYSRSn
11-10	LOADASYNC	R/W	0h	Shadow to Active CMPA Register Load on SYNC event 00: Shadow to Active Load of CMPA:CMPAHR occurs according to LOADAMODE (bits 1,0) (same as legacy) 01: Shadow to Active Load of CMPA:CMPAHR occurs both according to LOADAMODE bits and when SYNC occurs 10: Shadow to Active Load of CMPA:CMPAHR occurs only when a SYNC is received 11: Reserved Note: This bit is valid only if CMPCTL[SHDWAMODE] = 0. Reset type: SYSRSn
9	SHDWBFULL	R	0h	Counter-compare B (CMPB) Shadow Register Full Status Flag This bit self clears once a loadstrobe occurs. 0: CMPB shadow register not full yet 1: Indicates the CMPB shadow register is full a CPU write will overwrite current shadow value Reset type: SYSRSn
8	SHDWAFULL	R	0h	Counter-compare A (CMPA) Shadow Register Full Status Flag The flag bit is set when a 32-bit write to CMPA:CMPAHR register or a 16-bit write to CMPA register is made. A 16-bit write to CMPAHR register will not affect the flag. This bit self clears once a load-strobe occurs. 0: CMPA shadow register not full yet 1: Indicates the CMPA shadow register is full, a CPU write will overwrite the current shadow value Reset type: SYSRSn
7	RESERVED	R-0	0h	Reserved

Table 26-31. CMPCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	SHDWBMODE	R/W	0h	Counter-compare B (CMPB) Register Operating Mode 0: Shadow mode. Operates as a double buffer. All writes via the CPU access the shadow register 1: Immediate mode. Only the active compare B register is used. All writes and reads directly access the active register for immediate compare action Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4	SHDWAMODE	R/W	0h	Counter-compare A (CMPA) Register Operating Mode 0: Shadow mode. Operates as a double buffer. All writes via the CPU access the shadow register 1: Immediate mode. Only the active compare register is used. All writes and reads directly access the active register for immediate compare action Reset type: SYSRSn
3-2	LOADBMODE	R/W	0h	Active Counter-Compare B (CMPB) Load From Shadow Select Mode This bit has no effect in immediate mode (CMPCTL[SHDWBMODE] = 1). 00: Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01: Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10: Load on either CTR = Zero or CTR = PRD 11: Freeze (no loads possible) Reset type: SYSRSn
1-0	LOADAMODE	R/W	0h	Active Counter-Compare A (CMPA) Load From Shadow Select Mode This bit has no effect in immediate mode (CMPCTL[SHDWAMODE] = 1). 00: Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01: Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10: Load on either CTR = Zero or CTR = PRD 11: Freeze (no loads possible) Reset type: SYSRSn

26.17.2.9 CMPCTL2 Register (Offset = 9h) [Reset = 0h]

CMPCTL2 is shown in Figure 26-101 and described in Table 26-32.

Return to the Summary Table.

Counter Compare Control Register 2

Figure 26-101. CMPCTL2 Register

15	14	13	12	11	10	9	8
RESERVED		LOADDSYNC		LOADCSYNC		RESERVED	
R-0-0h		R/W-0h		R/W-0h		R-0-0h	
7	6	5	4	3	2	1	0
RESERVED	SHDWDMODE	RESERVED	SHDWCMODE	LOADDMODE		LOADCMODE	
R-0-0h	R/W-0h	R-0-0h	R/W-0h	R/W-0h		R/W-0h	

Table 26-32. CMPCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	RESERVED	R-0	0h	Reserved
13-12	LOADDSYNC	R/W	0h	Shadow to Active CMPD Register Load on SYNC event 00: Shadow to Active Load of CMPD occurs according to LOADDMODE 01: Shadow to Active Load of CMPD occurs both according to LOADDMODE bits and when SYNC occurs 10: Shadow to Active Load of CMPD occurs only when a SYNC is received 11: Reserved Note: This bit is valid only if CMPCTL2[SHDWDMODE] = 0. Reset type: SYSRSn
11-10	LOADCSYNC	R/W	0h	Shadow to Active CMPC Register Load on SYNC event 00: Shadow to Active Load of CMPC occurs according to LOADCMODE 01: Shadow to Active Load of CMPC occurs both according to LOADCMODE bits and when SYNC occurs 10: Shadow to Active Load of CMPC occurs only when a SYNC is received 11: Reserved Note: This bit is valid only if CMPCTL2[SHDWCMODE] = 0. Reset type: SYSRSn
9-7	RESERVED	R-0	0h	Reserved
6	SHDWDMODE	R/W	0h	Counter-Compare D Register Operating Mode 0: Shadow mode - operates as a double buffer. All writes via the CPU access Shadow register. 1: Immediate mode - only the Active compare register is used. All writes/reads via the CPU directly access the Active register for immediate Compare action. Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4	SHDWCMODE	R/W	0h	Counter-Compare C Register Operating Mode 0: Shadow mode - operates as a double buffer. All writes via the CPU access Shadow register. 1: Immediate mode - only the Active compare register is used. All writes/reads via the CPU directly access the Active register for immediate Compare action. Reset type: SYSRSn

Table 26-32. CMPCTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	LOADDMODE	R/W	0h	Active Counter-Compare D (CMPD) Load from Shadow Select Mode 00: Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01: Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10: Load on either CTR = Zero or CTR = PRD 11: Freeze (no loads possible) Note: Has no effect in Immediate mode. Reset type: SYSRSn
1-0	LOADCMODE	R/W	0h	Active Counter-Compare C (CMPC) Load from Shadow Select Mode 00: Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01: Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10: Load on either CTR = Zero or CTR = PRD 11: Freeze (no loads possible) Note: Has no effect in Immediate mode. Reset type: SYSRSn

26.17.2.10 DBCTL Register (Offset = Ch) [Reset = 0h]

DBCTL is shown in Figure 26-102 and described in Table 26-33.

Return to the Summary Table.

Dead-Band Generator Control Register

Figure 26-102. DBCTL Register

15	14	13	12	11	10	9	8
HALFCYCLE	DEDB_MODE	OUTSWAP		SHDWDBFED MODE	SHDWDBRED MODE	LOADFEDMODE	
R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	
7	6	5	4	3	2	1	0
LOADREDMODE		IN_MODE		POLSEL		OUT_MODE	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 26-33. DBCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	HALFCYCLE	R/W	0h	Half Cycle Clocking Enable Bit 0: Full cycle clocking enabled. The dead-band counters are clocked at the TBCLK rate. 1: Half cycle clocking enabled. The dead-band counters are clocked at TBCLK*2. Reset type: SYSRSn
14	DEDB_MODE	R/W	0h	Dead Band Dual-Edge B Mode Control (S8 switch) 0: Rising edge delay applied to InA/InB as selected by S4 switch (IN-MODE bits) on A signal path only. Falling edge delay applied to InA/InB as selected by S5 switch (INMODE bits) on B signal path only. 1: Rising edge delay and falling edge delay applied to source selected by S4 switch (INMODE bits) and output to B signal path only. Note: When this bit is set to 1, user should always either set OUT_MODE bits such that Apath = InA OR OUTSWAP bits such that OutA=Bpath otherwise, OutA will be invalid. Reset type: SYSRSn
13-12	OUTSWAP	R/W	0h	Dead Band Output Swap Control Bit 13 controls the S6 switch and bit 12 controls the S7 switch. 00: OutA and OutB signals are as defined by OUT-MODE bits. 01: OutA = A-path as defined by OUT-MODE bits. OutB = A-path as defined by OUT-MODE bits (rising edge delay or delay-bypassed A signal path). 10: OutA = B-path as defined by OUT-MODE bits (falling edge delay or delay-bypassed B signal path). OutB = B-path as defined by OUT-MODE bits. 11: OutA = B-path as defined by OUT-MODE bits (falling edge delay or delay-bypassed B signal path). OutB = A-path as defined by OUT-MODE bits (rising edge delay or delay-bypassed A signal path). Reset type: SYSRSn
11	SHDWDBFEDMODE	R/W	0h	FED Dead-Band Load Mode 0: Immediate mode. Only the active DBFED register is used. All writes/reads via the CPU directly access the active register for immediate "FED dead-band action." 1: Shadow mode. Operates as a double buffer. All writes via the CPU access Shadow register. Default at Reset is Immediate mode (for compatibility with legacy). Reset type: SYSRSn

Table 26-33. DBCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	SHDWDBREDMODE	R/W	0h	RED Dead-Band Load Mode 0: Immediate mode. Only the active DBRED register is used. All writes/reads via the CPU directly access the active register for immediate "RED dead-band action." 1: Shadow mode. Operates as a double buffer. All writes via the CPU access Shadow register. Default at Reset is Immediate mode (for compatibility with legacy). Reset type: SYSRSn
9-8	LOADFEDMODE	R/W	0h	Active DBFED Load from Shadow Select Mode 00: Load on Counter = 0 (CNT_eq) 01: Load on Counter = Period (PRD_eq) 10: Load on either Counter = 0, or Counter = Period 11: Freeze (no loads possible) Note: has no effect in Immediate mode. Reset type: SYSRSn
7-6	LOADREDMODE	R/W	0h	Active DBRED Load from Shadow Select Mode 00: Load on Counter = 0 (CNT_eq) 01: Load on Counter = Period (PRD_eq) 10: Load on either Counter = 0, or Counter = Period 11: Freeze (no loads possible) Note: has no effect in Immediate mode. Reset type: SYSRSn
5-4	IN_MODE	R/W	0h	Dead-Band Input Mode Control Bit 5 controls the S5 switch and bit 4 controls the S4 switch shown. This allows you to select the input source to the falling-edge and rising-edge delay. To produce classical dead-band waveforms the default is EPWMxA In is the source for both falling and rising-edge delays. 00: EPWMxA In (from the action-qualifier) is the source for both falling-edge and rising-edge delay. 01: EPWMxB In (from the action-qualifier) is the source for rising-edge delayed signal. EPWMxA In (from the action-qualifier) is the source for falling-edge delayed signal. 10: EPWMxA In (from the action-qualifier) is the source for rising-edge delayed signal. EPWMxB In (from the action-qualifier) is the source for falling-edge delayed signal. 11: EPWMxB In (from the action-qualifier) is the source for both rising-edge delay and falling-edge delayed signal. Reset type: SYSRSn
3-2	POLSEL	R/W	0h	Polarity Select Control Bit 3 controls the S3 switch and bit 2 controls the S2 switch. This allows you to selectively invert one of the delayed signals before it is sent out of the dead-band submodule. The following descriptions correspond to classical upper/lower switch control as found in one leg of a digital motor control inverter. These assume that DBCTL[OUT_MODE] = 1,1 and DBCTL[IN_MODE] = 0x0. Other enhanced modes are also possible, but not regarded as typical usage modes. 00: Active high (AH) mode. Neither EPWMxA nor EPWMxB is inverted (default). 01: Active low complementary (ALC) mode. EPWMxA is inverted. 10: Active high complementary (AHC). EPWMxB is inverted. 11: Active low (AL) mode. Both EPWMxA and EPWMxB are inverted. Reset type: SYSRSn

Table 26-33. DBCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	OUT_MODE	R/W	0h	Dead-Band Output Mode Control Bit 1 controls the S1 switch and bit 0 controls the S0 switch. 00: DBM is fully disabled or by-passed. In this mode the POLSEL and IN-MODE bits have no effect. 01: Apath = InA (delay is by-passed for A signal path) Bpath = FED (Falling Edge Delay in B signal path) 10: Apath = RED (Rising Edge Delay in A signal path) Bpath = InB (delay is by-passed for B signal path) 11: DBM is fully enabled (i.e. both RED and FED active) Reset type: SYSRSn

26.17.2.11 DBCTL2 Register (Offset = Dh) [Reset = 0h]

DBCTL2 is shown in Figure 26-103 and described in Table 26-34.

Return to the Summary Table.

Dead-Band Generator Control Register 2

Figure 26-103. DBCTL2 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED					SHDWDBCTLMODE	LOADDBCTLMODE	
R-0-0h					R/W-0h	R/W-0h	

Table 26-34. DBCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R-0	0h	Reserved
2	SHDWDBCTLMODE	R/W	0h	DBCTL Load Mode 0: Immediate mode - only the Active DBCTL register is used. All writes/reads via the CPU directly access the Active register. 1: Shadow mode - All writes and reads to bits [5:0] of the DBCTL register are shadowed. All other bits still access the active register. Reset type: SYSRSn
1-0	LOADDBCTLMODE	R/W	0h	Active DBCTL Load from Shadow Select Mode 00: Load on Counter = 0 (CNT_eq) 01: Load on Counter = Period (PRD_eq) 10: Load on either Counter = 0, or Counter = Period 11: Freeze (no loads possible) Note: has no effect in Immediate mode Reset type: SYSRSn

26.17.2.12 AQCTL Register (Offset = 10h) [Reset = 0h]

AQCTL is shown in Figure 26-104 and described in Table 26-35.

Return to the Summary Table.

Action Qualifier Control Register

Figure 26-104. AQCTL Register

15	14	13	12	11	10	9	8
RESERVED				LDAQBSYNC		LDAQASYNC	
R-0-0h				R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED	SHDWAQBMODE	RESERVED	SHDWAQAMODE	LDAQBMODE		LDAQAMODE	
R-0-0h	R/W-0h	R-0-0h	R/W-0h	R/W-0h		R/W-0h	

Table 26-35. AQCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R-0	0h	Reserved
11-10	LDAQBSYNC	R/W	0h	Shadow to Active AQCTLB Register Load on SYNC event 00: Shadow to Active Load of AQCTLB occurs according to LDAQBMODE 01: Shadow to Active Load of AQCTLB occurs both according to LDAQBMODE bits and when SYNC occurs. 10: Shadow to Active Load of AQCTLB occurs only when a SYNC is received. 11: Reserved Note: This bit is valid only if AQCTL[SHDWAQBMODE] = 1. Reset type: SYSRSn
9-8	LDAQASYNC	R/W	0h	Shadow to Active AQCTLA Register Load on SYNC event 00: Shadow to Active Load of AQCTLA occurs according to LDAQAMODE 01: Shadow to Active Load of AQCTLA occurs both according to LDAQAMODE bits and when SYNC occurs. 10: Shadow to Active Load of AQCTLA occurs only when a SYNC is received. 11: Reserved Note: This bit is valid only if AQCTL[SHDWAQAMODE] = 1. Reset type: SYSRSn
7	RESERVED	R-0	0h	Reserved
6	SHDWAQBMODE	R/W	0h	Action Qualifier B Register operating mode 1: Shadow mode - operates as a double buffer. All writes via the CPU access Shadow register. 0: Immediate mode - only the Active action qualifier register is used. All writes/reads via the CPU directly access the Active register. Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4	SHDWAQAMODE	R/W	0h	Action Qualifier A Register operating mode 1: Shadow mode - operates as a double buffer. All writes via the CPU access Shadow register. 0: Immediate mode - only the Active action qualifier register is used. All writes/reads via the CPU directly access the Active register. Reset type: SYSRSn

Table 26-35. AQCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	LDAQBMODE	R/W	0h	Active Action Qualifier B Load from Shadow Select Mode 00: Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01: Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10: Load on either CTR = Zero or CTR = PRD 11: Freeze (no loads possible) Note: has no effect in Immediate mode. Reset type: SYSRSn
1-0	LDAQAMODE	R/W	0h	Active Action Qualifier A Load from Shadow Select Mode 00: Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01: Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10: Load on either CTR = Zero or CTR = PRD 11: Freeze (no loads possible) Note: has no effect in Immediate mode. Reset type: SYSRSn

26.17.2.13 AQTSRCSEL Register (Offset = 11h) [Reset = 0h]

AQTSRCSEL is shown in Figure 26-105 and described in Table 26-36.

Return to the Summary Table.

Action Qualifier Trigger Event Source Select Register

Figure 26-105. AQTSRCSEL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
T2SEL				T1SEL			
R/W-0h				R/W-0h			

Table 26-36. AQTSRCSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R-0	0h	Reserved
7-4	T2SEL	R/W	0h	T2 Event Source Select Bits 0000: DCAEVT1 0001: DCAEVT2 0010: DCBEVT1 0011: DCBEVT2 0100: TZ1 0101: TZ2 0110: TZ3 0111: EPWMxSYNCI 1000: DCEVTFILT Others: Reserved Reset type: SYSRSn
3-0	T1SEL	R/W	0h	T1 Event Source Select Bits 0000: DCAEVT1 0001: DCAEVT2 0010: DCBEVT1 0011: DCBEVT2 0100: TZ1 0101: TZ2 0110: TZ3 0111: EPWMxSYNCI 1000: DCEVTFILT Others: Reserved Reset type: SYSRSn

26.17.2.14 PCCTL Register (Offset = 14h) [Reset = 0h]

PCCTL is shown in Figure 26-106 and described in Table 26-37.

Return to the Summary Table.

PWM Chopper Control Register

Figure 26-106. PCCTL Register

15	14	13	12	11	10	9	8
RESERVED					CHPDUTY		
R-0-0h					R/W-0h		
7	6	5	4	3	2	1	0
CHPFREQ			OSHTWTH				CHPEN
R/W-0h			R/W-0h				R/W-0h

Table 26-37. PCCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R-0	0h	Reserved
10-8	CHPDUTY	R/W	0h	Chopping Clock Duty Cycle 000: Duty = 1/8 (12.5%) 001: Duty = 2/8 (25.0%) 010: Duty = 3/8 (37.5%) 011: Duty = 4/8 (50.0%) 100: Duty = 5/8 (62.5%) 101: Duty = 6/8 (75.0%) 110: Duty = 7/8 (87.5%) 111: Reserved Reset type: SYSRSn
7-5	CHPFREQ	R/W	0h	Chopping Clock Frequency 000: Divide by 1 (no prescale, = 12.5 MHz at 100 MHz TBCLK) 001: Divide by 2 (6.25 MHz at 100 MHz TBCLK) 010: Divide by 3 (4.16 MHz at 100 MHz TBCLK) 011: Divide by 4 (3.12 MHz at 100 MHz TBCLK) 100: Divide by 5 (2.50 MHz at 100 MHz TBCLK) 101: Divide by 6 (2.08 MHz at 100 MHz TBCLK) 110: Divide by 7 (1.78 MHz at 100 MHz TBCLK) 111: Divide by 8 (1.56 MHz at 100 MHz TBCLK) Reset type: SYSRSn
4-1	OSHTWTH	R/W	0h	One-Shot Pulse Width 0000: 1 x EPWMCLK / 8 wide (= 80 ns at 100 MHz EPWMCLK) 0001: 2 x EPWMCLK / 8 wide (= 160 ns at 100 MHz EPWMCLK) 0010: 3 x EPWMCLK / 8 wide (= 240 ns at 100 MHz EPWMCLK) 0011: 4 x EPWMCLK / 8 wide (= 320 ns at 100 MHz EPWMCLK) 0100: 5 x EPWMCLK / 8 wide (= 400 ns at 100 MHz EPWMCLK) 0101: 6 x EPWMCLK / 8 wide (= 480 ns at 100 MHz EPWMCLK) 0110: 7 x EPWMCLK / 8 wide (= 560 ns at 100 MHz EPWMCLK) 0111: 8 x EPWMCLK / 8 wide (= 640 ns at 100 MHz EPWMCLK) 1000: 9 x EPWMCLK / 8 wide (= 720 ns at 100 MHz EPWMCLK) 1001: 10 x EPWMCLK / 8 wide (= 800 ns at 100 MHz EPWMCLK) 1010: 11 x EPWMCLK / 8 wide (= 880 ns at 100 MHz EPWMCLK) 1011: 12 x EPWMCLK / 8 wide (= 960 ns at 100 MHz EPWMCLK) 1100: 13 x EPWMCLK / 8 wide (= 1040 ns at 100 MHz EPWMCLK) 1101: 14 x EPWMCLK / 8 wide (= 1120 ns at 100 MHz EPWMCLK) 1110: 15 x EPWMCLK / 8 wide (= 1200 ns at 100 MHz EPWMCLK) 1111: 16 x EPWMCLK / 8 wide (= 1280 ns at 100 MHz EPWMCLK) Reset type: SYSRSn
0	CHPEN	R/W	0h	PWM-Chopping Enable 0: Disable (bypass) PWM chopping function 1: Enable chopping function Reset type: SYSRSn

26.17.2.15 VCAPCTL Register (Offset = 18h) [Reset = 0h]

VCAPCTL is shown in Figure 26-107 and described in Table 26-38.

Return to the Summary Table.

Valley Capture Control Register

Figure 26-107. VCAPCTL Register

15	14	13	12	11	10	9	8
RESERVED					EDGEFILTDLYSEL	VDELAYDIV	
R-0-0h					R/W-0h	R/W-0h	
7	6	5	4	3	2	1	0
VDELAYDIV	RESERVED			TRIGSEL		VCAPSTART	VCAPE
R/W-0h	R-0-0h			R/W-0h		R-0/W1S-0h	R/W-0h

Table 26-38. VCAPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R-0	0h	Reserved
10	EDGEFILTDLYSEL	R/W	0h	Valley Switching Mode Delay Selection 0: No delay applied to the edge filter output 1: HWDELAYVAL delay applied to the edge filter output Reset type: SYSRSn
9-7	VDELAYDIV	R/W	0h	Valley Delay Mode Divide Enable 000: HWVDELVAL = SWVDELVAL 001: HWVDELVAL = VCNTVAL+SWVDELVAL 010: HWVDELVAL = VCNTVAL>>1+SWVDELVAL 011: HWVDELVAL = VCNTVAL>>2+SWVDELVAL 100: HWVDELVAL = VCNTVAL>>4+SWVDELVAL Note: Delay value between the consecutive edge captures can optionally be divided by using these bits. Reset type: SYSRSn
6-5	RESERVED	R-0	0h	Reserved
4-2	TRIGSEL	R/W	0h	Status of Numbered of Captured Events 000: Capture sequence is triggered by software via writes to VCAPCTL[VCAPSTART]. 001: Capture sequence is triggered by CNT_zero event. 010: Capture sequence is triggered by PRD_eq event. 011: Capture sequence is triggered by CNT_zero or PRD_eq event. 100: Capture sequence is triggered by DCAEVT1 event. 101: Capture sequence is triggered by DCAEVT2 event. 110: Capture sequence is triggered by DCBEVT1 event. 111: Capture sequence is triggered by DCBEVT2 event. Note: Valley capture sequence triggered by the selected event in this register field. Once the chosen event occurs the capture sequence is armed. Event captures occur based of the event chosen in DCFCTL[SRCSEL] register. Note: Same event may not be chosen in both DCFCTL[SRCSEL] and VCAPCTL[TRIGSEL] registers. Note: Once the chosen event in VCAPCTL[TRIGSEL] occurs, irrespective of the current capture status, capture sequence is retriggered. Reset type: SYSRSn
1	VCAPSTART	R-0/W1S	0h	Valley Capture Start 0: Writing a 0 has no effect 1: Trigger the capture sequence once if VCAPCTL[TRIGSEL]=0x0 Note: This bit is used to start valley capture sequence through software. VCAPCTL[TRIGSEL] has to be chosen for software trigger for this bit to have any effect. Writing of 1 will result in one capture sequence trigger. Reset type: SYSRSn

Table 26-38. VCAPCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	VCAPE	R/W	0h	Valley Capture Enable/Disable 0: Disabled 1: Enabled Reset type: SYSRSn

26.17.2.16 VCNTCFG Register (Offset = 19h) [Reset = 0h]

VCNTCFG is shown in Figure 26-108 and described in Table 26-39.

Return to the Summary Table.

Valley Counter Config Register

Figure 26-108. VCNTCFG Register

15	14	13	12	11	10	9	8
STOPEDGESTS	RESERVED			STOPEDGE			
R-0h	R-0-0h			R/W-0h			
7	6	5	4	3	2	1	0
STARTEDGESTS	RESERVED			STARTEDGE			
R-0h	R-0-0h			R/W-0h			

Table 26-39. VCNTCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
15	STOPEDGESTS	R	0h	Stop Edge Status Bit 0: Stop edge has not occurred 1: Stop edge occurred Note: This bit is set only after the trigger sequence is armed (upon occurrence of trigger pulse selected through VCAPCTL[TRIGSEL]) and STOPEDGE occurs. Note:This bit is reset by the occurrence of the trigger pulse selected through VCAPCTL[TRIGSEL] Reset type: SYSRSn
14-12	RESERVED	R-0	0h	Reserved
11-8	STOPEDGE	R/W	0h	Counter Stop Edge Selection Once the counter operation is armed, upon occurrence of trigger pulse selected through VCAPCTL[TRIGSEL] pulse - valley counter would stop counting upon the occurrence of chosen number of events thorough this bit field. Stop counting on occurrence of: 0000: Do not stop 0001 1st edge 0010: 2nd edge 0011: 3rd edge ... 1,1,1,1: 15th edge Reset type: SYSRSn
7	STARTEDGESTS	R	0h	Start Edge Status Bit 0: Start edge has not occurred 1: Start edge occurred Note: This bit is set only after the trigger sequence is armed (upon occurrence of trigger pulse selected through VCAPCTL[TRIGSEL]) and STARTEDGE occurs. Note:This bit is reset by the occurrence of the trigger pulse selected through VCAPCTL[TRIGSEL] Reset type: SYSRSn
6-4	RESERVED	R-0	0h	Reserved

Table 26-39. VCNTCFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	STARTEDGE	R/W	0h	Counter Start Edge Selection Once the counter operation is armed, upon occurrence of trigger pulse selected through VCAPCTL[TRIGSEL] pulse - valley counter would start counting upon the occurrence of chosen number of events thorough this bit field. Start counting on occurrence of 0000: Do not start 0001: 1st edge 0010: 2nd edge 0011: 3rd edge ... 1111: 15th edge Reset type: SYSRSn

26.17.2.17 HRCNFG Register (Offset = 20h) [Reset = 0h]

HRCNFG is shown in Figure 26-109 and described in Table 26-40.

Return to the Summary Table.

HRPWM Configuration Register
This register is only accessible on EPWM modules with HRPWM capabilities.

Figure 26-109. HRCNFG Register

15	14	13	12	11	10	9	8
RESERVED		RESERVED	HRLOADB		CTLMODEB	EDGMODEB	
R/W-0h		R-0-0h	R/W-0h		R/W-0h	R/W-0h	
7	6	5	4	3	2	1	0
SWAPAB	AUTOCONV	SELOUTB	HRLOAD		CTLMODE	EDGMODE	
R/W-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	

Table 26-40. HRCNFG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	RESERVED	R/W	0h	Reserved
13	RESERVED	R-0	0h	Reserved
12-11	HRLOADB	R/W	0h	Shadow Mode Bit Selects the time event that loads the CMPBHR shadow value into the active register. 00: Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01: Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10: Load on either CTR = Zero or CTR = PRD 11: Reserved Reset type: SYSRSn
10	CTLMODEB	R/W	0h	Control Mode Bits Selects the register (CMP/TBPRD or TBPHS) that controls the MEP: 0: CMPBHR(8) or TBPRDHR(8) Register controls the edge position (i.e., this is duty or period control mode). (Default on Reset) 1: TBPHSHR(8) Register controls the edge position (i.e., this is phase control mode). Reset type: SYSRSn
9-8	EDGMODEB	R/W	0h	Edge Mode Bits Selects the edge of the PWM that is controlled by the micro-edge position (MEP) logic: 00: HRPWM capability is disabled (default on reset) 01: MEP control of rising edge (CMPBHR) 10: MEP control of falling edge (CMPBHR) 11: MEP control of both edges (TBPHSHR or TBPRDHR) Reset type: SYSRSn
7	SWAPAB	R/W	0h	Swap ePWM A & B Output Signals This bit enables the swapping of the A & B signal outputs. The selection is as follows: 0: ePWMxA and ePWMxB outputs are unchanged. 1: ePWMxA signal appears on ePWMxB output and ePWMxB signal appears on ePWMxA output. Reset type: SYSRSn

Table 26-40. HRCNFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	AUTOCONV	R/W	0h	Auto Convert Delay Line Value Selects whether the fractional duty cycle/period/phase in the CMPAHR/TBPRDHR/TBPHSHR register is automatically scaled by the MEP scale factor in the HRMSTEP register or manually scaled by calculations in application software. The SFO library function automatically updates the HRMSTEP register with the appropriate MEP scale factor. 0: Automatic HRMSTEP scaling is disabled. 1: Automatic HRMSTEP scaling is enabled. If application software is manually scaling the fractional duty cycle, or phase (i.e. software sets CMPAHR = (fraction(PWMduty * PWMperiod) * MEP Scale Factor)<<8 + 0x080 for duty cycle), then this mode must be disabled. Reset type: SYSRSn
5	SELOUTB	R/W	0h	EPWMxB Output Select Bit This bit selects which signal is output on the ePWMxB channel output. The inversion will take the high resolution mode into account and the inverted signal will contain any high resolution modification. The inversion takes place as the last step in modifying the ePWMxB signal. 0: ePWMxB output is normal. 1: ePWMxB output is inverted version of ePWMxA signal. Reset type: SYSRSn
4-3	HRLOAD	R/W	0h	Shadow Mode Bit Selects the time event that loads the CMPAHR shadow value into the active register. 00: Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01: Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10: Load on either CTR = Zero or CTR = PRD 11: Reserved Reset type: SYSRSn
2	CTLMODE	R/W	0h	Control Mode Bits Selects the register (CMP/TBPRD or TBPHS) that controls the MEP: 0: CMPAHR(8) or TBPRDHR(8) Register controls the edge position (i.e., this is duty or period control mode). (Default on Reset) 1: TBPHSHR(8) Register controls the edge position (i.e., this is phase control mode). Reset type: SYSRSn
1-0	EDGMODE	R/W	0h	Edge Mode Bits Selects the edge of the PWM that is controlled by the micro-edge position (MEP) logic: 00: HRPWM capability is disabled (default on reset) 01: MEP control of rising edge (CMPAHR) 10: MEP control of falling edge (CMPAHR) 11: MEP control of both edges (TBPHSHR or TBPRDHR) Reset type: SYSRSn

26.17.2.18 HRPWR Register (Offset = 21h) [Reset = 0h]

HRPWR is shown in Figure 26-110 and described in Table 26-41.

Return to the Summary Table.

HRPWM Power Register
This register is only accessible on EPWM modules with HRPWM capabilities.

Figure 26-110. HRPWR Register

15	14	13	12	11	10	9	8
CALPWRON	RESERVED					RESERVED	
R/W-0h	R-0-0h					R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	
R/W-0h		R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	

Table 26-41. HRPWR Register Field Descriptions

Bit	Field	Type	Reset	Description
15	CALPWRON	R/W	0h	MEP Calibration Power Bits (only available on ePWM1) 0: Disables MEP calibration logic in the HRPWM and reduces power consumption. 1: Enables MEP calibration logic Reset type: SYSRSn
14-10	RESERVED	R-0	0h	Reserved
9-6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1-0	RESERVED	R/W	0h	Reserved

26.17.2.19 HRMSTEP Register (Offset = 26h) [Reset = 0h]

HRMSTEP is shown in Figure 26-111 and described in Table 26-42.

Return to the Summary Table.

HRPWM MEP Step Register
This register is only accessible on EPWM modules with HRPWM capabilities. Only 16 bit accesses are allowed for this register. Debugger access in 32 bit mode may display incorrect values.

Figure 26-111. HRMSTEP Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
HRMSTEP							
R/W-0h							

Table 26-42. HRMSTEP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R-0	0h	Reserved
7-0	HRMSTEP	R/W	0h	High Resolution MEP Step When auto-conversion is enabled (HRCNFG[AUTOCONV] = 1), This 8-bit field contains the MEP_ScaleFactor (number of MEP steps per coarse steps) used by the hardware to automatically convert the value in the CMPAHR, CMPBHR, DBFEDHR, DBREDHR , TBPHSHR, or TBPRDHR register to a scaled micro-edge delay on the high-resolution ePWM output. The value in this register is written by the SFO calibration software at the end of each calibration run. Reset type: SYSRSn

26.17.2.20 HRCNFG2 Register (Offset = 27h) [Reset = 0h]

HRCNFG2 is shown in Figure 26-112 and described in Table 26-43.

Return to the Summary Table.

HRPWM Configuration 2 Register
This register is only accessible on EPWM modules with HRPWM capabilities. Only 16 bit accesses are allowed for this register. Debugger access in 32 bit mode may display incorrect values.

Figure 26-112. HRCNFG2 Register

15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED					
R/W-0h	R-0/W1S-0h	R-0-0h					
7	6	5	4	3	2	1	0
RESERVED		CTLMODEDBFED		CTLMODEDBRED		EDGMODEDB	
R-0-0h		R/W-0h		R/W-0h		R/W-0h	

Table 26-43. HRCNFG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R-0/W1S	0h	Reserved
13-6	RESERVED	R-0	0h	Reserved
5-4	CTLMODEDBFED	R/W	0h	Shadow Mode Bit - selection should match DBCTL[LOADFEDMODE] Selects the time event that loads the DBFEDHR shadow value into the active register. 00 Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01 Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10 Load on either CTR = Zero or CTR = PRD 11 Reserved Reset type: SYSRSn
3-2	CTLMODEDBRED	R/W	0h	Shadow Mode Bit - selection should match DBCTL[LOADREDMODE] Selects the time event that loads the DBREDHR shadow value into the active register. 00 Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01 Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10 Load on either CTR = Zero or CTR = PRD 11 Reserved Reset type: SYSRSn
1-0	EDGMODEDB	R/W	0h	Edge Mode Bits Selects the edge of the PWM that is controlled by the micro-edge position (MEP) logic: 00 HRPWM capability is disabled (default on reset) 01 MEP control of rising edge (DBREDHR) 10 MEP control of falling edge (DBFEDHR) 11 MEP control of both edges (rising edge of DBREDHR or falling edge of DBFEDHR) Reset type: SYSRSn

26.17.2.21 HRPCTL Register (Offset = 2Dh) [Reset = 0h]

HRPCTL is shown in Figure 26-113 and described in Table 26-44.

Return to the Summary Table.

High Resolution Period Control Register
This register is only accessible on EPWM modules with HRPWM capabilities.

Figure 26-113. HRPCTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED	PWMSYNCSELX			RESERVED	TBPHSHRLOADE	PWMSYNCSEL	HRPE
R-0-0h	R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 26-44. HRPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R-0	0h	Reserved
6-4	PWMSYNCSELX	R/W	0h	Extended selection bits for EPWMSYNCPER 000: EPWMSYNCPER is defined by PWMSYNCSEL - > default condition (compatible with previous EPWM versions) 001: Reserved 010: Reserved 011: Reserved 100: CTR = CMPC, Count direction Up 101: CTR = CMPC, Count direction Down 110: CTR = CMPD, Count direction Up 111: CTR = CMPD, Count direction Down Reset type: SYSRSn
3	RESERVED	R/W	0h	Reserved
2	TBPHSHRLOADE	R/W	0h	TBPHSHR Load Enable This bit allows you to synchronize ePWM modules with a high-resolution phase on a SYNCIN, TBCTL[SWFSYNC] or digital compare event. This allows for multiple ePWM modules operating at the same frequency to be phase aligned with high-resolution. 0: Disables synchronization of high-resolution phase on a SYNCIN, TBCTL[SWFSYNC] or digital compare event: 1: Synchronize the high-resolution phase on a SYNCIN, TBCTL[SWFSYNC] or digital comparator synchronization event. The phase is synchronized using the contents of the high-resolution phase TBPHSHR register. The TBCTL[PHSEN] bit which enables the loading of the TBCTR register with TBPHS register value on a SYNCIN or TBCTL[SWFSYNC] event works independently. However, users need to enable this bit also if they want to control phase in conjunction with the high-resolution period feature. This bit and the TBCTL[PHSEN] bit must be set to 1 when high-resolution period is enabled for up-down count mode even if TBPHSHR = 0x0000. This bit does not need to be set when only high-resolution duty is enabled. Reset type: SYSRSn
1	PWMSYNCSEL	R/W	0h	PWMSYNC Source Select Bit: This bit selects the source for the EPWMSYNCPER signal that goes to the CMPSS and GPDAC: 0 CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 1 CTR = zero: Time-base counter equal to zero (TBCTR = 0x00) Reset type: SYSRSn

Table 26-44. HRPCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	HRPE	R/W	0h	High Resolution Period Enable Bit 0: High resolution period feature disabled. In this mode the ePWM behaves as a Type 4 ePWM. 1: High resolution period enabled. In this mode the HRPWM module can control high-resolution of both the duty and frequency. When high-resolution period is enabled, TBCTL[CTRMODE] = 0,1 (down-count mode) is not supported. Reset type: SYSRSn

26.17.2.22 TRREM Register (Offset = 2Eh) [Reset = 0h]

TRREM is shown in Figure 26-114 and described in Table 26-45.

Return to the Summary Table.

HRPWM High Resolution Remainder Register
This register is only accessible on EPWM modules with HRPWM capabilities.

Figure 26-114. TRREM Register

15	14	13	12	11	10	9	8
RESERVED					TRREM		
R-0-0h					R/W-0h		
7	6	5	4	3	2	1	0
TRREM							
R/W-0h							

Table 26-45. TRREM Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R-0	0h	Reserved
10-0	TRREM	R/W	0h	HRPWM Remainder Bits: This 11-bit value keeps track of the remainder portion of the HRPWM algorithm calculations. This value keeps track of the remainder portion of the HRPWM hardware calculations. Notes: 1. The lower 8-bits of the TRREM register can be automatically initialized with the TBPHSHR value on a SYNCIN or TBCTL[SWFSYNC] event or DC event (if enabled). The user can also write a value with the CPU. 2. Priority of TRREM register updates: Sync (software or hardware) TBPHSHR copied to TRREM : Highest Priority HRPWM Hardware (updates TRREM register): Next priority CPU Write To TRREM Register: Lowest Priority 3. Bit 10 of TRREM register is not used in asymmetrical mode. This bit can be forced to zero. TRREM will be initialized to 0x0 and 0x100 in Up and Up-down modes respectively. Asymmetrical Mode: TRREM[7:0] = TBPHSHR[15:8] TRREM[10,9,8] = 0,0,0 Symmetrical Mode: TRREM[7:0] = TBPHSHR[15:8] TRREM[10,9,8] = 0,0,1 Reset type: SYSRSn

26.17.2.23 GLDCTL Register (Offset = 34h) [Reset = 0h]

GLDCTL is shown in Figure 26-115 and described in Table 26-46.

Return to the Summary Table.

Global PWM Load Control Register

Figure 26-115. GLDCTL Register

15	14	13	12	11	10	9	8
RESERVED			GLDCNT			GLDPRD	
R-0-0h			R-0h			R/W-0h	
7	6	5	4	3	2	1	0
GLDPRD	RESERVED	OSHTMODE	GLDMODE				GLD
R/W-0h	R-0-0h	R/W-0h	R/W-0h				R/W-0h

Table 26-46. GLDCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R-0	0h	Reserved
12-10	GLDCNT	R	0h	Global Load Strobe Counter Register These bits indicate how many selected events have occurred: 000: No events 001: 1 event 010: 2 events 011: 3 events 100: 4 events 101: 5 events 110: 6 events 111: 7 events Reset type: SYSRSn
9-7	GLDPRD	R/W	0h	Global Load Strobe Period Select Register These bits select how many selected events need to occur before a load strobe is generated 000: Disable counter 001: Generate strobe on GLDCNT = 001 (1st event) 010: Generate strobe on GLDCNT = 010 (2nd event) 011: Generate strobe on GLDCNT = 011 (3rd event) 100: Generate strobe on GLDCNT = 011 (4th event) 101: Generate strobe on GLDCNT = 001 (5th event) 110: Generate strobe on GLDCNT = 010 (6th event) 111: Generate strobe on GLDCNT = 011 (7th event) Reset type: SYSRSn
6	RESERVED	R-0	0h	Reserved
5	OSHTMODE	R/W	0h	One Shot Load Mode Control Bit 0: One shot load mode is disabled and shadow to active loading happens continuously on all the chosen load strobes. 1: One shot mode is active. All load strobes are blocked until GLDCTL2[OSHTLD] is written with 1. Note: One Shot mode can only be used with global shadow to active load mode enabled (GLDCTL[GLD]=1) Reset type: SYSRSn

Table 26-46. GLDCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-1	GLDMODE	R/W	0h	Global Load Pulse selection for Shadow to Active Mode Reloads 0000: Load on Counter = 0 (CNT_ZRO) 0001: Load on Counter = Period (PRD_EQ) 0010: Load on either Counter = 0, or Counter = Period 0011: Load on SYNCEVT - this is logical OR of DCAEVT1.sync, DCBEVT1.sync, EPWMxSYNCI and TBCTL[SWFSYNC] 0100: Load on SYNCEVT or CNT_ZRO 0101: Load on SYNCEVT or PRD_EQ 0110: Load on SYNCEVT or CNT_ZRO or PRD_EQ 1000: Reserved ... 1110: Reserved 1111: Load on GLDCTL2[GFRCLD] write Reset type: SYSRSn
0	GLD	R/W	0h	Global Shadow to Active Load Event Control 0: Shadow to active reload for all shadowed registers happens as per the individual reload control bits specified (Compatible with previous EPWM versions). 1: When set, all the shadow to active reload events are defined by GLDMODE bits in GLDCTL register. All the shadow registers use same reload pulse from shadow to active reloading. Individual LOADMODE bits are ignored. Reset type: SYSRSn

26.17.2.24 GLDCFG Register (Offset = 35h) [Reset = 0h]

GLDCFG is shown in Figure 26-116 and described in Table 26-47.

Return to the Summary Table.

Global PWM Load Config Register

Figure 26-116. GLDCFG Register

15	14	13	12	11	10	9	8
RESERVED					AQCSFRC	AQCTLB_AQCTLB2	AQCTLA_AQCTLA2
R-0-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DBCTL	DBFED_DBFEDHR	DBRED_DBREDHR	CMPD	CMPC	CMPB_CMPBHR	CMPA_CMPAHR	TBPRD_TBPRDHR
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 26-47. GLDCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R-0	0h	Reserved
10	AQCSFRC	R/W	0h	Global load event configuration for AQCSFRC 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn
9	AQCTLB_AQCTLB2	R/W	0h	Global load event configuration for AQCTLB_AQCTLB2 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn
8	AQCTLA_AQCTLA2	R/W	0h	Global load event configuration for AQCTLA_AQCTLA2 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn
7	DBCTL	R/W	0h	Global load event configuration for DBCTL 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn
6	DBFED_DBFEDHR	R/W	0h	Global load event configuration for DBFED_DBFEDHR 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn
5	DBRED_DBREDHR	R/W	0h	Global load event configuration for DBRED_DBREDHR 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn

Table 26-47. GLDCFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	CMPD	R/W	0h	Global load event configuration for CMPD 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn
3	CMPC	R/W	0h	Global load event configuration for CMPC 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn
2	CMPB_CMPBHR	R/W	0h	Global load event configuration for CMPB_CMPBHR 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn
1	CMPA_CMPAHR	R/W	0h	Global load event configuration for CMPA_CMPAHR 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn
0	TBPRD_TBPRDHR	R/W	0h	Global load event configuration for TBPRD_TBPRDHR 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn

26.17.2.25 EPWMXLINK Register (Offset = 38h) [Reset = X]

EPWMXLINK is shown in Figure 26-117 and described in Table 26-48.

Return to the Summary Table.

EPWMx Link Register
This register controls which EPWMs are linked to other EPWM modules. The default reset value will vary for each module. The reset value will link each EPWM module to itself to prevent unintentional linking of modules.

Figure 26-117. EPWMXLINK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GLDCTL2LINK				RESERVED								CMPDLINK			
R/W-X				R-0-0h								R/W-X			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPCLINK				CMPBLINK				CMPALINK				TBPRDLINK			
R/W-X				R/W-X				R/W-X				R/W-X			

Table 26-48. EPWMXLINK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GLDCTL2LINK	R/W	X	GLDCTL2 Link Bits Writes to the GLDCTL2 registers in the ePWM module selected by the following bit selections results in a simultaneous write to the current ePWM module's GLDCTL2 registers. 0000: ePWM1 0001: ePWM2 ... Up to the last instance of ePWM. All others are reserved. Reset type: SYSRSn
27-20	RESERVED	R-0	0h	Reserved
19-16	CMPDLINK	R/W	X	CMPD Link Bits Writes to the CMPD registers in the ePWM module selected by the following bit selections results in a simultaneous write to the current ePWM module's CMPD registers. 0000: ePWM1 0001: ePWM2 ... Up to the last instance of ePWM. All others are reserved. Reset type: SYSRSn
15-12	CMPCLINK	R/W	X	CMPC Link Bits Writes to the CMPC registers in the ePWM module selected by the following bit selections results in a simultaneous write to the current ePWM module's CMPC registers. 0000: ePWM1 0001: ePWM2 ... Up to the last instance of ePWM. All others are reserved. Reset type: SYSRSn
11-8	CMPBLINK	R/W	X	CMPB_CMPBHR Link Bits Writes to the CMPB_CMPBHR registers in the ePWM module selected by the following bit selections results in a simultaneous write to the current ePWM module's CMPB_CMPBHR registers. 0000: ePWM1 0001: ePWM2 ... Up to the last instance of ePWM. All others are reserved. Reset type: SYSRSn

Table 26-48. EPWMXLINK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-4	CMPALINK	R/W	X	CMPA_CMPAHR Link Bits Writes to the CMPA_CMPAHR registers in the ePWM module selected by the following bit selections results in a simultaneous write to the current ePWM module's CMPA_CMPAHR registers. 0000: ePWM1 0001: ePWM2 ... Up to the last instance of ePWM. All others are reserved. Reset type: SYSRSn
3-0	TBPRDLINK	R/W	X	TBPRD_TBPRDHR Link Bits Writes to the TBPRD:TBPRDHR registers in the ePWM module selected by the following bit selections results in a simultaneous write to the current ePWM module's TBPRD_TBPRDHR registers. 0000: ePWM1 0001: ePWM2 ... Up to the last instance of ePWM. All others are reserved. Reset type: SYSRSn

26.17.2.26 AQCTLA Register (Offset = 40h) [Reset = 0h]

AQCTLA is shown in Figure 26-118 and described in Table 26-49.

Return to the Summary Table.

Action Qualifier Control Register For Output A

Figure 26-118. AQCTLA Register

15	14	13	12	11	10	9	8
RESERVED				CBD		CBU	
R-0-0h				R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
CAD		CAU		PRD		ZRO	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 26-49. AQCTLA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R-0	0h	Reserved
11-10	CBD	R/W	0h	Action When TBCTR = CMPB on Down Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
9-8	CBU	R/W	0h	Action When TBCTR = CMPB on Up Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
7-6	CAD	R/W	0h	Action When TBCTR = CMPA on Down Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
5-4	CAU	R/W	0h	Action When TBCTR = CMPA on Up Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn

Table 26-49. AQCTLA Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	PRD	R/W	0h	Action When TBCTR = TBPRD Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
1-0	ZRO	R/W	0h	Action When TBCTR = 0 Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn

26.17.2.27 AQCTLA2 Register (Offset = 41h) [Reset = 0h]

AQCTLA2 is shown in Figure 26-119 and described in Table 26-50.

Return to the Summary Table.

Additional Action Qualifier Control Register For Output A

Figure 26-119. AQCTLA2 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
T2D		T2U		T1D		T1U	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 26-50. AQCTLA2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R-0	0h	Reserved
7-6	T2D	R/W	0h	Action when event occurs on T2 in DOWN-Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
5-4	T2U	R/W	0h	Action when event occurs on T2 in UP-Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
3-2	T1D	R/W	0h	Action when event occurs on T1 in DOWN-Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
1-0	T1U	R/W	0h	Action when event occurs on T1 in UP-Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn

26.17.2.28 AQCTLB Register (Offset = 42h) [Reset = 0h]

AQCTLB is shown in Figure 26-120 and described in Table 26-51.

Return to the Summary Table.

Action Qualifier Control Register For Output B

Figure 26-120. AQCTLB Register

15	14	13	12	11	10	9	8
RESERVED				CBD		CBU	
R-0-0h				R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
CAD		CAU		PRD		ZRO	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 26-51. AQCTLB Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R-0	0h	Reserved
11-10	CBD	R/W	0h	Action When TBCTR = CMPB on Down Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
9-8	CBU	R/W	0h	Action When TBCTR = CMPB on Up Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
7-6	CAD	R/W	0h	Action When TBCTR = CMPA on Down Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
5-4	CAU	R/W	0h	Action When TBCTR = CMPA on Up Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn

Table 26-51. AQCTLB Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	PRD	R/W	0h	Action When TBCTR = TBPRD Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
1-0	ZRO	R/W	0h	Action When TBCTR = 0 Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn

26.17.2.29 AQCTLB2 Register (Offset = 43h) [Reset = 0h]

AQCTLB2 is shown in Figure 26-121 and described in Table 26-52.

Return to the Summary Table.

Additional Action Qualifier Control Register For Output B

Figure 26-121. AQCTLB2 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
T2D		T2U		T1D		T1U	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 26-52. AQCTLB2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R-0	0h	Reserved
7-6	T2D	R/W	0h	Action when event occurs on T2 in DOWN-Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
5-4	T2U	R/W	0h	Action when event occurs on T2 in UP-Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
3-2	T1D	R/W	0h	Action when event occurs on T1 in DOWN-Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
1-0	T1U	R/W	0h	Action when event occurs on T1 in UP-Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn

26.17.2.30 AQSFRC Register (Offset = 47h) [Reset = 0h]

AQSFRC is shown in Figure 26-122 and described in Table 26-53.

Return to the Summary Table.

Action Qualifier Software Force Register

Figure 26-122. AQSFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RLDCSF		OTSFB	ACTSFB		OTSFA	ACTSFA	
R/W-0h		R-0/W1S-0h	R/W-0h		R-0/W1S-0h	R/W-0h	

Table 26-53. AQSFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R-0	0h	Reserved
7-6	RLDCSF	R/W	0h	AQCSFRC Active Register Reload From Shadow Options 00: Load on time-base counter equals zero 01: Load on time-base counter equals period 10: Load on time-base counter equals zero or counter equals period 11: Load immediately (the active register is directly accessed by the CPU and is not loaded from the shadow register). Reset type: SYSRSn
5	OTSFB	R-0/W1S	0h	One-Time Software Forced Event on Output B 0: Writing a 0 (zero) has no effect. Always reads back a 0. This bit is auto cleared once a write to this register is complete (i.e., a forced event is initiated.). This is a one-shot forced event. It can be overridden by another subsequent event on output B. 1: Initiates a single software forced event Reset type: SYSRSn
4-3	ACTSFB	R/W	0h	Action When One-Time Software Force B is Invoked 00: Does nothing (action disabled) 01: Clear (low) 10: Set (high) 11: Toggle (Low -> High, High -> Low) Note: This action is not qualified by counter direction (CNT_dir) Reset type: SYSRSn
2	OTSFA	R-0/W1S	0h	One-Time Software Forced Event on Output A 0: Writing a 0 (zero) has no effect. Always reads back a 0. This bit is auto cleared once a write to this register is complete (i.e., a forced event is initiated). This is a one-shot forced event. It can be overridden by another subsequent event on output A. 1: Initiates a single software forced event Reset type: SYSRSn
1-0	ACTSFA	R/W	0h	Action When One-Time Software Force A Is Invoked 00: Does nothing (action disabled) 01: Clear (low) 10: Set (high) 11: Toggle (Low -> High, High -> Low) Note: This action is not qualified by counter direction (CNT_dir) Reset type: SYSRSn

26.17.2.31 AQCSFRC Register (Offset = 49h) [Reset = 0h]

AQCSFRC is shown in Figure 26-123 and described in Table 26-54.

Return to the Summary Table.

Action Qualifier Continuous S/W Force Register

Figure 26-123. AQCSFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				CSFB		CSFA	
R-0-0h				R/W-0h		R/W-0h	

Table 26-54. AQCSFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R-0	0h	Reserved
3-2	CSFB	R/W	0h	Continuous Software Force on Output B In immediate mode, a continuous force takes effect on the next TBCLK edge. In shadow mode, a continuous force takes effect on the next TBCLK edge after a shadow load into the active register. To configure shadow mode, use AQSFRC[RLDCSF]. 00: Software forcing is disabled and has no effect 01: Forces a continuous low on output B 10: Forces a continuous high on output B 11: Software forcing is disabled and has no effect Reset type: SYSRSn
1-0	CSFA	R/W	0h	Continuous Software Force on Output A In immediate mode, a continuous force takes effect on the next TBCLK edge. In shadow mode, a continuous force takes effect on the next TBCLK edge after a shadow load into the active register. 00: Software forcing is disabled and has no effect 01: Forces a continuous low on output A 10: Forces a continuous high on output A 11: Software forcing is disabled and has no effect Reset type: SYSRSn

26.17.2.32 DBREDHR Register (Offset = 50h) [Reset = 0h]

DBREDHR is shown in Figure 26-124 and described in Table 26-55.

Return to the Summary Table.

Dead-Band Generator Rising Edge Delay High Resolution Mirror Register

Figure 26-124. DBREDHR Register

15	14	13	12	11	10	9	8
DBREDHR							RESERVED
R/W-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED							RESERVED
R-0h							R-0h

Table 26-55. DBREDHR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	DBREDHR	R/W	0h	Dead Band Rising Edge Delay High Resolution Bits Reset type: SYSRSn
8	RESERVED	R	0h	Reserved
7-1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

26.17.2.33 DBRED Register (Offset = 51h) [Reset = 0h]

DBRED is shown in Figure 26-125 and described in Table 26-56.

Return to the Summary Table.

Dead-Band Generator Rising Edge Delay High Resolution Mirror Register

Figure 26-125. DBRED Register

15	14	13	12	11	10	9	8
RESERVED			DBRED				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
DBRED							
R/W-0h							

Table 26-56. DBRED Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	RESERVED	R	0h	Reserved
13-0	DBRED	R/W	0h	Rising edge delay value Reset type: SYSRSn

26.17.2.34 DBFEDHR Register (Offset = 52h) [Reset = 0h]

DBFEDHR is shown in Figure 26-126 and described in Table 26-57.

Return to the Summary Table.

Dead-Band Generator Falling Edge Delay High Resolution Register

Figure 26-126. DBFEDHR Register

15	14	13	12	11	10	9	8
DBFEDHR							RESERVED
R/W-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED							RESERVED
R-0h							R-0h

Table 26-57. DBFEDHR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	DBFEDHR	R/W	0h	Dead Band Falling Edge Delay High Resolution Bits Reset type: SYSRSn
8	RESERVED	R	0h	Reserved
7-1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

26.17.2.35 DBFED Register (Offset = 53h) [Reset = 0h]

DBFED is shown in Figure 26-127 and described in Table 26-58.

Return to the Summary Table.

Dead-Band Generator Falling Edge Delay Count Register

Figure 26-127. DBFED Register

15	14	13	12	11	10	9	8
RESERVED		DBFED					
R-0h		R/W-0h					
7	6	5	4	3	2	1	0
DBFED							
R/W-0h							

Table 26-58. DBFED Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	RESERVED	R	0h	Reserved
13-0	DBFED	R/W	0h	Falling Edge Delay Count 14-bit counter Reset type: SYSRSn

26.17.2.36 TBPHS Register (Offset = 60h) [Reset = 0h]

TBPHS is shown in Figure 26-128 and described in Table 26-59.

Return to the Summary Table.

Time Base Phase High

Figure 26-128. TBPHS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBPHS																TBPHSHR															
R/W-0h																R/W-0h															

Table 26-59. TBPHS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	TBPHS	R/W	0h	Phase Offset Register These bits set time-base counter phase of the selected ePWM relative to the time-base that is supplying the synchronization input signal. - If TBCTL[PHSEN] = 0, then the synchronization event is ignored and the time-base counter is not loaded with the phase. - If TBCTL[PHSEN] = 1, then the time-base counter (TBCTR) will be loaded with the phase (TBPHS) when a synchronization event occurs. The synchronization event can be initiated by the input synchronization signal (EPWMxSYNCI) or by a software forced synchronization. Reset type: SYSRSn
15-0	TBPHSHR	R/W	0h	Phase Offset (High Resolution) Register. TBPHSHR must not be used. Instead TRREM (HRPWM remainder register) must be used to mimic the functionality of TBPHSHR. The lower 8 bits in this register are ignored - writes are ignored and reads return zero Reset type: SYSRSn

26.17.2.37 TBPRDHR Register (Offset = 62h) [Reset = 0h]

TBPRDHR is shown in Figure 26-129 and described in Table 26-60.

Return to the Summary Table.

Time Base Period High Resolution Register

Figure 26-129. TBPRDHR Register

15	14	13	12	11	10	9	8
TBPRDHR							
R/W-0h							
7	6	5	4	3	2	1	0
TBPRDHR							
R/W-0h							

Table 26-60. TBPRDHR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TBPRDHR	R/W	0h	Period High Resolution Bits The upper 8-bits contain the high-resolution portion of the period value. The TBPRDHR register is not affected by the TBCTL[PRDLD] bit. Reads from this register always reflect the shadow register. Likewise writes are also to the shadow register. The TBPRDHR register is only used when the high resolution period feature is enabled. This register is only available with ePWM modules which support high-resolution period control. The lower 8 bits in this register are ignored - writes are ignored and reads return zero Reset type: SYSRSn

26.17.2.38 TBPRD Register (Offset = 63h) [Reset = 0h]

TBPRD is shown in Figure 26-130 and described in Table 26-61.

Return to the Summary Table.

Time Base Period Register

Figure 26-130. TBPRD Register

15	14	13	12	11	10	9	8
TBPRD							
R/W-0h							
7	6	5	4	3	2	1	0
TBPRD							
R/W-0h							

Table 26-61. TBPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TBPRD	R/W	0h	Time Base Period Register These bits determine the period of the time-base counter. This sets the PWM frequency. Shadowing of this register is enabled and disabled by the TBCTL[PRDLD] bit. By default this register is shadowed. - If TBCTL[PRDLD] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the active register will be loaded from the shadow register when the time-base counter equals zero. - If TBCTL[PRDLD] = 1, then the shadow is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware. - The active and shadow registers share the same memory map address. Reset type: SYSRSn

26.17.2.39 CMPA Register (Offset = 6Ah) [Reset = 0h]

CMPA is shown in Figure 26-131 and described in Table 26-62.

Return to the Summary Table.

Counter Compare A Register

Figure 26-131. CMPA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPA																CMPAHR															
R/W-0h																R/W-0h															

Table 26-62. CMPA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	CMPA	R/W	0h	Compare A Register The value in the active CMPA register is continuously compared to the time-base counter (TBCTR). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare A" event. This event is sent to the action-qualifier where it is qualified and converted it into one or more actions. These actions can be applied to either the EPWMxA or the EPWMxB output depending on the configuration of the AQCTLA and AQCTLB registers. The actions that can be defined in the AQCTLA and AQCTLB registers include: - Do nothing the event is ignored. - Clear: Pull the EPWMxA and/or EPWMxB signal low - Set: Pull the EPWMxA and/or EPWMxB signal high - Toggle the EPWMxA and/or EPWMxB signal Shadowing of this register is enabled and disabled by the CMPCTL[SHDWAMODE] bit. By default this register is shadowed. - If CMPCTL[SHDWAMODE] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the CMPCTL[LOADAMODE] bit field determines which event will load the active register from the shadow register. - Before a write, the CMPCTL[SHDWAFULL] bit can be read to determine if the shadow register is currently full. - If CMPCTL[SHDWAMODE] = 1, then the shadow register is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware. - In either mode, the active and shadow registers share the same memory map address. Reset type: SYSRSn
15-0	CMPAHR	R/W	0h	Compare A HRPWM Extension Register The UPPER 8-bits contain the high-resolution portion (most significant 8-bits) of the counter-compare A value. CMPA:CMPAHR can be accessed in a single 32-bit read/write. Shadowing is enabled and disabled by the CMPCTL[SHDWAMODE] bit as described for the CMPA register. The lower 8 bits in this register are ignored Reset type: SYSRSn

26.17.2.40 CMPB Register (Offset = 6Ch) [Reset = 0h]

CMPB is shown in Figure 26-132 and described in Table 26-63.

Return to the Summary Table.

Compare B Register

Figure 26-132. CMPB Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPB																CMPBHR															
R/W-0h																R/W-0h															

Table 26-63. CMPB Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	CMPB	R/W	0h	Compare B Register The value in the active CMPB register is continuously compared to the time-base counter (TBCTR). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare B" event. This event is sent to the action-qualifier where it is qualified and converted it into one or more actions. These actions can be applied to either the EPWMxA or the EPWMxB output depending on the configuration of the AQCTLA and AQCTLB registers. The actions that can be defined in the AQCTLA and AQCTLB registers include: - Do nothing the event is ignored. - Clear: Pull the EPWMxA and/or EPWMxB signal low - Set: Pull the EPWMxA and/or EPWMxB signal high - Toggle the EPWMxA and/or EPWMxB signal Shadowing of this register is enabled and disabled by the CMPCTL[SHDWBMODE] bit. By default this register is shadowed. - If CMPCTL[SHDWBMODE] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the CMPCTL[LOADBMODE] bit field determines which event will load the active register from the shadow register. - Before a write, the CMPCTL[SHDWBFULL] bit can be read to determine if the shadow register is currently full. - If CMPCTL[SHDWBMODE] = 1, then the shadow register is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware. - In either mode, the active and shadow registers share the same memory map address. Reset type: SYSRSn
15-0	CMPBHR	R/W	0h	Compare B High Resolution Bits The lower 8 bits in this register are ignored Reset type: SYSRSn

26.17.2.41 CMPC Register (Offset = 6Fh) [Reset = 0h]

CMPC is shown in Figure 26-133 and described in Table 26-64.

Return to the Summary Table.

Counter Compare C Register
LINK feature access should always be 16-bit

Figure 26-133. CMPC Register

15	14	13	12	11	10	9	8
CMPC							
R/W-0h							
7	6	5	4	3	2	1	0
CMPC							
R/W-0h							

Table 26-64. CMPC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	CMPC	R/W	0h	Compare C Register The value in the active CMPC register is continuously compared to the time-base counter (TBCTR). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare C" event. Shadowing of this register is enabled and disabled by the CMPCTL2[SHDWCMODE] bit. By default this register is shadowed. - If CMPCTL2[SHDWCMODE] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the CMPCTL2[LOADCMODE] bit field determines which event will load the active register from the shadow register: - If CMPCTL2[SHDWCMODE] = 1, then the shadow register is disabled and any write or read will go directly to the active register that is, the register actively controlling the hardware. - In either mode, the active and shadow registers share the same memory map address. Reset type: SYSRSn

26.17.2.42 CMPD Register (Offset = 71h) [Reset = 0h]

CMPD is shown in Figure 26-134 and described in Table 26-65.

Return to the Summary Table.

Counter Compare D Register
LINK feature access should always be 16-bit

Figure 26-134. CMPD Register

15	14	13	12	11	10	9	8
CMPD							
R/W-0h							
7	6	5	4	3	2	1	0
CMPD							
R/W-0h							

Table 26-65. CMPD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	CMPD	R/W	0h	Compare D Register The value in the active CMPD register is continuously compared to the time-base counter (TBCTR). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare D" event. Shadowing of this register is enabled and disabled by the CMPCTL2[SHDWDMODE] bit. By default this register is shadowed. - If CMPCTL2[SHDWDMODE] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the CMPCTL2[LOADDMODE] bit field determines which event will load the active register from the shadow register: - If CMPCTL2[SHDWDMODE] = 1, then the shadow register is disabled and any write or read will go directly to the active register that is, the register actively controlling the hardware. - In either mode, the active and shadow registers share the same memory map address. Reset type: SYSRSn

26.17.2.43 GLDCTL2 Register (Offset = 74h) [Reset = 0h]

GLDCTL2 is shown in Figure 26-135 and described in Table 26-66.

Return to the Summary Table.

Global PWM Load Control Register 2

Figure 26-135. GLDCTL2 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						GFRCLD	OSHTLD
R-0-0h						R-0/W1S-0h	R-0/W1S-0h

Table 26-66. GLDCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R-0	0h	Reserved
1	GFRCLD	R-0/W1S	0h	Force Load Event in One Shot Mode 0: Writing of 0 will be ignored. Always reads back a 0. 1: Force one load event at the input of the event pre-scale counter. This bit is intended to be used for testing and/or software force loading of the events in global load mode. Reset type: SYSRSn
0	OSHTLD	R-0/W1S	0h	Enable Reload Event in One Shot Mode 0: Writing of 0 will be ignored. Always reads back a 0. 1: Turns the one shot latch condition ON. Upon occurrence of a chosen load strobe, one shadow to active reload occurs and the latch will be cleared. Hence writing 1 to this bit would allow one load strobe event to pass through and block further strobe events. Reset type: SYSRSn

26.17.2.44 SWVDELVAL Register (Offset = 77h) [Reset = 0h]

SWVDELVAL is shown in Figure 26-136 and described in Table 26-67.

Return to the Summary Table.

Software Valley Mode Delay Register

Figure 26-136. SWVDELVAL Register

15	14	13	12	11	10	9	8
SWVDELVAL							
R/W-0h							
7	6	5	4	3	2	1	0
SWVDELVAL							
R/W-0h							

Table 26-67. SWVDELVAL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SWVDELVAL	R/W	0h	Software Valley Delay Value Register This register can be optionally used define offset value for the hardware calculated delay HWDELAYVAL as defined in VCAPCTL[VDELAYDIV] bits. Reset type: SYSRSn

26.17.2.45 TZSEL Register (Offset = 80h) [Reset = 0h]

TZSEL is shown in Figure 26-137 and described in Table 26-68.

Return to the Summary Table.

Trip Zone Select Register

Figure 26-137. TZSEL Register

15	14	13	12	11	10	9	8
DCBEVT1	DCAEVT1	OSHT6	OSHT5	OSHT4	OSHT3	OSHT2	OSHT1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DCBEVT2	DCAEVT2	CBC6	CBC5	CBC4	CBC3	CBC2	CBC1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 26-68. TZSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	DCBEVT1	R/W	0h	Digital Compare Output B Event 1 Select 0: Disable DCBEVT1 as one-shot-trip source for this ePWM module. 1: Enable DCBEVT1 as one-shot-trip source for this ePWM module. Reset type: SYSRSn
14	DCAEVT1	R/W	0h	Digital Compare Output A Event 1 Select 0: Disable DCAEVT1 as one-shot-trip source for this ePWM module. 1: Enable DCAEVT1 as one-shot-trip source for this ePWM module. Reset type: SYSRSn
13	OSHT6	R/W	0h	Trip-zone 6 (TZ6) Select 0: Disable TZ6 as a one-shot trip source for this ePWM module 1: Enable TZ6 as a one-shot trip source for this ePWM module Reset type: SYSRSn
12	OSHT5	R/W	0h	Trip-zone 5 (TZ5) Select 0: Disable TZ5 as a one-shot trip source for this ePWM module 1: Enable TZ5 as a one-shot trip source for this ePWM module Reset type: SYSRSn
11	OSHT4	R/W	0h	Trip-zone 4 (TZ4) Select 0: Disable TZ4 as a one-shot trip source for this ePWM module 1: Enable TZ4 as a one-shot trip source for this ePWM module Reset type: SYSRSn
10	OSHT3	R/W	0h	Trip-zone 3 (TZ3) Select 0: Disable TZ3 as a one-shot trip source for this ePWM module 1: Enable TZ3 as a one-shot trip source for this ePWM module Reset type: SYSRSn
9	OSHT2	R/W	0h	Trip-zone 2 (TZ2) Select 0: Disable TZ2 as a one-shot trip source for this ePWM module 1: Enable TZ2 as a one-shot trip source for this ePWM module Reset type: SYSRSn
8	OSHT1	R/W	0h	Trip-zone 1 (TZ1) Select 0: Disable TZ1 as a one-shot trip source for this ePWM module 1: Enable TZ1 as a one-shot trip source for this ePWM module Reset type: SYSRSn
7	DCBEVT2	R/W	0h	Digital Compare Output B Event 2 Select 0: Disable DCBEVT2 as a CBC trip source for this ePWM module 1: Enable DCBEVT2 as a CBC trip source for this ePWM module Reset type: SYSRSn
6	DCAEVT2	R/W	0h	Digital Compare Output A Event 2 Select 0: Disable DCAEVT2 as a CBC trip source for this ePWM module 1: Enable DCAEVT2 as a CBC trip source for this ePWM module Reset type: SYSRSn

Table 26-68. TZSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	CBC6	R/W	0h	Trip-zone 6 (TZ6) Select 0: Disable TZ6 as a CBC trip source for this ePWM module 1: Enable TZ6 as a CBC trip source for this ePWM module Reset type: SYSRSn
4	CBC5	R/W	0h	Trip-zone 5 (TZ5) Select 0: Disable TZ5 as a CBC trip source for this ePWM module 1: Enable TZ5 as a CBC trip source for this ePWM module Reset type: SYSRSn
3	CBC4	R/W	0h	Trip-zone 4 (TZ4) Select 0: Disable TZ4 as a CBC trip source for this ePWM module 1: Enable TZ4 as a CBC trip source for this ePWM module Reset type: SYSRSn
2	CBC3	R/W	0h	Trip-zone 3 (TZ3) Select 0: Disable TZ3 as a CBC trip source for this ePWM module 1: Enable TZ3 as a CBC trip source for this ePWM module Reset type: SYSRSn
1	CBC2	R/W	0h	Trip-zone 2 (TZ2) Select 0: Disable TZ2 as a CBC trip source for this ePWM module 1: Enable TZ2 as a CBC trip source for this ePWM module Reset type: SYSRSn
0	CBC1	R/W	0h	Trip-zone 1 (TZ1) Select 0: Disable TZ1 as a CBC trip source for this ePWM module 1: Enable TZ1 as a CBC trip source for this ePWM module Reset type: SYSRSn

26.17.2.46 TZDCSEL Register (Offset = 82h) [Reset = 0h]

TZDCSEL is shown in Figure 26-138 and described in Table 26-69.

Return to the Summary Table.

Trip Zone Digital Comparator Select Register

Figure 26-138. TZDCSEL Register

15	14	13	12	11	10	9	8
RESERVED				DCBEVT2			DCBEVT1
R-0-0h				R/W-0h			R/W-0h
7	6	5	4	3	2	1	0
DCBEVT1		DCAEVT2			DCAEVT1		
R/W-0h		R/W-0h			R/W-0h		

Table 26-69. TZDCSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R-0	0h	Reserved
11-9	DCBEVT2	R/W	0h	Digital Compare Output B Event 2 Selection 000: Event disabled 001: DCBH = low, DCBL = don't care 010: DCBH = high, DCBL = don't care 011: DCBL = low, DCBH = don't care 100: DCBL = high, DCBH = don't care 101: DCBL = high, DCBH = low 110: Reserved 111: Reserved Reset type: SYSRSn
8-6	DCBEVT1	R/W	0h	Digital Compare Output B Event 1 Selection 000: Event disabled 001: DCBH = low, DCBL = don't care 010: DCBH = high, DCBL = don't care 011: DCBL = low, DCBH = don't care 100: DCBL = high, DCBH = don't care 101: DCBL = high, DCBH = low 110: Reserved 111: Reserved Reset type: SYSRSn
5-3	DCAEVT2	R/W	0h	Digital Compare Output A Event 2 Selection 000: Event disabled 001: DCAH = low, DCAL = don't care 010: DCAH = high, DCAL = don't care 011: DCAL = low, DCAH = don't care 100: DCAL = high, DCAH = don't care 101: DCAL = high, DCAH = low 110: Reserved 111: Reserved Reset type: SYSRSn
2-0	DCAEVT1	R/W	0h	Digital Compare Output A Event 1 Selection 000: Event disabled 001: DCAH = low, DCAL = don't care 010: DCAH = high, DCAL = don't care 011: DCAL = low, DCAH = don't care 100: DCAL = high, DCAH = don't care 101: DCAL = high, DCAH = low 110: Reserved 111: Reserved Reset type: SYSRSn

26.17.2.47 TZCTL Register (Offset = 84h) [Reset = 0h]

TZCTL is shown in Figure 26-139 and described in Table 26-70.

Return to the Summary Table.

Trip Zone Control Register

Figure 26-139. TZCTL Register

15	14	13	12	11	10	9	8
RESERVED				DCBEVT2		DCBEVT1	
R-0-0h				R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
DCAEVT2		DCAEVT1		TZB		TZA	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 26-70. TZCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R-0	0h	Reserved
11-10	DCBEVT2	R/W	0h	Digital Compare Output B Event 2 Action On EPWMxB 00: High-impedance (EPWMxB = High-impedance state) 01: Force EPWMxB to a high state. 10: Force EPWMxB to a low state. 11: Do Nothing, trip action is disabled Reset type: SYSRSn
9-8	DCBEVT1	R/W	0h	Digital Compare Output B Event 1 Action On EPWMxB 00: High-impedance (EPWMxB = High-impedance state) 01: Force EPWMxB to a high state. 10: Force EPWMxB to a low state. 11: Do Nothing, trip action is disabled Reset type: SYSRSn
7-6	DCAEVT2	R/W	0h	Digital Compare Output A Event 2 Action On EPWMxA 00: High-impedance (EPWMxA = High-impedance state) 01: Force EPWMxA to a high state. 10: Force EPWMxA to a low state. 11: Do Nothing, trip action is disabled Reset type: SYSRSn
5-4	DCAEVT1	R/W	0h	Digital Compare Output A Event 1 Action On EPWMxA 00: High-impedance (EPWMxA = High-impedance state) 01: Force EPWMxA to a high state. 10: Force EPWMxA to a low state. 11: Do Nothing, trip action is disabled Reset type: SYSRSn
3-2	TZB	R/W	0h	TZ1 to TZ6, DCAEVT1/2, DCBEVT1/2Trip Action On EPWMxB When a trip event occurs the following action is taken on output EPWMxB. Which trip-zone pins can cause an event is defined in the TZSEL register. 00: High-impedance (EPWMxB = High-impedance state) 01: Force EPWMxB to a high state 10: Force EPWMxB to a low state 11: Do nothing, no action is taken on EPWMxB. Reset type: SYSRSn
1-0	TZA	R/W	0h	TZ1 to TZ6, DCAEVT1/2, DCBEVT1/2 Trip Action On EPWMxA When a trip event occurs the following action is taken on output EPWMxA. Which trip-zone pins can cause an event is defined in the TZSEL register. 00: High-impedance (EPWMxA = High-impedance state) 01: Force EPWMxA to a high state 10: Force EPWMxA to a low state 11: Do nothing, no action is taken on EPWMxA. Reset type: SYSRSn

26.17.2.48 TZCTL2 Register (Offset = 85h) [Reset = 0h]

TZCTL2 is shown in Figure 26-140 and described in Table 26-71.

Return to the Summary Table.

Additional Trip Zone Control Register

Figure 26-140. TZCTL2 Register

15	14	13	12	11	10	9	8
ETZE	RESERVED			TZBD			TZBU
R/W-0h	R-0-0h			R/W-0h			R/W-0h
7	6	5	4	3	2	1	0
TZBU		TZAD			TZAU		
R/W-0h		R/W-0h			R/W-0h		

Table 26-71. TZCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	ETZE	R/W	0h	TZCTL2 Enable 0: Use trip action from TZCTL (legacy EPWM compatibility) 1: Use trip action defined in TZCTL2, TZCTLDCA and TZCTLDCB. Settings in TZCTL are ignored Reset type: SYSRSn
14-12	RESERVED	R-0	0h	Reserved
11-9	TZBD	R/W	0h	TZ1 to TZ6 Trip Action On EPWMxB while Count direction is DOWN 000: HiZ (EPWMxB = HiZ state) 001: Forced Hi (EPWMxB = High state) 010: Forced Lo (EPWMxB = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn
8-6	TZBU	R/W	0h	TZ1 to TZ6, DCAEVT1/2, DCBEVT1/2 Trip Action On EPWMxB while Count direction is UP 000: HiZ (EPWMxB = HiZ state) 001: Forced Hi (EPWMxB = High state) 010: Forced Lo (EPWMxB = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn
5-3	TZAD	R/W	0h	TZ1 to TZ6, DCAEVT1/2, DCBEVT1/2 Trip Action On EPWMxA while Count direction is DOWN 000: HiZ (EPWMxA = HiZ state) 001: Forced Hi (EPWMxA = High state) 010: Forced Lo (EPWMxA = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn

Table 26-71. TZCTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	TZAU	R/W	0h	TZ1 to TZ6, DCAEVT1/2, DCBEVT1/2 Trip Action On EPWMxA while Count direction is UP 000: HiZ (EPWMxA = HiZ state) 001: Forced Hi (EPWMxA = High state) 010: Forced Lo (EPWMxA = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn

26.17.2.49 TZCTLDCA Register (Offset = 86h) [Reset = 0h]

TZCTLDCA is shown in Figure 26-141 and described in Table 26-72.

Return to the Summary Table.

Trip Zone Control Register Digital Compare A

Figure 26-141. TZCTLDCA Register

15	14	13	12	11	10	9	8
RESERVED				DCAEVT2D			DCAEVT2U
R-0-0h				R/W-0h			R/W-0h
7	6	5	4	3	2	1	0
DCAEVT2U		DCAEVT1D			DCAEVT1U		
R/W-0h		R/W-0h			R/W-0h		

Table 26-72. TZCTLDCA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R-0	0h	Reserved
11-9	DCAEVT2D	R/W	0h	Digital Compare Output A Event 2 Action On EPWMxA while Count direction is DOWN 000: HiZ (EPWMxA = HiZ state) 001: Forced Hi (EPWMxA = High state) 010: Forced Lo (EPWMxA = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn
8-6	DCAEVT2U	R/W	0h	Digital Compare Output A Event 2 Action On EPWMxA while Count direction is UP 000: HiZ (EPWMxA = HiZ state) 001: Forced Hi (EPWMxA = High state) 010: Forced Lo (EPWMxA = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn
5-3	DCAEVT1D	R/W	0h	Digital Compare Output A Event 1 Action On EPWMxA while Count direction is DOWN 000: HiZ (EPWMxA = HiZ state) 001: Forced Hi (EPWMxA = High state) 010: Forced Lo (EPWMxA = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn

Table 26-72. TZCTLDCA Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	DCAEVT1U	R/W	0h	Digital Compare Output A Event 1 Action On EPWMxA while Count direction is UP 000: HiZ (EPWMxA = HiZ state) 001: Forced Hi (EPWMxA = High state) 010: Forced Lo (EPWMxA = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn

26.17.2.50 TZCTLDCB Register (Offset = 87h) [Reset = 0h]

TZCTLDCB is shown in Figure 26-142 and described in Table 26-73.

Return to the Summary Table.

Trip Zone Control Register Digital Compare B

Figure 26-142. TZCTLDCB Register

15	14	13	12	11	10	9	8
RESERVED				DCBEVT2D			DCBEVT2U
R-0-0h				R/W-0h			R/W-0h
7	6	5	4	3	2	1	0
DCBEVT2U			DCBEVT1D			DCBEVT1U	
R/W-0h			R/W-0h			R/W-0h	

Table 26-73. TZCTLDCB Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R-0	0h	Reserved
11-9	DCBEVT2D	R/W	0h	Digital Compare Output B Event 2 Action On EPWMxB while Count direction is DOWN 000: HiZ (EPWMxB = HiZ state) 001: Forced Hi (EPWMxB = High state) 010: Forced Lo (EPWMxB = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn
8-6	DCBEVT2U	R/W	0h	Digital Compare Output B Event 2 Action On EPWMxB while Count direction is UP 000: HiZ (EPWMxB = HiZ state) 001: Forced Hi (EPWMxB = High state) 010: Forced Lo (EPWMxB = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn
5-3	DCBEVT1D	R/W	0h	Digital Compare Output B Event 1 Action On EPWMxB while Count direction is DOWN 000: HiZ (EPWMxB = HiZ state) 001: Forced Hi (EPWMxB = High state) 010: Forced Lo (EPWMxB = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn

Table 26-73. TZCTLDCB Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	DCBEVT1U	R/W	0h	Digital Compare Output B Event 1 Action On EPWMxB while Count direction is UP 000: HiZ (EPWMxB = HiZ state) 001: Forced Hi (EPWMxB = High state) 010: Forced Lo (EPWMxB = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn

26.17.2.51 TZEINT Register (Offset = 8Dh) [Reset = 0h]

TZEINT is shown in Figure 26-143 and described in Table 26-74.

Return to the Summary Table.

Trip Zone Enable Interrupt Register

Figure 26-143. TZEINT Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED	DCBEVT2	DCBEVT1	DCAEVT2	DCAEVT1	OST	CBC	RESERVED
R-0-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0-0h

Table 26-74. TZEINT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R-0	0h	Reserved
6	DCBEVT2	R/W	0h	Digital Compare Output B Event 2 Interrupt Enable 0: Disabled 1: Enabled Reset type: SYSRSn
5	DCBEVT1	R/W	0h	Digital Compare Output B Event 1 Interrupt Enable 0: Disabled 1: Enabled Reset type: SYSRSn
4	DCAEVT2	R/W	0h	Digital Compare Output A Event 2 Interrupt Enable 0: Disabled 1: Enabled Reset type: SYSRSn
3	DCAEVT1	R/W	0h	Digital Compare Output A Event 1 Interrupt Enable 0: Disabled 1: Enabled Reset type: SYSRSn
2	OST	R/W	0h	Trip-zone One-Shot Interrupt Enable 0: Disable one-shot interrupt generation 1: Enable Interrupt generation a one-shot trip event will cause a EPWMx_TZINT PIE interrupt. Reset type: SYSRSn
1	CBC	R/W	0h	Trip-zone Cycle-by-Cycle Interrupt Enable 0: Disable cycle-by-cycle interrupt generation. 1: Enable interrupt generation a cycle-by-cycle trip event will cause an EPWMx_TZINT PIE interrupt. Reset type: SYSRSn
0	RESERVED	R-0	0h	Reserved

26.17.2.52 TZFLG Register (Offset = 93h) [Reset = 0h]

TZFLG is shown in Figure 26-144 and described in Table 26-75.

Return to the Summary Table.

Trip Zone Flag Register

Figure 26-144. TZFLG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED	DCBEVT2	DCBEVT1	DCAEVT2	DCAEVT1	OST	CBC	INT
R-0-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 26-75. TZFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R-0	0h	Reserved
6	DCBEVT2	R	0h	Latched Status Flag for Digital Compare Output B Event 2 0: Indicates no trip event has occurred on DCBEVT2 1: Indicates a trip event has occurred for the event defined for DCBEVT2 Reset type: SYSRSn
5	DCBEVT1	R	0h	Latched Status Flag for Digital Compare Output B Event 1 0: Indicates no trip event has occurred on DCBEVT1 1: Indicates a trip event has occurred for the event defined for DCBEVT1 Reset type: SYSRSn
4	DCAEVT2	R	0h	Latched Status Flag for Digital Compare Output A Event 2 0: Indicates no trip event has occurred on DCAEVT2 1: Indicates a trip event has occurred for the event defined for DCAEVT2 Reset type: SYSRSn
3	DCAEVT1	R	0h	Latched Status Flag for Digital Compare Output A Event 1 0: Indicates no trip event has occurred on DCAEVT1 1: Indicates a trip event has occurred for the event defined for DCAEVT1 Reset type: SYSRSn
2	OST	R	0h	Latched Status Flag for A One-Shot Trip Event 0: No one-shot trip event has occurred. 1: Indicates a trip event has occurred on a pin selected as a one-shot trip source. This bit is cleared by writing the appropriate value to the TZCLR register. Reset type: SYSRSn
1	CBC	R	0h	Latched Status Flag for Cycle-By-Cycle Trip Event 0: No cycle-by-cycle trip event has occurred. 1: Indicates a trip event has occurred on a signal selected as a cycle-by-cycle trip source. The TZFLG[CBC] bit will remain set until it is manually cleared by the user. If the cycle-by-cycle trip event is still present when the CBC bit is cleared, then CBC will be immediately set again. The specified condition on the signal is automatically cleared when the ePWM time-base counter reaches zero (TBCTR = 0x00) if the trip condition is no longer present. The condition on the signal is only cleared when the TBCTR = 0x00 no matter where in the cycle the CBC flag is cleared. This bit is cleared by writing the appropriate value to the TZCLR register. Reset type: SYSRSn

Table 26-75. TZFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INT	R	0h	Latched Trip Interrupt Status Flag 0: Indicates no interrupt has been generated. 1: Indicates an EPWMx_TZINT PIE interrupt was generated because of a trip condition. No further EPWMx_TZINT PIE interrupts will be generated until this flag is cleared. If the interrupt flag is cleared when either CBC or OST is set, then another interrupt pulse will be generated. Clearing all flag bits will prevent further interrupts. This bit is cleared by writing the appropriate value to the TZCLR register. Reset type: SYSRSn

26.17.2.53 TZCBCFLG Register (Offset = 94h) [Reset = 0h]

TZCBCFLG is shown in Figure 26-145 and described in Table 26-76.

Return to the Summary Table.

Trip Zone CBC Flag Register

Figure 26-145. TZCBCFLG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
DCBEVT2	DCAEVT2	CBC6	CBC5	CBC4	CBC3	CBC2	CBC1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 26-76. TZCBCFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R-0	0h	Reserved
7	DCBEVT2	R	0h	Latched Status Flag for Digital Compare B Output Event 2 Trip Latch 0: Reading a 0 indicates that no trip has occurred on DCBEVT2. 1: Reading a 1 indicates a trip has occured on the DCBEVT2 selected event. Reset type: SYSRSn
6	DCAEVT2	R	0h	Latched Status Flag for Digital Compare A Output Event 2 Trip Latch 0: Reading a 0 indicates that no trip has occurred on DCAEVT2. 1: Reading a 1 indicates a trip has occured on the DCAEVT2 selected event. Reset type: SYSRSn
5	CBC6	R	0h	Latched Status Flag for CBC6 Trip Latch 0: Reading a 0 indicates that no trip has occurred on CBC6. 1: Reading a 1 indicates a trip has occured on the CBC6 selected event. Reset type: SYSRSn
4	CBC5	R	0h	Latched Status Flag for CBC5 Trip Latch 0: Reading a 0 indicates that no trip has occurred on CBC5. 1: Reading a 1 indicates a trip has occured on the CBC5 selected event. Reset type: SYSRSn
3	CBC4	R	0h	Latched Status Flag for CBC4 Trip Latch 0: Reading a 0 indicates that no trip has occurred on CBC4. 1: Reading a 1 indicates a trip has occured on the CBC4 selected event. Reset type: SYSRSn
2	CBC3	R	0h	Latched Status Flag for CBC3 Trip Latch 0: Reading a 0 indicates that no trip has occurred on CBC3. 1: Reading a 1 indicates a trip has occured on the CBC3 selected event. Reset type: SYSRSn
1	CBC2	R	0h	Latched Status Flag for CBC2 Trip Latch 0: Reading a 0 indicates that no trip has occurred on CBC2. 1: Reading a 1 indicates a trip has occured on the CBC2 selected event. Reset type: SYSRSn
0	CBC1	R	0h	Latched Status Flag for CBC1 Trip Latch 0: Reading a 0 indicates that no trip has occurred on CBC1. 1: Reading a 1 indicates a trip has occured on the CBC1 selected event. Reset type: SYSRSn

26.17.2.54 TZOSTFLG Register (Offset = 95h) [Reset = 0h]

TZOSTFLG is shown in Figure 26-146 and described in Table 26-77.

Return to the Summary Table.

Trip Zone OST Flag Register

Figure 26-146. TZOSTFLG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
DCBEVT1	DCAEVT1	OST6	OST5	OST4	OST3	OST2	OST1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 26-77. TZOSTFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R-0	0h	Reserved
7	DCBEVT1	R	0h	Latched Status Flag for Digital Compare B Output Event 1 Trip Latch 0: Reading a 0 indicates that no trip has occurred on DCBEVT1. 1: Reading a 1 indicates a trip has occured on the DCBEVT1 selected event. Reset type: SYSRSn
6	DCAEVT1	R	0h	Latched Status Flag for Digital Compare A Output Event 1 Trip Latch 0: Reading a 0 indicates that no trip has occurred on DCAEVT1. 1: Reading a 1 indicates a trip has occured on the DCAEVT1 selected event. Reset type: SYSRSn
5	OST6	R	0h	Latched Status Flag for OST6 Trip Latch 0: Reading a 0 indicates that no trip has occurred on OST6. 1: Reading a 1 indicates a trip has occured on the OST6 selected event. Reset type: SYSRSn
4	OST5	R	0h	Latched Status Flag for OST5 Trip Latch 0: Reading a 0 indicates that no trip has occurred on OST5. 1: Reading a 1 indicates a trip has occured on the OST5 selected event. Reset type: SYSRSn
3	OST4	R	0h	Latched Status Flag for OST4 Trip Latch 0: Reading a 0 indicates that no trip has occurred on OST4. 1: Reading a 1 indicates a trip has occured on the OST4 selected event. Reset type: SYSRSn
2	OST3	R	0h	Latched Status Flag for OST3 Trip Latch 0: Reading a 0 indicates that no trip has occurred on OST3. 1: Reading a 1 indicates a trip has occured on the OST3 selected event. Reset type: SYSRSn
1	OST2	R	0h	Latched Status Flag for OST2 Trip Latch 0: Reading a 0 indicates that no trip has occurred on OST2. 1: Reading a 1 indicates a trip has occured on the OST2 selected event. Reset type: SYSRSn
0	OST1	R	0h	Latched Status Flag for OST1 Trip Latch 0: Reading a 0 indicates that no trip has occurred on OST1. 1: Reading a 1 indicates a trip has occured on the OST1 selected event. Reset type: SYSRSn

26.17.2.55 TZCLR Register (Offset = 97h) [Reset = 0h]

TZCLR is shown in Figure 26-147 and described in Table 26-78.

Return to the Summary Table.

Trip Zone Clear Register

Figure 26-147. TZCLR Register

15	14	13	12	11	10	9	8
CBCPULSE		RESERVED					
R/W-0h		R-0-0h					
7	6	5	4	3	2	1	0
RESERVED	DCBEVT2	DCBEVT1	DCAEVT2	DCAEVT1	OST	CBC	INT
R-0-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 26-78. TZCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	CBCPULSE	R/W	0h	Clear Pulse for Cycle-By-Cycle (CBC) Trip Latch This bit field determines which pulse clears the CBC trip latch. 00: CTR = zero pulse clears CBC trip latch. (Same as legacy designs.) 01: CTR = PRD pulse clears CBC trip latch. 10: CTR = zero or CTR = PRD pulse clears CBC trip latch. 11: CBC trip latch is not cleared Reset type: SYSRSn
13-7	RESERVED	R-0	0h	Reserved
6	DCBEVT2	R-0/W1S	0h	Clear Flag for Digital Compare Output B Event 2 0: Writing 0 has no effect. This bit always reads back 0. 1: Writing 1 clears the DCBEVT2 event trip condition. Reset type: SYSRSn
5	DCBEVT1	R-0/W1S	0h	Clear Flag for Digital Compare Output B Event 1 0: Writing 0 has no effect. This bit always reads back 0. 1: Writing 1 clears the DCBEVT1 event trip condition. Reset type: SYSRSn
4	DCAEVT2	R-0/W1S	0h	Clear Flag for Digital Compare Output A Event 2 0: Writing 0 has no effect. This bit always reads back 0. 1: Writing 1 clears the DCAEVT2 event trip condition. Reset type: SYSRSn
3	DCAEVT1	R-0/W1S	0h	Clear Flag for Digital Compare Output A Event 1 0: Writing 0 has no effect. This bit always reads back 0. 1: Writing 1 clears the DCAEVT1 event trip condition. Reset type: SYSRSn
2	OST	R-0/W1S	0h	Clear Flag for One-Shot Trip (OST) Latch 0: Has no effect. Always reads back a 0. 1: Clears this Trip (set) condition. Reset type: SYSRSn
1	CBC	R-0/W1S	0h	Clear Flag for Cycle-By-Cycle (CBC) Trip Latch 0: Has no effect. Always reads back a 0. 1: Clears this Trip (set) condition. Reset type: SYSRSn
0	INT	R-0/W1S	0h	Global Interrupt Clear Flag 0: Has no effect. Always reads back a 0. 1: Clears the trip-interrupt flag for this ePWM module (TZFLG[INT]). NOTE: No further EPWMx_TZINT PIE interrupts will be generated until the flag is cleared. If the TZFLG[INT] bit is cleared and any of the other flag bits are set, then another interrupt pulse will be generated. Clearing all flag bits will prevent further interrupts. Reset type: SYSRSn

26.17.2.56 TZCBCCLR Register (Offset = 98h) [Reset = 0h]

TZCBCCLR is shown in Figure 26-148 and described in Table 26-79.

Return to the Summary Table.

Trip Zone CBC Clear Register

Figure 26-148. TZCBCCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
DCBEVT2	DCAEVT2	CBC6	CBC5	CBC4	CBC3	CBC2	CBC1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 26-79. TZCBCCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R-0	0h	Reserved
7	DCBEVT2	R-0/W1S	0h	Clear Flag for Digital Compare Output B Event 2 selected for CBC 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZCBCFLG[DCBEVT2] bit. Reset type: SYSRSn
6	DCAEVT2	R-0/W1S	0h	Clear Flag for Digital Compare Output A Event 2 selected for CBC 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZCBCFLG[DCAEVT2] bit. Reset type: SYSRSn
5	CBC6	R-0/W1S	0h	Clear Flag for Cycle-By-Cycle (CBC6) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZCBCFLG[CBC6] bit. Reset type: SYSRSn
4	CBC5	R-0/W1S	0h	Clear Flag for Cycle-By-Cycle (CBC5) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZCBCFLG[CBC5] bit. Reset type: SYSRSn
3	CBC4	R-0/W1S	0h	Clear Flag for Cycle-By-Cycle (CBC4) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZCBCFLG[CBC4] bit. Reset type: SYSRSn
2	CBC3	R-0/W1S	0h	Clear Flag for Cycle-By-Cycle (CBC3) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZCBCFLG[CBC3] bit. Reset type: SYSRSn
1	CBC2	R-0/W1S	0h	Clear Flag for Cycle-By-Cycle (CBC2) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZCBCFLG[CBC2] bit. Reset type: SYSRSn
0	CBC1	R-0/W1S	0h	Clear Flag for Cycle-By-Cycle (CBC1) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZCBCFLG[CBC1] bit. Reset type: SYSRSn

26.17.2.57 TZOSTCLR Register (Offset = 99h) [Reset = 0h]

TZOSTCLR is shown in Figure 26-149 and described in Table 26-80.

Return to the Summary Table.

Trip Zone OST Clear Register

Figure 26-149. TZOSTCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
DCBEVT1	DCAEVT1	OST6	OST5	OST4	OST3	OST2	OST1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 26-80. TZOSTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R-0	0h	Reserved
7	DCBEVT1	R-0/W1S	0h	Clear Flag for Digital Compare Output B Event 1 selected for OST 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZOSTFLG[DCBEVT1] bit. Reset type: SYSRSn
6	DCAEVT1	R-0/W1S	0h	Clear Flag for Digital Compare Output A Event 1 selected for OST 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZOSTFLG[DCAEVT1] bit. Reset type: SYSRSn
5	OST6	R-0/W1S	0h	Clear Flag for Oneshot (OST6) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZOSTFLG[OST6] bit. Reset type: SYSRSn
4	OST5	R-0/W1S	0h	Clear Flag for Oneshot (OST5) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZOSTFLG[OST5] bit. Reset type: SYSRSn
3	OST4	R-0/W1S	0h	Clear Flag for Oneshot (OST4) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZOSTFLG[OST4] bit. Reset type: SYSRSn
2	OST3	R-0/W1S	0h	Clear Flag for Oneshot (OST3) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZOSTFLG[OST3] bit. Reset type: SYSRSn
1	OST2	R-0/W1S	0h	Clear Flag for Oneshot (OST2) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZOSTFLG[OST2] bit. Reset type: SYSRSn
0	OST1	R-0/W1S	0h	Clear Flag for Oneshot (OST1) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZOSTFLG[OST1] bit. Reset type: SYSRSn

26.17.2.58 TZFRC Register (Offset = 9Bh) [Reset = 0h]

TZFRC is shown in Figure 26-150 and described in Table 26-81.

Return to the Summary Table.

Trip Zone Force Register

Figure 26-150. TZFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED	DCBEVT2	DCBEVT1	DCAEVT2	DCAEVT1	OST	CBC	RESERVED
R-0-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0-0h

Table 26-81. TZFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R-0	0h	Reserved
6	DCBEVT2	R-0/W1S	0h	Force Flag for Digital Compare Output B Event 2 0: Writing 0 has no effect. This bit always reads back 0. 1: Writing 1 forces the DCBEVT2 event trip condition and sets the TZFLG[DCBEVT2] bit. Reset type: SYSRSn
5	DCBEVT1	R-0/W1S	0h	Force Flag for Digital Compare Output B Event 1 0: Writing 0 has no effect. This bit always reads back 0. 1: Writing 1 forces the DCBEVT1 event trip condition and sets the TZFLG[DCBEVT1] bit. Reset type: SYSRSn
4	DCAEVT2	R-0/W1S	0h	Force Flag for Digital Compare Output A Event 2 0: Writing 0 has no effect. This bit always reads back 0. 1: Writing 1 forces the DCAEVT2 event trip condition and sets the TZFLG[DCAEVT2] bit. Reset type: SYSRSn
3	DCAEVT1	R-0/W1S	0h	Force Flag for Digital Compare Output A Event 1 0: Writing 0 has no effect. This bit always reads back 0 1: Writing 1 forces the DCAEVT1 event trip condition and sets the TZFLG[DCAEVT1] bit. Reset type: SYSRSn
2	OST	R-0/W1S	0h	Force a One-Shot Trip Event via Software 0: Writing of 0 is ignored. Always reads back a 0. 1: Forces a one-shot trip event and sets the TZFLG[OST] bit. Reset type: SYSRSn
1	CBC	R-0/W1S	0h	Force a Cycle-by-Cycle Trip Event via Software 0: Writing of 0 is ignored. Always reads back a 0. 1: Forces a cycle-by-cycle trip event and sets the TZFLG[CBC] bit. Reset type: SYSRSn
0	RESERVED	R-0	0h	Reserved

26.17.2.59 ETSEL Register (Offset = A4h) [Reset = 0h]

ETSEL is shown in Figure 26-151 and described in Table 26-82.

Return to the Summary Table.

Event Trigger Selection Register

Figure 26-151. ETSEL Register

15	14	13	12	11	10	9	8
SOCBEN	SOCBSEL			SOCAEN	SOCASEL		
R/W-0h	R/W-0h			R/W-0h	R/W-0h		
7	6	5	4	3	2	1	0
RESERVED	INTSELCMP	SOCBSELCMP	SOCASELCMP	INTEN	INTSEL		
R-0-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h		

Table 26-82. ETSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SOCBEN	R/W	0h	Enable the ADC Start of Conversion B (EPWMxSOCB) Pulse 0: Disable EPWMxSOCB. 1: Enable EPWMxSOCB pulse. Reset type: SYSRSn
14-12	SOCBSEL	R/W	0h	EPWMxSOCB Selection Options These bits determine when a EPWMxSOCB pulse will be generated. 000: Enable DCBEVT1.soc event 001: Enable event time-base counter equal to zero. (TBCTR = 0x00) 010: Enable event time-base counter equal to period (TBCTR = TBPRD) 011: Enable event time-base counter equal to zero or period (TBCTR = 0x00 or TBCTR = TBPRD). This mode is useful in up-down count mode. 100: Enable event time-base counter equal to CMPA when the timer is incrementing or CMPC when the timer is incrementing 101: Enable event time-base counter equal to CMPA when the timer is decrementing or CMPC when the timer is decrementing 110: Enable event: time-base counter equal to CMPB when the timer is incrementing or CMPD when the timer is incrementing 111: Enable event: time-base counter equal to CMPB when the timer is decrementing or CMPD when the timer is decrementing (*) Event selected is determined by SOCBSELCMP bit. Reset type: SYSRSn
11	SOCAEN	R/W	0h	Enable the ADC Start of Conversion A (EPWMxSOCA) Pulse 0: Disable EPWMxSOCA. 1: Enable EPWMxSOCA pulse. Reset type: SYSRSn

Table 26-82. ETSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10-8	SOCASEL	R/W	0h	EPWMxSOCA Selection Options These bits determine when a EPWMxSOCA pulse will be generated. 000: Enable DCAEVT1.soc event 001: Enable event time-base counter equal to zero. (TBCTR = 0x00) 010: Enable event time-base counter equal to period (TBCTR = TBPRD) 011: Enable event time-base counter equal to zero or period (TBCTR = 0x00 or TBCTR = TBPRD). This mode is useful in up-down count mode. 100: Enable event time-base counter equal to CMPA when the timer is incrementing or CMPC when the timer is incrementing 101: Enable event time-base counter equal to CMPA when the timer is decrementing or CMPC when the timer is decrementing 110: Enable event: time-base counter equal to CMPB when the timer is incrementing or CMPD when the timer is incrementing 111: Enable event: time-base counter equal to CMPB when the timer is decrementing or CMPD when the timer is decrementing (*) Event selected is determined by SOCASELCMP bit. Reset type: SYSRSn
7	RESERVED	R-0	0h	Reserved
6	INTSELCMP	R/W	0h	EPWMxINT Compare Register Selection Options 0: Enable event time-base counter equal to CMPA when the timer is incrementing / Enable event time-base counter equal to CMPA when the timer is decrementing / Enable event: time-base counter equal to CMPB when the timer is incrementing / Enable event: time-base counter equal to CMPB when the timer is decrementing to INTSEL selection mux. 1: Enable event time-base counter equal to CMPC when the timer is incrementing / Enable event time-base counter equal to CMPC when the timer is decrementing / Enable event: time-base counter equal to CMPD when the timer is incrementing / Enable event: time-base counter equal to CMPD when the timer is decrementing to INTSEL selection mux. Reset type: SYSRSn
5	SOCBSELCMP	R/W	0h	EPWMxSOCB Compare Register Selection Options 0: Enable event time-base counter equal to CMPA when the timer is incrementing / Enable event time-base counter equal to CMPA when the timer is decrementing / Enable event: time-base counter equal to CMPB when the timer is incrementing / Enable event: time-base counter equal to CMPB when the timer is decrementing to SOCBSEL selection mux. 1: Enable event time-base counter equal to CMPC when the timer is incrementing / Enable event time-base counter equal to CMPC when the timer is decrementing / Enable event: time-base counter equal to CMPD when the timer is incrementing / Enable event: time-base counter equal to CMPD when the timer is decrementing to SOCBSEL selection mux. Reset type: SYSRSn
4	SOCASELCMP	R/W	0h	EPWMxSOCA Compare Register Selection Options 0: Enable event time-base counter equal to CMPA when the timer is incrementing / Enable event time-base counter equal to CMPA when the timer is decrementing / Enable event: time-base counter equal to CMPB when the timer is incrementing / Enable event: time-base counter equal to CMPB when the timer is decrementing to SOCASEL selection mux. 1: Enable event time-base counter equal to CMPC when the timer is incrementing / Enable event time-base counter equal to CMPC when the timer is decrementing / Enable event: time-base counter equal to CMPD when the timer is incrementing / Enable event: time-base counter equal to CMPD when the timer is decrementing to SOCASEL selection mux. Reset type: SYSRSn

Table 26-82. ETSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	INTEN	R/W	0h	Enable ePWM Interrupt (EPWMx_INT) Generation 0: Disable EPWMx_INT generation 1: Enable EPWMx_INT generation Reset type: SYSRSn
2-0	INTSEL	R/W	0h	ePWM Interrupt (EPWMx_INT) Selection Options 000: Reserved 001: Enable event time-base counter equal to zero. (TBCTR = 0x00) 010: Enable event time-base counter equal to period (TBCTR = TBPRD) 011: Enable event time-base counter equal to zero or period (TBCTR = 0x00 or TBCTR = TBPRD). This mode is useful in up-down count mode. 100: Enable event time-base counter equal to CMPA when the timer is incrementing or CMPC when the timer is incrementing 101: Enable event time-base counter equal to CMPA when the timer is decrementing or CMPC when the timer is decrementing 110: Enable event: time-base counter equal to CMPB when the timer is incrementing or CMPD when the timer is incrementing 111: Enable event: time-base counter equal to CMPB when the timer is decrementing or CMPD when the timer is decrementing (*) Event selected is determined by INTSELCMP bit. Reset type: SYSRSn

26.17.2.60 ETPS Register (Offset = A6h) [Reset = 0h]

ETPS is shown in Figure 26-152 and described in Table 26-83.

Return to the Summary Table.

Event Trigger Pre-Scale Register

Figure 26-152. ETPS Register

15	14	13	12	11	10	9	8
SOCBCNT		SOCBPRD		SOCACNT		SOCAPRD	
R-0h		R/W-0h		R-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		SOCPSSEL	INTPSSEL	INTCNT		INTPRD	
R-0-0h		R/W-0h	R/W-0h	R-0h		R/W-0h	

Table 26-83. ETPS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	SOCBCNT	R	0h	ePWM ADC Start-of-Conversion B Event (EPWMxSOCB) Counter Register These bits indicate how many selected ETSEL[SOCBSEL] events have occurred: 00: No events have occurred. 01: 1 event has occurred. 10: 2 events have occurred. 11: 3 events have occurred. Reset type: SYSRSn
13-12	SOCBPRD	R/W	0h	ePWM ADC Start-of-Conversion B Event (EPWMxSOCB) Period Select These bits determine how many selected ETSEL[SOCBSEL] events need to occur before an EPWMxSOCB pulse is generated. To be generated, the pulse must be enabled (ETSEL[SOCBEN] = 1). The SOCB pulse will be generated even if the status flag is set from a previous start of conversion (ETFLG[SOCB] = 1). Once the SOCB pulse is generated, the ETPS[SOCBCNT] bits will automatically be cleared. 00: Disable the SOCB event counter. No EPWMxSOCB pulse will be generated 01: Generate the EPWMxSOCB pulse on the first event: ETPS[SOCBCNT] = 0,1 10: Generate the EPWMxSOCB pulse on the second event: ETPS[SOCBCNT] = 1,0 11: Generate the EPWMxSOCB pulse on the third event: ETPS[SOCBCNT] = 1,1 Reset type: SYSRSn
11-10	SOCACNT	R	0h	ePWM ADC Start-of-Conversion A Event (EPWMxSOCA) Counter Register These bits indicate how many selected ETSEL[SOCASEL] events have occurred: 00: No events have occurred. 01: 1 event has occurred. 10: 2 events have occurred. 11: 3 events have occurred. Reset type: SYSRSn

Table 26-83. ETPS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9-8	SOCAPRD	R/W	0h	ePWM ADC Start-of-Conversion A Event (EPWMxSOCA) Period Select These bits determine how many selected ETSEL[SOCASEL] events need to occur before an EPWMxSOCA pulse is generated. To be generated, the pulse must be enabled (ETSEL[SOCAEN] = 1). The SOCA pulse will be generated even if the status flag is set from a previous start of conversion (ETFLG[SOCA] = 1). Once the SOCA pulse is generated, the ETPS[SOCACNT] bits will automatically be cleared. 00: Disable the SOCA event counter. No EPWMxSOCA pulse will be generated 01: Generate the EPWMxSOCA pulse on the first event: ETPS[SOCACNT] = 0,1 10: Generate the EPWMxSOCA pulse on the second event: ETPS[SOCACNT] = 1,0 11: Generate the EPWMxSOCA pulse on the third event: ETPS[SOCACNT] = 1,1 Reset type: SYSRSn
7-6	RESERVED	R-0	0h	Reserved
5	SOCPSSEL	R/W	0h	EPWMxSOC A/B Pre-Scale Selection Bits 0: Selects ETPS [SOCACNT/SOCBCNT] and [SOCAPRD/ SOCBPRD] registers to determine frequency of events (SOC pulse once every 0-3 events). 1: Selects ETSOCPS [SOCACNT2/SOCBCNT2] and [SOCAPRD2/ SOCBPRD2] registers to determine frequency of events (SOC pulse once every 0-15 events). Reset type: SYSRSn
4	INTPSSEL	R/W	0h	EPWMxINTn Pre-Scale Selection Bits 0: Selects ETPS [INTCNT, and INTPRD] registers to determine frequency of events (interrupt once every 0-3 events). 1: Selects ETINTPS [INTCNT2, and INTPRD2] registers to determine frequency of events (interrupt once every 0-15 events). Reset type: SYSRSn
3-2	INTCNT	R	0h	ePWM Interrupt Event (EPWMx_INT) Counter Register These bits indicate how many selected ETSEL[INTSEL] events have occurred. These bits are automatically cleared when an interrupt pulse is generated. If interrupts are disabled, ETSEL[INT] = 0 or the interrupt flag is set, ETFLG[INT] = 1, the counter will stop counting events when it reaches the period value ETPS[INTCNT] = ETPS[INTPRD]. 00: No events have occurred. 01: 1 event has occurred. 10: 2 events have occurred. 11: 3 events have occurred. Reset type: SYSRSn

Table 26-83. ETPS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	INTPRD	R/W	0h	ePWM Interrupt (EPWMx_INT) Period Select These bits determine how many selected ETSEL[INTSEL] events need to occur before an interrupt is generated. To be generated, the interrupt must be enabled (ETSEL[INT] = 1). If the interrupt status flag is set from a previous interrupt (ETFLG[INT] = 1) then no interrupt will be generated until the flag is cleared via the ETCLR[INT] bit. This allows for one interrupt to be pending while another is still being serviced. Once the interrupt is generated, the ETPS[INTCNT] bits will automatically be cleared. Writing a INTPRD value that is the same as the current counter value will trigger an interrupt if it is enabled and the status flag is clear. Writing a INTPRD value that is less than the current counter value will result in an undefined state. If a counter event occurs at the same instant as a new zero or non-zero INTPRD value is written, the counter is incremented. 00: Disable the interrupt event counter. No interrupt will be generated and ETFRC[INT] is ignored. 01: Generate an interrupt on the first event INTCNT = 01 (first event) 10: Generate interrupt on ETPS[INTCNT] = 1,0 (second event) 11: Generate interrupt on ETPS[INTCNT] = 1,1 (third event) Reset type: SYSRSn

26.17.2.61 ETFLG Register (Offset = A8h) [Reset = 0h]

ETFLG is shown in Figure 26-153 and described in Table 26-84.

Return to the Summary Table.

Event Trigger Flag Register

Figure 26-153. ETFLG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				SOCB	SOCA	RESERVED	INT
R-0-0h				R-0h	R-0h	R-0-0h	R-0h

Table 26-84. ETFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R-0	0h	Reserved
3	SOCB	R	0h	Latched ePWM ADC Start-of-Conversion A (EPWMxSOCB) Status Flag Unlike the ETFLG[INT] flag, the EPWMxSOCB output will continue to pulse even if the flag bit is set. 0: Indicates no event occurred 1: Indicates that a start of conversion pulse was generated on EPWMxSOCB. The EPWMxSOCB output will continue to be generated even if the flag bit is set. Reset type: SYSRSn
2	SOCA	R	0h	Latched ePWM ADC Start-of-Conversion A (EPWMxSOCA) Status Flag Unlike the ETFLG[INT] flag, the EPWMxSOCA output will continue to pulse even if the flag bit is set. 0: Indicates no event occurred 1: Indicates that a start of conversion pulse was generated on EPWMxSOCA. The EPWMxSOCA output will continue to be generated even if the flag bit is set. Reset type: SYSRSn
1	RESERVED	R-0	0h	Reserved
0	INT	R	0h	Latched ePWM Interrupt (EPWMx_INT) Status Flag 0: Indicates no event occurred 1: Indicates that an ePWMx interrupt (EPWMx_INT) was generated. No further interrupts will be generated until the flag bit is cleared. Up to one interrupt can be pending while the ETFLG[INT] bit is still set. If an interrupt is pending, it will not be generated until after the ETFLG[INT] bit is cleared. Reset type: SYSRSn

26.17.2.62 ETCLR Register (Offset = AAh) [Reset = 0h]

ETCLR is shown in Figure 26-154 and described in Table 26-85.

Return to the Summary Table.

Event Trigger Clear Register

Figure 26-154. ETCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				SOCB	SOCA	RESERVED	INT
R-0-0h				R-0/W1S-0h	R-0/W1S-0h	R-0-0h	R-0/W1S-0h

Table 26-85. ETCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R-0	0h	Reserved
3	SOCB	R-0/W1S	0h	ePWM ADC Start-of-Conversion A (EPWMxSOCB) Flag Clear Bit 0: Writing a 0 has no effect. Always reads back a 0 1: Clears the ETFLG[SOCB] flag bit Reset type: SYSRSn
2	SOCA	R-0/W1S	0h	ePWM ADC Start-of-Conversion A (EPWMxSOCA) Flag Clear Bit 0: Writing a 0 has no effect. Always reads back a 0 1: Clears the ETFLG[SOCA] flag bit Reset type: SYSRSn
1	RESERVED	R-0	0h	Reserved
0	INT	R-0/W1S	0h	ePWM Interrupt (EPWMx_INT) Flag Clear Bit 0: Writing a 0 has no effect. Always reads back a 0 1: Clears the ETFLG[INT] flag bit and enable further interrupts pulses to be generated Reset type: SYSRSn

26.17.2.63 ETFRC Register (Offset = ACh) [Reset = 0h]

ETFRC is shown in Figure 26-155 and described in Table 26-86.

Return to the Summary Table.

Event Trigger Force Register

Figure 26-155. ETFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED				SOCB	SOCA	RESERVED	INT
R-0-0h				R-0/W1S-0h	R-0/W1S-0h	R-0-0h	R-0/W1S-0h

Table 26-86. ETFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R-0	0h	Reserved
3	SOCB	R-0/W1S	0h	SOCB Force Bit The SOCB pulse will only be generated if the event is enabled in the ETSEL register. The ETFLG[SOCB] flag bit will be set regardless. 0: Writing 0 to this bit will be ignored. Always reads back a 0. 1: Generates a pulse on EPWMxSOCB and set the SOCBFLG bit. This bit is used for test purposes. Reset type: SYSRSn
2	SOCA	R-0/W1S	0h	SOCA Force Bit The SOCA pulse will only be generated if the event is enabled in the ETSEL register. The ETFLG[SOCA] flag bit will be set regardless. 0: Writing 0 to this bit will be ignored. Always reads back a 0. 1: Generates a pulse on EPWMxSOCA and set the SOCAFLG bit. This bit is used for test purposes. Reset type: SYSRSn
1	RESERVED	R-0	0h	Reserved
0	INT	R-0/W1S	0h	INT Force Bit The interrupt will only be generated if the event is enabled in the ETSEL register. The INT flag bit will be set regardless. 0: Writing 0 to this bit will be ignored. Always reads back a 0. 1: Generates an interrupt on EPWMxINT and set the INT flag bit. This bit is used for test purposes. Reset type: SYSRSn

26.17.2.64 ETINTPS Register (Offset = AEh) [Reset = 0h]

ETINTPS is shown in Figure 26-156 and described in Table 26-87.

Return to the Summary Table.

Event-Trigger Interrupt Pre-Scale Register

Figure 26-156. ETINTPS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
INTCNT2				INTPRD2			
R-0h				R/W-0h			

Table 26-87. ETINTPS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R-0	0h	Reserved
7-4	INTCNT2	R	0h	EPWMxINT Counter 2 When ETPS[INTPSSEL]=1, these bits indicate how many selected events have occurred: 0000: No events 0001: 1 event 0010: 2 events 0011: 3 events 0100: 4 events ... 1111: 15 events Reset type: SYSRSn
3-0	INTPRD2	R/W	0h	EPWMxINT Period 2 Select When ETPS[INTPSSEL] = 1, these bits select how many selected events need to occur before an interrupt is generated: 0000: Disable counter 0001: Generate interrupt on INTCNT = 1 (first event) 0010: Generate interrupt on INTCNT = 2 (second event) 0011: Generate interrupt on INTCNT = 3 (third event) 0100: Generate interrupt on INTCNT = 4 (fourth event) ... 1111: Generate interrupt on INTCNT = 15 (fifteenth event) Reset type: SYSRSn

26.17.2.65 ETSOCPS Register (Offset = B0h) [Reset = 0h]

ETSOCPS is shown in Figure 26-157 and described in Table 26-88.

Return to the Summary Table.

Event-Trigger SOC Pre-Scale Register

Figure 26-157. ETSOCPS Register

15	14	13	12	11	10	9	8
SOCBCNT2				SOCBPRD2			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
SOCACNT2				SOCAPRD2			
R-0h				R/W-0h			

Table 26-88. ETSOCPS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	SOCBCNT2	R	0h	EPWMxSOCB Counter 2 When ETPS[SOCPSSEL] = 1, these bits indicate how many selected events have occurred: 0000: No events 0001: 1 event 0010: 2 events 0011: 3 events 0100: 4 events ... 1111: 15 events Reset type: SYSRSn
11-8	SOCBPRD2	R/W	0h	EPWMxSOCB Period 2 Select When ETPS[SOCPSSEL] = 1, these bits select how many selected event need to occur before an SOCB pulse is generated: 0000: Disable counter 0001: Generate SOC pulse on SOCBCNT2 = 1 (first event) 0010: Generate SOC pulse on SOCBCNT2 = 2 (second event) 0011: Generate SOC pulse on SOCBCNT2 = 3 (third event) 0100: Generate SOC pulse on SOCBCNT2 = 4 (fourth event) ... 1111: Generate SOC pulse on SOCBCNT2 = 15 (fifteenth event) Reset type: SYSRSn
7-4	SOCACNT2	R	0h	EPWMxSOCA Counter 2 When ETPS[SOCPSSEL] = 1, these bits indicate how many selected events have occurred: 0000: No events 0001: 1 event 0010: 2 events 0011: 3 events 0100: 4 events ... 1111: 15 events Reset type: SYSRSn
3-0	SOCAPRD2	R/W	0h	EPWMxSOCA Period 2 Select When ETPS[SOCPSSEL] = 1, these bits select how many selected event need to occur before an SOCA pulse is generated: 0000: Disable counter 0001: Generate SOC pulse on SOCACNT2 = 1 (first event) 0010: Generate SOC pulse on SOCACNT2 = 2 (second event) 0011: Generate SOC pulse on SOCACNT2 = 3 (third event) 0100: Generate SOC pulse on SOCACNT2 = 4 (fourth event) ... 1111: Generate SOC pulse on SOCACNT2 = 15 (fifteenth event) Reset type: SYSRSn

26.17.2.66 ETCNTINITCTL Register (Offset = B2h) [Reset = 0h]

ETCNTINITCTL is shown in Figure 26-158 and described in Table 26-89.

Return to the Summary Table.

Event-Trigger Counter Initialization Control Register

Figure 26-158. ETCNTINITCTL Register

15	14	13	12	11	10	9	8
SOCBINITEN	SOCAINITEN	INTINITEN	SOCBINITFRC	SOCAINITFRC	INTINITFRC	RESERVED	
R/W-0h	R/W-0h	R/W-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0-0h	
7	6	5	4	3	2	1	0
RESERVED							
R-0-0h							

Table 26-89. ETCNTINITCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SOCBINITEN	R/W	0h	EPWMxSOCB Counter 2 Initialization Enable 0: Has no effect. 1: Enable initialization of EPWMxSOCB counter with contents of ETCNTINIT[SOCBINIT] on a SYNC event or software force. Reset type: SYSRSn
14	SOCAINITEN	R/W	0h	EPWMxSOCA Counter 2 Initialization Enable 0: Has no effect. 1: Enable initialization of EPWMxSOCA counter with contents of ETCNTINIT[SOCAINIT] on a SYNC event or software force. Reset type: SYSRSn
13	INTINITEN	R/W	0h	EPWMxINT Counter 2 Initialization Enable 0: Has no effect. 1: Enable initialization of EPWMxINT counter 2 with contents of ETCNTINIT[INTINIT] on a SYNC event or software force. Reset type: SYSRSn
12	SOCBINITFRC	R-0/W1S	0h	EPWMxSOCB Counter 2 Initialization Force 0: Has no effect. 1: This bit forces the ET EPWMxSOCB counter to be initialized with the contents of ETCNTINIT[SOCBINIT]. Reset type: SYSRSn
11	SOCAINITFRC	R-0/W1S	0h	EPWMxSOCA Counter 2 Initialization Force 0: Has no effect. 1: This bit forces the ET EPWMxSOCA counter to be initialized with the contents of ETCNTINIT[SOCAINIT]. Reset type: SYSRSn
10	INTINITFRC	R-0/W1S	0h	EPWMxINT Counter 2 Initialization Force 0: Has no effect. 1: This bit forces the ET EPWMxINT counter to be initialized with the contents of ETCNTINIT[INTINIT]. Reset type: SYSRSn
9-0	RESERVED	R-0	0h	Reserved

26.17.2.67 ETCNTINIT Register (Offset = B4h) [Reset = 0h]

ETCNTINIT is shown in Figure 26-159 and described in Table 26-90.

Return to the Summary Table.

Event-Trigger Counter Initialization Register

Figure 26-159. ETCNTINIT Register

15	14	13	12	11	10	9	8
RESERVED				SOCBINIT			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
SOCAINIT				INTINIT			
R/W-0h				R/W-0h			

Table 26-90. ETCNTINIT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-8	SOCBINIT	R/W	0h	EPWMxSOCB Counter 2 Initialization Bits The ET EPWMxSOCB counter is initialized with the contents of this register on an ePWM SYNC event or a software force. Reset type: SYSRSn
7-4	SOCAINIT	R/W	0h	EPWMxSOCA Counter 2 Initialization Bits The ET EPWMxSOCA counter is initialized with the contents of this register on an ePWM SYNC event or a software force. Reset type: SYSRSn
3-0	INTINIT	R/W	0h	EPWMxINT Counter 2 Initialization Bits The ET EPWMxINT counter is initialized with the contents of this register on an ePWM SYNC event or a software force. Reset type: SYSRSn

26.17.2.68 DCTRIPSEL Register (Offset = C0h) [Reset = 0h]

DCTRIPSEL is shown in Figure 26-160 and described in Table 26-91.

Return to the Summary Table.

Digital Compare Trip Select Register

Figure 26-160. DCTRIPSEL Register

15	14	13	12	11	10	9	8
DCBLCOMPSEL				DCBHCOMPSEL			
R/W-0h				R/W-0h			
7	6	5	4	3	2	1	0
DCALCOMPSEL				DCAHCOMPSEL			
R/W-0h				R/W-0h			

Table 26-91. DCTRIPSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	DCBLCOMPSEL	R/W	0h	Digital Compare B Low Input Select Bits 0000: TRIPIN1 0001: TRIPIN2 0010: TRIPIN3 0011: TRIPIN4 ... 1011: TRIPIN12 1100: Reserved 1101: TRIPIN14 1110: TRIPIN15 1111: Trip combination input (all trip inputs selected by DCBLTRIPSEL register ORed together) Reset type: SYSRSn
11-8	DCBHCOMPSEL	R/W	0h	Digital Compare B High Input Select Bits 0000: TRIPIN1 0001: TRIPIN2 0010: TRIPIN3 0011: TRIPIN4 ... 1011: TRIPIN12 1100: Reserved 1101: TRIPIN14 1110: TRIPIN15 1111: Trip combination input (all trip inputs selected by DCBHTRIPSEL register ORed together) Reset type: SYSRSn
7-4	DCALCOMPSEL	R/W	0h	Digital Compare A Low Input Select Bits 0000: TRIPIN1 0001: TRIPIN2 0010: TRIPIN3 0011: TRIPIN4 ... 1011: TRIPIN12 1100: Reserved 1101: TRIPIN14 1110: TRIPIN15 1111: Trip combination input (all trip inputs selected by DCALTRIPSEL register ORed together) Reset type: SYSRSn

Table 26-91. DCTRIPSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	DCAHCOMPSEL	R/W	0h	Digital Compare A High Input Select Bits 0000: TRIPIN1 0001: TRIPIN2 0010: TRIPIN3 0011: TRIPIN4 ... 1011: TRIPIN12 1100: Reserved 1101: TRIPIN14 1110: TRIPIN15 1111: Trip combination input (all trip inputs selected by DCAHTRIPSEL register ORed together) Reset type: SYSRSn

26.17.2.69 DCACTL Register (Offset = C3h) [Reset = 0h]

DCACTL is shown in Figure 26-161 and described in Table 26-92.

Return to the Summary Table.

Digital Compare A Control Register

Figure 26-161. DCACTL Register

15	14	13	12	11	10	9	8
EVT2LAT	EVT2LATCLRSEL		EVT2LATSEL	RESERVED		EVT2FRCSYNCSEL	EVT2SRCSEL
R-0h	R/W-0h		R/W-0h	R-0-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
EVT1LAT	EVT1LATCLRSEL		EVT1LATSEL	EVT1SYNCE	EVT1SOCE	EVT1FRCSYNCSEL	EVT1SRCSEL
R-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 26-92. DCACTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	EVT2LAT	R	0h	Indicates the status of DCAEVT2LAT signal. 0 : The DCAEVT2LAT latch is cleared. 1 : The DCAEVT2LAT latch is set. Reset type: SYSRSn
14-13	EVT2LATCLRSEL	R/W	0h	DCAEVT2 Latched clear source select: 00: CNT_ZERO event clears DCAEVT2 latch. 01: PRD_EQ event clears DCAEVT2 latch. 10: CNT_ZERO event or PRD_EQ event clears DCAEVT2 latch. 11: Reserved. Reset type: SYSRSn
12	EVT2LATSEL	R/W	0h	DCAEVT2 Latched signal select: 0: Does not select the DCAEVT2 latched signal as source of DCAEVT2.force. 1: Selects the DCAEVT2 latched signal as source of DCAEVT2.force. Reset type: SYSRSn
11-10	RESERVED	R-0	0h	Reserved
9	EVT2FRCSYNCSEL	R/W	0h	DCAEVT2 Force Synchronization Signal Select 0: Source is synchronized with EPWMCLK 1: Source is passed through asynchronously Reset type: SYSRSn
8	EVT2SRCSEL	R/W	0h	DCAEVT2 Source Signal Select 0: Source Is DCAEVT2 Signal 1: Source Is DCEVTFILT Signal Reset type: SYSRSn
7	EVT1LAT	R	0h	Indicates the status of DCAEVT1LAT signal. 0 : The DCAEVT1LAT latch is cleared. 1 : The DCAEVT1LAT latch is set. Reset type: SYSRSn
6-5	EVT1LATCLRSEL	R/W	0h	DCAEVT1 Latched clear source select: 00: CNT_ZERO event clears DCAEVT1 latch. 01: PRD_EQ event clears DCAEVT1 latch. 10: CNT_ZERO event or PRD_EQ event clears DCAEVT1 latch. 11 : Reserved. Reset type: SYSRSn
4	EVT1LATSEL	R/W	0h	DCAEVT1 Latched signal select: 0: Does not select the DCAEVT1 latched signal as source of DCAEVT1.force. 1: Selects the DCAEVT1 latched signal as source of DCAEVT1.force. Reset type: SYSRSn

Table 26-92. DCACTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	EVT1SYNCE	R/W	0h	DCAEVT1 SYNC, Enable/Disable 0: SYNC Generation Disabled 1: SYNC Generation Enabled Reset type: SYSRSn
2	EVT1SOCE	R/W	0h	DCAEVT1 SOC, Enable/Disable 0: SOC Generation Disabled 1: SOC Generation Enabled Reset type: SYSRSn
1	EVT1FRCSYNCSEL	R/W	0h	DCAEVT1 Force Synchronization Signal Select 0: Source is synchronized with EPWMCLK 1: Source is passed through asynchronously Reset type: SYSRSn
0	EVT1SRCSEL	R/W	0h	DCAEVT1 Source Signal Select 0: Source Is DCAEVT1 Signal 1: Source Is DCEVTFILT Signal Reset type: SYSRSn

26.17.2.70 DCBCTL Register (Offset = C4h) [Reset = 0h]

DCBCTL is shown in Figure 26-162 and described in Table 26-93.

Return to the Summary Table.

Digital Compare B Control Register

Figure 26-162. DCBCTL Register

15	14	13	12	11	10	9	8
EVT2LAT	EVT2LATCLRSEL		EVT2LATSEL	RESERVED		EVT2FRCSYNCSEL	EVT2SRCSEL
R-0h	R/W-0h		R/W-0h	R-0-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
EVT1LAT	EVT1LATCLRSEL		EVT1LATSEL	EVT1SYNCE	EVT1SOCE	EVT1FRCSYNCSEL	EVT1SRCSEL
R-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 26-93. DCBCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	EVT2LAT	R	0h	Indicates the status of DCBEVT2LAT signal. 0 The DCBEVT2LAT latch is cleared. 1 The DCBEVT2LAT latch is set. Reset type: SYSRSn
14-13	EVT2LATCLRSEL	R/W	0h	DCBEVT2 Latched clear source select: 00 CNT_ZERO event clears DCBEVT2 latch. 01 PRD_EQ event clears DCBEVT2 latch. 10 CNT_ZERO event or PRD_EQ event clears DCBEVT2 latch. 11 Reserved. Reset type: SYSRSn
12	EVT2LATSEL	R/W	0h	DCBEVT2 Latched signal select: 0 Does not select the DCBEVT2 latched signal (Refer figure "Modifications to DCBEVT1.force/DCBEVT2.force generation.") as source of DCBEVT2.force. 1 Selects the DCBEVT2 latched signal as source of DCBEVT2.force. Reset type: SYSRSn
11-10	RESERVED	R-0	0h	Reserved
9	EVT2FRCSYNCSEL	R/W	0h	DCBEVT2 Force Synchronization Signal Select 0: Source is synchronized with EPWMCLK 1: Source is passed through asynchronously Reset type: SYSRSn
8	EVT2SRCSEL	R/W	0h	DCBEVT2 Source Signal Select 0: Source Is DCBEVT2 Signal 1: Source Is DCEVTFILT Signal Reset type: SYSRSn
7	EVT1LAT	R	0h	Indicates the status of DCBEVT1LAT signal. 0 The DCBEVT1LAT latch is cleared. 1 The DCBEVT1LAT latch is set. Reset type: SYSRSn
6-5	EVT1LATCLRSEL	R/W	0h	DCBEVT1 Latched clear source select: 00 CNT_ZERO event clears DCBEVT1 latch. 01 PRD_EQ event clears DCBEVT1 latch. 10 CNT_ZERO event or PRD_EQ event clears DCBEVT1 latch. 11 Reserved. Reset type: SYSRSn
4	EVT1LATSEL	R/W	0h	DCBEVT1 Latched signal select: 0 Does not select the DCBEVT1 latched signal (Refer figure "Modifications to DCBEVT1.force/DCBEVT2.force generation.") as source of DCBEVT1.force. 1 Selects the DCBEVT1 latched signal as source of DCBEVT1.force. Reset type: SYSRSn

Table 26-93. DCBCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	EVT1SYNCE	R/W	0h	DCBEVT1 SYNC, Enable/Disable 0: SYNC Generation Disabled 1: SYNC Generation Enabled Reset type: SYSRSn
2	EVT1SOCE	R/W	0h	DCBEVT1 SOC, Enable/Disable 0: SOC Generation Disabled 1: SOC Generation Enabled Reset type: SYSRSn
1	EVT1FRCSYNCSEL	R/W	0h	DCBEVT1 Force Synchronization Signal Select 0: Source is synchronized with EPWMCLK 1: Source is passed through asynchronously Reset type: SYSRSn
0	EVT1SRCSEL	R/W	0h	DCBEVT1 Source Signal Select 0: Source Is DCBEVT1 Signal 1: Source Is DCEVTFILT Signal Reset type: SYSRSn

26.17.2.71 DCFCTL Register (Offset = C7h) [Reset = 0h]

DCFCTL is shown in Figure 26-163 and described in Table 26-94.

Return to the Summary Table.

Digital Compare Filter Control Register

Figure 26-163. DCFCTL Register

15	14	13	12	11	10	9	8
EDGESTATUS			EDGECOUNT			EDGEMODE	
R-0h			R/W-0h			R/W-0h	
7	6	5	4	3	2	1	0
RESERVED	EDGEFILTSEL	PULSESEL		BLANKINV	BLANKE	SRCSEL	
R-0-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	

Table 26-94. DCFCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	EDGESTATUS	R	0h	Edge Status: These bits reflect the total number of edges currently captured. When the value matches the EDGECOUNT, the status bits are set to zero. and a TBCLK wide pulse is generated which can then be output on the DCEVTFILT signal. The edge counter can be reset by writing 000 to the EDGECOUNT value: Reset type: SYSRSn
12-10	EDGECOUNT	R/W	0h	Edge Count: These bits select how many edges to count before generating a TBCLK wide pulse on the DCEVTFILT signal: 000: no edges, reset current EDGESTATUS bits to 0,0,0 001: 1 edge 010: 2 edges 011: 3 edges 100: 4 edges 101: 5 edges 110: 6 edges 111: 7 edges Reset type: SYSRSn
9-8	EDGEMODE	R/W	0h	Edge Mode Select: 00: Low To High Edge 01: High To Low Edge 10: Both Edges 11: Reserved Reset type: SYSRSn
7	RESERVED	R-0	0h	Reserved
6	EDGEFILTSEL	R/W	0h	Edge Filter Select: 0: Edge Filter Not Selected 1: Edge Filter Selected Reset type: SYSRSn
5-4	PULSESEL	R/W	0h	Pulse Select For Blanking & Capture Alignment 00: Time-base counter equal to period (TBCTR = TBPRD) 01: Time-base counter equal to zero (TBCTR = 0x00) 10: Time-base counter equal to zero (TBCTR = 0x00) or period (TBCTR = TBPRD) 11: Reserved Reset type: SYSRSn
3	BLANKINV	R/W	0h	Blanking Window Inversion 0: Blanking window not inverted 1: Blanking window inverted Reset type: SYSRSn

Table 26-94. DCFCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	BLANKE	R/W	0h	Blanking Window Enable/Disable 0: Blanking window is disabled 1: Blanking window is enabled Reset type: SYSRSn
1-0	SRCSEL	R/W	0h	Filter Block Signal Source Select 00: Source Is DCAEVT1 Signal 01: Source Is DCAEVT2 Signal 10: Source Is DCBEVT1 Signal 11: Source Is DCBEVT2 Signal Reset type: SYSRSn

26.17.2.72 DCCAPCTL Register (Offset = C8h) [Reset = 0h]

DCCAPCTL is shown in Figure 26-164 and described in Table 26-95.

Return to the Summary Table.

Digital Compare Capture Control Register

Figure 26-164. DCCAPCTL Register

15	14	13	12	11	10	9	8
CAPMODE	CAPCLR	CAPSTS	RESERVED				
R/W-0h	R-0/W1S-0h	R-0h	R-0-0h				
7	6	5	4	3	2	1	0
RESERVED						SHDWMODE	CAPE
R-0-0h						R/W-0h	R/W-0h

Table 26-95. DCCAPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	CAPMODE	R/W	0h	Counter Capture Mode 0: When a DCEVTFILT occurs and the counter capture is enabled, then the current TBCNT value is captured in the active register. When the respective trip event occurs, further trip (capture) events are ignored until the next PRD_eq or CNT_zero event (as selected by the PULSESEL bit in the DCFCTL register) re-triggers the capture mechanism. If active mode is enabled, via SHDWMODE bit in DCCAPCTL register, CPU reads of this register will return the active register value. If shadow mode is enabled, via SHDWMODE bit in DCCAPCTL register, the active register is copied to the shadow register on the PRD_eq or CNT_zero event (whichever is selected by PULSESEL bit in DCFCTL register). CPU reads of this register will return the shadow register value. 1: When a DCEVTFILT occurs and the counter capture is enabled, then the current TBCNT value is captured in the active register. When the respective trip event occurs - it will set the CAPSTS flag and further trip (capture) events are ignored until this bit is cleared. CAPSTS can be cleared by writing to CAPCLR bit in DCCAPCTL register and it re-triggers the capture mechanism. If active mode is enabled, via SHDWMODE bit in DCCAPCTL register, CPU reads of this register will return the active register value. If shadow mode is enabled, via SHDWMODE bit in DCCAPCTL register, the active register is copied to the shadow register on the PRD_eq or CNT_zero event (whichever is selected by PULSESEL bit in DCFCTL register). CPU reads of this register will return the shadow register value. Reset type: SYSRSn
14	CAPCLR	R-0/W1S	0h	DC Capture Latched Status Clear Flag 0: Writing a 0 has no effect. 1: Writing a 1 will clear this CAPSTS (set) condition. Reset type: SYSRSn
13	CAPSTS	R	0h	Latched Status Flag for Capture Event 0: No DC capture event occurred. 1: A DC capture event has occurred. Reset type: SYSRSn
12-2	RESERVED	R-0	0h	Reserved

Table 26-95. DCCAPCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	SHDWMODE	R/W	0h	TBCTR Counter Capture Shadow Select Mode 0: Enable shadow mode. The DCCAP active register is copied to shadow register on a TBCTR = TBPRD or TBCTR = zero event as defined by the DCFCTL[PULSESEL] bit. CPU reads of the DCCAP register will return the shadow register contents. 1: Active Mode. In this mode the shadow register is disabled. CPU reads from the DCCAP register will always return the active register contents. Reset type: SYSRSn
0	CAPE	R/W	0h	TBCTR Counter Capture Enable/Disable 0: Disable the time-base counter capture. 1: Enable the time-base counter capture. Reset type: SYSRSn

26.17.2.73 DCFOFFSET Register (Offset = C9h) [Reset = 0h]

DCFOFFSET is shown in Figure 26-165 and described in Table 26-96.

Return to the Summary Table.

Digital Compare Filter Offset Register

Figure 26-165. DCFOFFSET Register

15	14	13	12	11	10	9	8
DCFOFFSET							
R/W-0h							
7	6	5	4	3	2	1	0
DCFOFFSET							
R/W-0h							

Table 26-96. DCFOFFSET Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DCFOFFSET	R/W	0h	Blanking Window Offset These 16-bits specify the number of TBCLK cycles from the blanking window reference to the point when the blanking window is applied. The blanking window reference is either period or zero as defined by the DCFCTL[PULSESEL] bit. This offset register is shadowed and the active register is loaded at the reference point defined by DCFCTL[PULSESEL]. The offset counter is also initialized and begins to count down when the active register is loaded. When the counter expires, the blanking window is applied. If the blanking window is currently active, then the blanking window counter is restarted. Reset type: SYSRSn

26.17.2.74 DCFOFFSETCNT Register (Offset = CAh) [Reset = 0h]

DCFOFFSETCNT is shown in Figure 26-166 and described in Table 26-97.

Return to the Summary Table.

Digital Compare Filter Offset Counter Register

Figure 26-166. DCFOFFSETCNT Register

15	14	13	12	11	10	9	8
DCFOFFSETCNT							
R-0h							
7	6	5	4	3	2	1	0
DCFOFFSETCNT							
R-0h							

Table 26-97. DCFOFFSETCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DCFOFFSETCNT	R	0h	Blanking Offset Counter These 16-bits are read only and indicate the current value of the offset counter. The counter counts down to zero and then stops until it is re-loaded on the next period or zero event as defined by the DCFCTL[PULSESEL] bit. The offset counter is not affected by the free/soft emulation bits. That is, it will always continue to count down if the device is halted by a emulation stop. Reset type: SYSRSn

26.17.2.75 DCFWINDOW Register (Offset = CBh) [Reset = 0h]

DCFWINDOW is shown in Figure 26-167 and described in Table 26-98.

Return to the Summary Table.

Digital Compare Filter Window Register

Figure 26-167. DCFWINDOW Register

15	14	13	12	11	10	9	8
DCFWINDOW							
R/W-0h							
7	6	5	4	3	2	1	0
DCFWINDOW							
R/W-0h							

Table 26-98. DCFWINDOW Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DCFWINDOW	R/W	0h	Blanking Window Width 00h: No blanking window is generated. 01-FFFFh: Specifies the width of the blanking window in TBCLK cycles. The blanking window begins when the offset counter expires. When this occurs, the window counter is loaded and begins to count down. If the blanking window is currently active and the offset counter expires, the blanking window counter is not restarted and the blanking window is cut short prematurely. Care should be taken to avoid this situation. The blanking window can cross a PWM period boundary. Reset type: SYSRSn

26.17.2.76 DCFWINDOWCNT Register (Offset = CCh) [Reset = 0h]

DCFWINDOWCNT is shown in Figure 26-168 and described in Table 26-99.

Return to the Summary Table.

Digital Compare Filter Window Counter Register

Figure 26-168. DCFWINDOWCNT Register

15	14	13	12	11	10	9	8
DCFWINDOWCNT							
R-0h							
7	6	5	4	3	2	1	0
DCFWINDOWCNT							
R-0h							

Table 26-99. DCFWINDOWCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DCFWINDOWCNT	R	0h	Blanking Window Counter These 16 bits are read only and indicate the current value of the window counter. The counter counts down to zero and then stops until it is re-loaded when the offset counter reaches zero again. Reset type: SYSRSn

26.17.2.77 DCCAP Register (Offset = CFh) [Reset = 0h]

DCCAP is shown in Figure 26-169 and described in Table 26-100.

Return to the Summary Table.

Digital Compare Counter Capture Register

Figure 26-169. DCCAP Register

15	14	13	12	11	10	9	8
DCCAP							
R-0h							
7	6	5	4	3	2	1	0
DCCAP							
R-0h							

Table 26-100. DCCAP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DCCAP	R	0h	Digital Compare Time-Base Counter Capture To enable time-base counter capture, set the DCCAPCLT[CAPE] bit to 1. If enabled, reflects the value of the time-base counter (TBCTR) on the low to high edge transition of a filtered (DCEVTFLT) event. Further capture events are ignored until the next period or zero as selected by the DCFCTL[PULSESEL] bit. Shadowing of DCCAP is enabled and disabled by the DCCAPCTL[SHDWMODE] bit. By default this register is shadowed. - If DCCAPCTL[SHDWMODE] = 0, then the shadow is enabled. In this mode, the active register is copied to the shadow register on the TBCTR = TBPRD or TBCTR = zero as defined by the DCFCTL[PULSESEL] bit. CPU reads of this register will return the shadow register value. - If DCCAPCTL[SHDWMODE] = 1, then the shadow register is disabled. In this mode, CPU reads will return the active register value. The active and shadow registers share the same memory map address. Reset type: SYSRSn

26.17.2.78 DCAHTRIPSEL Register (Offset = D2h) [Reset = 0h]

DCAHTRIPSEL is shown in Figure 26-170 and described in Table 26-101.

Return to the Summary Table.

Digital Compare AH Trip Select

Figure 26-170. DCAHTRIPSEL Register

15	14	13	12	11	10	9	8
RESERVED	TRIPINPUT15	TRIPINPUT14	RESERVED	TRIPINPUT12	TRIPINPUT11	TRIPINPUT10	TRIPINPUT9
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TRIPINPUT8	TRIPINPUT7	TRIPINPUT6	TRIPINPUT5	TRIPINPUT4	TRIPINPUT3	TRIPINPUT2	TRIPINPUT1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 26-101. DCAHTRIPSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	TRIPINPUT15	R/W	0h	TRIP Input 15 0: Trip Input 15 not selected as combinational ORed input 1: Trip Input 15 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
13	TRIPINPUT14	R/W	0h	TRIP Input 14 0: Trip Input 14 not selected as combinational ORed input 1: Trip Input 14 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
12	RESERVED	R/W	0h	Reserved
11	TRIPINPUT12	R/W	0h	TRIP Input 12 0: Trip Input 12 not selected as combinational ORed input 1: Trip Input 12 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
10	TRIPINPUT11	R/W	0h	TRIP Input 11 0: Trip Input 11 not selected as combinational ORed input 1: Trip Input 11 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
9	TRIPINPUT10	R/W	0h	TRIP Input 10 0: Trip Input 10 not selected as combinational ORed input 1: Trip Input 10 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
8	TRIPINPUT9	R/W	0h	TRIP Input 9 0: Trip Input 9 not selected as combinational ORed input 1: Trip Input 9 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
7	TRIPINPUT8	R/W	0h	TRIP Input 8 0: Trip Input 8 not selected as combinational ORed input 1: Trip Input 8 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
6	TRIPINPUT7	R/W	0h	TRIP Input 7 0: Trip Input 7 not selected as combinational ORed input 1: Trip Input 7 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
5	TRIPINPUT6	R/W	0h	TRIP Input 6 0: Trip Input 6 not selected as combinational ORed input 1: Trip Input 6 selected as combinational ORed input to DCAH mux Reset type: SYSRSn

Table 26-101. DCAHTRIPSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	TRIPINPUT5	R/W	0h	TRIP Input 5 0: Trip Input 5 not selected as combinational ORed input 1: Trip Input 5 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
3	TRIPINPUT4	R/W	0h	TRIP Input 4 0: Trip Input 4 not selected as combinational ORed input 1: Trip Input 4 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
2	TRIPINPUT3	R/W	0h	TRIP Input 3 0: Trip Input 3 not selected as combinational ORed input 1: Trip Input 3 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
1	TRIPINPUT2	R/W	0h	TRIP Input 2 0: Trip Input 2 not selected as combinational ORed input 1: Trip Input 2 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
0	TRIPINPUT1	R/W	0h	TRIP Input 1 0: Trip Input 1 not selected as combinational ORed input 1: Trip Input 1 selected as combinational ORed input to DCAH mux Reset type: SYSRSn

26.17.2.79 DCALTRIPSEL Register (Offset = D3h) [Reset = 0h]

DCALTRIPSEL is shown in Figure 26-171 and described in Table 26-102.

Return to the Summary Table.

Digital Compare AL Trip Select

Figure 26-171. DCALTRIPSEL Register

15	14	13	12	11	10	9	8
RESERVED	TRIPINPUT15	TRIPINPUT14	RESERVED	TRIPINPUT12	TRIPINPUT11	TRIPINPUT10	TRIPINPUT9
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TRIPINPUT8	TRIPINPUT7	TRIPINPUT6	TRIPINPUT5	TRIPINPUT4	TRIPINPUT3	TRIPINPUT2	TRIPINPUT1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 26-102. DCALTRIPSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	TRIPINPUT15	R/W	0h	TRIP Input 15 0: Trip Input 15 not selected as combinational ORed input 1: Trip Input 15 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
13	TRIPINPUT14	R/W	0h	TRIP Input 14 0: Trip Input 14 not selected as combinational ORed input 1: Trip Input 14 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
12	RESERVED	R/W	0h	Reserved
11	TRIPINPUT12	R/W	0h	TRIP Input 12 0: Trip Input 12 not selected as combinational ORed input 1: Trip Input 12 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
10	TRIPINPUT11	R/W	0h	TRIP Input 11 0: Trip Input 11 not selected as combinational ORed input 1: Trip Input 11 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
9	TRIPINPUT10	R/W	0h	TRIP Input 10 0: Trip Input 10 not selected as combinational ORed input 1: Trip Input 10 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
8	TRIPINPUT9	R/W	0h	TRIP Input 9 0: Trip Input 9 not selected as combinational ORed input 1: Trip Input 9 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
7	TRIPINPUT8	R/W	0h	TRIP Input 8 0: Trip Input 8 not selected as combinational ORed input 1: Trip Input 8 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
6	TRIPINPUT7	R/W	0h	TRIP Input 7 0: Trip Input 7 not selected as combinational ORed input 1: Trip Input 7 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
5	TRIPINPUT6	R/W	0h	TRIP Input 6 0: Trip Input 6 not selected as combinational ORed input 1: Trip Input 6 selected as combinational ORed input to DCAL mux Reset type: SYSRSn

Table 26-102. DCALTRIPSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	TRIPINPUT5	R/W	0h	TRIP Input 5 0: Trip Input 5 not selected as combinational ORed input 1: Trip Input 5 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
3	TRIPINPUT4	R/W	0h	TRIP Input 4 0: Trip Input 4 not selected as combinational ORed input 1: Trip Input 4 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
2	TRIPINPUT3	R/W	0h	TRIP Input 3 0: Trip Input 3 not selected as combinational ORed input 1: Trip Input 3 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
1	TRIPINPUT2	R/W	0h	TRIP Input 2 0: Trip Input 2 not selected as combinational ORed input 1: Trip Input 2 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
0	TRIPINPUT1	R/W	0h	TRIP Input 1 0: Trip Input 1 not selected as combinational ORed input 1: Trip Input 1 selected as combinational ORed input to DCAL mux Reset type: SYSRSn

26.17.2.80 DCBHTRIPSEL Register (Offset = D4h) [Reset = 0h]

DCBHTRIPSEL is shown in Figure 26-172 and described in Table 26-103.

Return to the Summary Table.

Digital Compare BH Trip Select

Figure 26-172. DCBHTRIPSEL Register

15	14	13	12	11	10	9	8
RESERVED	TRIPINPUT15	TRIPINPUT14	RESERVED	TRIPINPUT12	TRIPINPUT11	TRIPINPUT10	TRIPINPUT9
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TRIPINPUT8	TRIPINPUT7	TRIPINPUT6	TRIPINPUT5	TRIPINPUT4	TRIPINPUT3	TRIPINPUT2	TRIPINPUT1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 26-103. DCBHTRIPSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	TRIPINPUT15	R/W	0h	TRIP Input 15 0: Trip Input 15 not selected as combinational ORed input 1: Trip Input 15 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
13	TRIPINPUT14	R/W	0h	TRIP Input 14 0: Trip Input 14 not selected as combinational ORed input 1: Trip Input 14 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
12	RESERVED	R/W	0h	Reserved
11	TRIPINPUT12	R/W	0h	TRIP Input 12 0: Trip Input 12 not selected as combinational ORed input 1: Trip Input 12 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
10	TRIPINPUT11	R/W	0h	TRIP Input 11 0: Trip Input 11 not selected as combinational ORed input 1: Trip Input 11 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
9	TRIPINPUT10	R/W	0h	TRIP Input 10 0: Trip Input 10 not selected as combinational ORed input 1: Trip Input 10 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
8	TRIPINPUT9	R/W	0h	TRIP Input 9 0: Trip Input 9 not selected as combinational ORed input 1: Trip Input 9 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
7	TRIPINPUT8	R/W	0h	TRIP Input 8 0: Trip Input 8 not selected as combinational ORed input 1: Trip Input 8 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
6	TRIPINPUT7	R/W	0h	TRIP Input 7 0: Trip Input 7 not selected as combinational ORed input 1: Trip Input 7 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
5	TRIPINPUT6	R/W	0h	TRIP Input 6 0: Trip Input 6 not selected as combinational ORed input 1: Trip Input 6 selected as combinational ORed input to DCBH mux Reset type: SYSRSn

Table 26-103. DCBHTRIPSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	TRIPINPUT5	R/W	0h	TRIP Input 5 0: Trip Input 5 not selected as combinational ORed input 1: Trip Input 5 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
3	TRIPINPUT4	R/W	0h	TRIP Input 4 0: Trip Input 4 not selected as combinational ORed input 1: Trip Input 4 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
2	TRIPINPUT3	R/W	0h	TRIP Input 3 0: Trip Input 3 not selected as combinational ORed input 1: Trip Input 3 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
1	TRIPINPUT2	R/W	0h	TRIP Input 2 0: Trip Input 2 not selected as combinational ORed input 1: Trip Input 2 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
0	TRIPINPUT1	R/W	0h	TRIP Input 1 0: Trip Input 1 not selected as combinational ORed input 1: Trip Input 1 selected as combinational ORed input to DCBH mux Reset type: SYSRSn

26.17.2.81 DCBLTRIPSEL Register (Offset = D5h) [Reset = 0h]

DCBLTRIPSEL is shown in Figure 26-173 and described in Table 26-104.

Return to the Summary Table.

Digital Compare BL Trip Select

Figure 26-173. DCBLTRIPSEL Register

15	14	13	12	11	10	9	8
RESERVED	TRIPINPUT15	TRIPINPUT14	RESERVED	TRIPINPUT12	TRIPINPUT11	TRIPINPUT10	TRIPINPUT9
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TRIPINPUT8	TRIPINPUT7	TRIPINPUT6	TRIPINPUT5	TRIPINPUT4	TRIPINPUT3	TRIPINPUT2	TRIPINPUT1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 26-104. DCBLTRIPSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	TRIPINPUT15	R/W	0h	TRIP Input 15 0: Trip Input 15 not selected as combinational ORed input 1: Trip Input 15 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
13	TRIPINPUT14	R/W	0h	TRIP Input 14 0: Trip Input 14 not selected as combinational ORed input 1: Trip Input 14 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
12	RESERVED	R/W	0h	Reserved
11	TRIPINPUT12	R/W	0h	TRIP Input 12 0: Trip Input 12 not selected as combinational ORed input 1: Trip Input 12 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
10	TRIPINPUT11	R/W	0h	TRIP Input 11 0: Trip Input 11 not selected as combinational ORed input 1: Trip Input 11 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
9	TRIPINPUT10	R/W	0h	TRIP Input 10 0: Trip Input 10 not selected as combinational ORed input 1: Trip Input 10 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
8	TRIPINPUT9	R/W	0h	TRIP Input 9 0: Trip Input 9 not selected as combinational ORed input 1: Trip Input 9 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
7	TRIPINPUT8	R/W	0h	TRIP Input 8 0: Trip Input 8 not selected as combinational ORed input 1: Trip Input 8 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
6	TRIPINPUT7	R/W	0h	TRIP Input 7 0: Trip Input 7 not selected as combinational ORed input 1: Trip Input 7 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
5	TRIPINPUT6	R/W	0h	TRIP Input 6 0: Trip Input 6 not selected as combinational ORed input 1: Trip Input 6 selected as combinational ORed input to DCAL mux Reset type: SYSRSn

Table 26-104. DCBLTRIPSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	TRIPINPUT5	R/W	0h	TRIP Input 5 0: Trip Input 5 not selected as combinational ORed input 1: Trip Input 5 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
3	TRIPINPUT4	R/W	0h	TRIP Input 4 0: Trip Input 4 not selected as combinational ORed input 1: Trip Input 4 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
2	TRIPINPUT3	R/W	0h	TRIP Input 3 0: Trip Input 3 not selected as combinational ORed input 1: Trip Input 3 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
1	TRIPINPUT2	R/W	0h	TRIP Input 2 0: Trip Input 2 not selected as combinational ORed input 1: Trip Input 2 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
0	TRIPINPUT1	R/W	0h	TRIP Input 1 0: Trip Input 1 not selected as combinational ORed input 1: Trip Input 1 selected as combinational ORed input to DCAL mux Reset type: SYSRSn

26.17.2.82 EPWMLOCK Register (Offset = FAh) [Reset = 0h]

EPWMLOCK is shown in Figure 26-174 and described in Table 26-105.

Return to the Summary Table.

EPWM Lock Register

Figure 26-174. EPWMLOCK Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			DCLOCK	TZCLRLOCK	TZCFGLOCK	GLLOCK	HRLOCK
R-0h			R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h

Table 26-105. EPWMLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to this register succeeds only if this field is written with a value of 0xa5a5 Note: [1] Due to this KEY, only 32-bit writes will succeed (provided the KEY matches). 16-bit writes to the upper or lower half of this register will be ignored Reset type: SYSRSn
15-5	RESERVED	R	0h	Reserved
4	DCLOCK	R/WOnce	0h	0:Digital Compare registers from 0xC0 to 0xD9 offsets are protected by EALLOW. 1: Digital Compare registers from 0xC0 and 0xD9 offsets are locked and not writable. Reset type: SYSRSn
3	TZCLRLOCK	R/WOnce	0h	0:Trip Zone registers from 0x97 to 0x9B offsets are protected by EALLOW. 1: Trip Zone registers from 0x97 and 0x9B offsets are locked and not writable. Reset type: SYSRSn
2	TZCFGLOCK	R/WOnce	0h	0:TripZone registers from 0x80 to 0x8D and TZTRIPOUTSEL at 0x9D offsets are protected by EALLOW. 1: TripZone registers from 0x80 and 0x8D and TZTRIPOUTSEL at 0x9D offsets are locked and not writable. Reset type: SYSRSn
1	GLLOCK	R/WOnce	0h	0:Global Load registers from 0x34 to 0x35 offsets are protected by EALLOW. 1: Global Load registers from 0x34 to 0x35 offsets are locked and not writable Reset type: SYSRSn
0	HRLOCK	R/WOnce	0h	0: HRPWM registers from 0x20 to 0x2D offsets are protected by EALLOW 1: HRPWM registers from 0x20 and 0x2D offsets are locked and not writable. Reset type: SYSRSn

26.17.2.83 HWVDELVAL Register (Offset = FDh) [Reset = 0h]

HWVDELVAL is shown in Figure 26-175 and described in Table 26-106.

Return to the Summary Table.

Hardware Valley Mode Delay Register

Figure 26-175. HWVDELVAL Register

15	14	13	12	11	10	9	8
HWVDELVAL							
R-0h							
7	6	5	4	3	2	1	0
HWVDELVAL							
R-0h							

Table 26-106. HWVDELVAL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	HWVDELVAL	R	0h	Hardware Valley Delay Value Register This read only register reflects the hardware delay value calculated by the equations defined in VCAPCTL[VDELAYDIV]. This reflects the latest value from the hardware calculations and can change every time valley capture sequence is triggered and VCAP1 and VCAP2 values are updated. Reset type: SYSRSn

26.17.2.84 VCNTVAL Register (Offset = FEh) [Reset = 0h]

VCNTVAL is shown in Figure 26-176 and described in Table 26-107.

Return to the Summary Table.

Hardware Valley Counter Register

Figure 26-176. VCNTVAL Register

15	14	13	12	11	10	9	8
VCNTVAL							
R-0h							
7	6	5	4	3	2	1	0
VCNTVAL							
R-0h							

Table 26-107. VCNTVAL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	VCNTVAL	R	0h	Valley Time Base Counter Register This register reflects the captured VCNT value upon occurrence of STOPEDGE selected in VCNTCFG register. Reset type: SYSRSn

26.17.3 SYNC_SOC_REGS Registers

Table 26-108 lists the memory-mapped registers for the SYNC_SOC_REGS registers. All register offset addresses not listed in Table 26-108 should be considered as reserved locations and the register contents should not be modified.

Table 26-108. SYNC_SOC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	SYNCSELECT	Sync Input and Output Select Register	EALLOW	Go
2h	ADCSOCOUTSELECT	External ADC (Off Chip) SOC Select Register	EALLOW	Go
4h	SYNCSOCLOCK	SYNCSEL and EXTADCSOC Select Lock register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 26-109 shows the codes that are used for access types in this section.

Table 26-109. SYNC_SOC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

26.17.3.1 SYNCSELECT Register (Offset = 0h) [Reset = 0h]

SYNCSELECT is shown in Figure 26-177 and described in Table 26-110.

Return to the Summary Table.

Sync Input and Output Select Register

Figure 26-177. SYNCSELECT Register

31	30	29	28	27	26	25	24
RESERVED				SYNCOUT			
R/W-0h				R/W-0h			
23	22	21	20	19	18	17	16
RESERVED				RESERVED			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED	RESERVED			RESERVED			RESERVED
R-0-0h	R/W-0h			R/W-0h			R/W-0h
7	6	5	4	3	2	1	0
RESERVED		RESERVED			RESERVED		
R/W-0h		R/W-0h			R/W-0h		

Table 26-110. SYNCSELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R/W	0h	Reserved

Table 26-110. SYNCSELECT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
28-24	SYNCOUT	R/W	0h	Select Syncout Source: 00000: EPWM1SYNCOUT selected to drive the SYNCOUT pin. 00001: EPWM2SYNCOUT selected to drive the SYNCOUT pin. 00010: EPWM3SYNCOUT selected to drive the SYNCOUT pin. 00011: EPWM4SYNCOUT selected to drive the SYNCOUT pin. 00100: EPWM5SYNCOUT selected to drive the SYNCOUT pin. 00101: EPWM6SYNCOUT selected to drive the SYNCOUT pin. 00110: EPWM7SYNCOUT selected to drive the SYNCOUT pin. 00111: EPWM8SYNCOUT selected to drive the SYNCOUT pin. 01000: EPWM9SYNCOUT selected to drive the SYNCOUT pin. 01001: EPWM10SYNCOUT selected to drive the SYNCOUT pin. 01010: EPWM11SYNCOUT selected to drive the SYNCOUT pin. 01011: EPWM12SYNCOUT selected to drive the SYNCOUT pin. 01100: EPWM13SYNCOUT selected to drive the SYNCOUT pin. 01101: EPWM14SYNCOUT selected to drive the SYNCOUT pin. 01110: EPWM15SYNCOUT selected to drive the SYNCOUT pin. 01111: EPWM16SYNCOUT selected to drive the SYNCOUT pin. 10000: Reserved 10001: Reserved 10010: Reserved 10011: Reserved 10100: Reserved 10101: Reserved 10110: Reserved 10111: Reserved 11000: ECAP1SYNCOUT selected to drive the SYNCOUT pin. 11001: ECAP2SYNCOUT selected to drive the SYNCOUT pin. 11010: ECAP3SYNCOUT selected to drive the SYNCOUT pin. 11011: ECAP4SYNCOUT selected to drive the SYNCOUT pin. 11100: ECAP5SYNCOUT selected to drive the SYNCOUT pin. 11101: ECAP6SYNCOUT selected to drive the SYNCOUT pin. 11110: ECAP7SYNCOUT selected to drive the SYNCOUT pin. 11111: Reserved Notes: [1] Reserved position defaults to 00 selection Reset type: CPU1.SYSRSn
23-20	RESERVED	R/W	0h	Reserved
19-16	RESERVED	R/W	0h	Reserved
15	RESERVED	R-0	0h	Reserved
14-12	RESERVED	R/W	0h	Reserved
11-9	RESERVED	R/W	0h	Reserved
8-6	RESERVED	R/W	0h	Reserved
5-3	RESERVED	R/W	0h	Reserved
2-0	RESERVED	R/W	0h	Reserved

26.17.3.2 ADCSOCOUTSELECT Register (Offset = 2h) [Reset = 0h]

ADCSOCOUTSELECT is shown in Figure 26-178 and described in Table 26-111.

Return to the Summary Table.

External ADC (Off Chip) SOC Select Register

Figure 26-178. ADCSOCOUTSELECT Register

31	30	29	28	27	26	25	24
PWM16SOCBEN	PWM15SOCBEN	PWM14SOCBEN	PWM13SOCBEN	PWM12SOCBEN	PWM11SOCBEN	PWM10SOCBEN	PWM9SOCBEN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
PWM8SOCBEN	PWM7SOCBEN	PWM6SOCBEN	PWM5SOCBEN	PWM4SOCBEN	PWM3SOCBEN	PWM2SOCBEN	PWM1SOCBEN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
PWM16SOCAEN	PWM15SOCAEN	PWM14SOCAEN	PWM13SOCAEN	PWM12SOCAEN	PWM11SOCAEN	PWM10SOCAEN	PWM9SOCAEN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
PWM8SOCAEN	PWM7SOCAEN	PWM6SOCAEN	PWM5SOCAEN	PWM4SOCAEN	PWM3SOCAEN	PWM2SOCAEN	PWM1SOCAEN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 26-111. ADCSOCOUTSELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	PWM16SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
30	PWM15SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
29	PWM14SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
28	PWM13SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
27	PWM12SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
26	PWM11SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
25	PWM10SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn

Table 26-111. ADCSOCOUTSELECT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24	PWM9SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
23	PWM8SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
22	PWM7SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
21	PWM6SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
20	PWM5SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
19	PWM4SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
18	PWM3SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
17	PWM2SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
16	PWM1SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: CPU1.SYSRSn
15	PWM16SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
14	PWM15SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
13	PWM14SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
12	PWM13SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
11	PWM12SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn

Table 26-111. ADCSOCOUTSELECT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	PWM11SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
9	PWM10SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
8	PWM9SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
7	PWM8SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
6	PWM7SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
5	PWM6SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
4	PWM5SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
3	PWM4SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
2	PWM3SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
1	PWM2SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn
0	PWM1SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: CPU1.SYSRSn

26.17.3.3 SYNCSOCLOCK Register (Offset = 4h) [Reset = 0h]

SYNCSOCLOCK is shown in Figure 26-179 and described in Table 26-112.

Return to the Summary Table.

SYNCSEL and EXTADCSOC Select Lock register

Figure 26-179. SYNCSOCLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						ADCSOCOUTSELECT	SYNCSELECT
R-0-0h						R/WSonce-0h	R/WSonce-0h

Table 26-112. SYNCSOCLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-2	RESERVED	R-0	0h	Reserved
1	ADCSOCOUTSELECT	R/WSonce	0h	ADCSOCOUTSELECT Register Lock bit: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any bit in this register, once set can only be creaed through a CPU1.SYSRSn. Write of 0 to any bit of this regtister has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: CPU1.SYSRSn
0	SYNCSELECT	R/WSonce	0h	SYNCSELECT Register Lock bit: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any bit in this register, once set can only be creaed through a CPU1.SYSRSn. Write of 0 to any bit of this regtister has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: CPU1.SYSRSn

26.17.4 Register to Driverlib Function Mapping

26.17.4.1 EPWM Registers to Driverlib Functions

Table 26-113. EPWM Registers to Driverlib Functions

File	Driverlib Function
TBCTL	
epwm.c	EPWM_setEmulationMode
epwm.h	EPWM_setCountModeAfterSync
epwm.h	EPWM_setClockPrescaler
epwm.h	EPWM_forceSyncPulse
epwm.h	EPWM_setOneShotSyncOutTrigger
epwm.h	EPWM_setPeriodLoadMode
epwm.h	EPWM_enablePhaseShiftLoad
epwm.h	EPWM_disablePhaseShiftLoad
epwm.h	EPWM_setTimeBaseCounterMode
epwm.h	EPWM_selectPeriodLoadEvent
epwm.h	EPWM_enableOneShotSync
epwm.h	EPWM_disableOneShotSync
epwm.h	EPWM_startOneShotSync
TBCTL2	
epwm.h	EPWM_selectPeriodLoadEvent
epwm.h	EPWM_enableOneShotSync
epwm.h	EPWM_disableOneShotSync
epwm.h	EPWM_startOneShotSync
SYNCINSEL	
epwm.h	EPWM_setSyncInPulseSource
TBCTR	
epwm.h	EPWM_setTimeBaseCounter
epwm.h	EPWM_getTimeBaseCounterValue
TBSTS	
epwm.h	EPWM_getTimeBaseCounterOverflowStatus
epwm.h	EPWM_clearTimeBaseCounterOverflowEvent
epwm.h	EPWM_getSyncStatus
epwm.h	EPWM_clearSyncEvent
epwm.h	EPWM_getTimeBaseCounterDirection
SYNCOUTEN	
epwm.h	EPWM_enableSyncOutPulseSource
epwm.h	EPWM_disableSyncOutPulseSource
TBCTL3	
epwm.h	EPWM_setOneShotSyncOutTrigger
CMPCTL	
epwm.h	EPWM_setCounterCompareShadowLoadMode
epwm.h	EPWM_disableCounterCompareShadowLoadMode
epwm.h	EPWM_getCounterCompareShadowStatus
CMPCTL2	
epwm.h	EPWM_setCounterCompareShadowLoadMode
epwm.h	EPWM_disableCounterCompareShadowLoadMode

Table 26-113. EPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
DBCTL	
epwm.h	EPWM_setDeadBandOutputSwapMode
epwm.h	EPWM_setDeadBandDelayMode
epwm.h	EPWM_setDeadBandDelayPolarity
epwm.h	EPWM_setRisingEdgeDeadBandDelayInput
epwm.h	EPWM_setFallingEdgeDeadBandDelayInput
epwm.h	EPWM_setDeadBandControlShadowLoadMode
epwm.h	EPWM_disableDeadBandControlShadowLoadMode
epwm.h	EPWM_setRisingEdgeDelayCountShadowLoadMode
epwm.h	EPWM_disableRisingEdgeDelayCountShadowLoadMode
epwm.h	EPWM_setFallingEdgeDelayCountShadowLoadMode
epwm.h	EPWM_disableFallingEdgeDelayCountShadowLoadMode
epwm.h	EPWM_setDeadBandCounterClock
DBCTL2	
epwm.h	EPWM_setDeadBandControlShadowLoadMode
epwm.h	EPWM_disableDeadBandControlShadowLoadMode
AQCTL	
epwm.h	EPWM_setActionQualifierShadowLoadMode
epwm.h	EPWM_disableActionQualifierShadowLoadMode
epwm.h	EPWM_setActionQualifierAction
epwm.h	EPWM_setActionQualifierActionComplete
epwm.h	EPWM_setAdditionalActionQualifierActionComplete
AQTSRCSEL	
epwm.h	EPWM_setActionQualifierT1TriggerSource
epwm.h	EPWM_setActionQualifierT2TriggerSource
PCCTL	
epwm.h	EPWM_enableChopper
epwm.h	EPWM_disableChopper
epwm.h	EPWM_setChopperDutyCycle
epwm.h	EPWM_setChopperFreq
epwm.h	EPWM_setChopperFirstPulseWidth
VCAPCTL	
epwm.h	EPWM_enableValleyCapture
epwm.h	EPWM_disableValleyCapture
epwm.h	EPWM_startValleyCapture
epwm.h	EPWM_setValleyTriggerSource
epwm.h	EPWM_enableValleyHWDelay
epwm.h	EPWM_disableValleyHWDelay
epwm.h	EPWM_setValleyDelayDivider
VCNTCFG	
epwm.h	EPWM_setValleyTriggerEdgeCounts
epwm.h	EPWM_getValleyEdgeStatus
HRCNFG	
-	
HRPWR	

Table 26-113. EPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
HRMSTEP	
-	
HRCNFG2	
-	
HRPCTL	
-	
TRREM	
-	
GLDCTL	
epwm.h	EPWM_enableGlobalLoad
epwm.h	EPWM_disableGlobalLoad
epwm.h	EPWM_setGlobalLoadTrigger
epwm.h	EPWM_setGlobalLoadEventPrescale
epwm.h	EPWM_getGlobalLoadEventCount
epwm.h	EPWM_disableGlobalLoadOneShotMode
epwm.h	EPWM_enableGlobalLoadOneShotMode
epwm.h	EPWM_setGlobalLoadOneShotLatch
epwm.h	EPWM_forceGlobalLoadOneShotEvent
GLDCFG	
epwm.h	EPWM_enableGlobalLoadRegisters
epwm.h	EPWM_disableGlobalLoadRegisters
XLINK	
epwm.h	EPWM_setupEPWMLinks
AQCTLA	
epwm.h	EPWM_setActionQualifierAction
epwm.h	EPWM_setActionQualifierActionComplete
epwm.h	EPWM_setAdditionalActionQualifierActionComplete
AQCTLA2	
epwm.h	EPWM_setActionQualifierAction
epwm.h	EPWM_setAdditionalActionQualifierActionComplete
AQCTLB	
-	See AQCTLA
AQCTLB2	
-	See AQCTLA2
AQSFRC	
epwm.h	EPWM_setActionQualifierContSWForceShadowMode
epwm.h	EPWM_setActionQualifierSWAction
epwm.h	EPWM_forceActionQualifierSWAction
AQCSFRC	
epwm.h	EPWM_setActionQualifierContSWForceAction
DBREDHR	
-	
DBRED	
epwm.h	EPWM_setRisingEdgeDelayCount

Table 26-113. EPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
DBFEDHR	
-	
DBFED	
epwm.h	EPWM_setFallingEdgeDelayCount
TBPHS	
epwm.h	EPWM_setPhaseShift
TBPRDHR	
-	
TBPRD	
epwm.h	EPWM_setTimeBasePeriod
epwm.h	EPWM_getTimeBasePeriod
CMPA	
epwm.h	EPWM_setCounterCompareValue
epwm.h	EPWM_getCounterCompareValue
CMPB	
-	See CMPA
CMPC	
epwm.h	EPWM_setCounterCompareShadowLoadMode
epwm.h	EPWM_disableCounterCompareShadowLoadMode
epwm.h	EPWM_getCounterCompareShadowStatus
CMPD	
-	See CMPC
GLDCTL2	
epwm.h	EPWM_setGlobalLoadOneShotLatch
epwm.h	EPWM_forceGlobalLoadOneShotEvent
SWVDELVAL	
epwm.h	EPWM_setValleySWDelayValue
TZSEL	
epwm.h	EPWM_enableTripZoneSignals
epwm.h	EPWM_disableTripZoneSignals
TZDCSEL	
epwm.h	EPWM_setTripZoneDigitalCompareEventCondition
TZCTL	
epwm.h	EPWM_enableTripZoneAdvAction
epwm.h	EPWM_disableTripZoneAdvAction
epwm.h	EPWM_setTripZoneAction
epwm.h	EPWM_setTripZoneAdvAction
epwm.h	EPWM_setTripZoneAdvDigitalCompareActionA
epwm.h	EPWM_setTripZoneAdvDigitalCompareActionB
TZCTL2	
epwm.h	EPWM_enableTripZoneAdvAction
epwm.h	EPWM_disableTripZoneAdvAction
epwm.h	EPWM_setTripZoneAdvAction
epwm.h	EPWM_setTripZoneAdvDigitalCompareActionA
epwm.h	EPWM_setTripZoneAdvDigitalCompareActionB

Table 26-113. EPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
TZCTLDCA	
epwm.h	EPWM_setTripZoneAdvDigitalCompareActionA
TZCTLDCB	
epwm.h	EPWM_setTripZoneAdvDigitalCompareActionB
TZEINT	
epwm.h	EPWM_enableTripZoneInterrupt
epwm.h	EPWM_disableTripZoneInterrupt
TZFLG	
epwm.h	EPWM_getTripZoneFlagStatus
TZCBCFLG	
epwm.h	EPWM_getCycleByCycleTripZoneFlagStatus
TZOSTFLG	
epwm.h	EPWM_getOneShotTripZoneFlagStatus
TZCLR	
epwm.h	EPWM_selectCycleByCycleTripZoneClearEvent
epwm.h	EPWM_clearTripZoneFlag
TZCBCCLR	
epwm.h	EPWM_clearCycleByCycleTripZoneFlag
TZOSTCLR	
epwm.h	EPWM_clearOneShotTripZoneFlag
TZFRC	
epwm.h	EPWM_forceTripZoneEvent
ETSEL	
epwm.h	EPWM_enableInterrupt
epwm.h	EPWM_disableInterrupt
epwm.h	EPWM_setInterruptSource
epwm.h	EPWM_enableADCTrigger
epwm.h	EPWM_disableADCTrigger
epwm.h	EPWM_setADCTriggerSource
ETPS	
epwm.h	EPWM_setInterruptEventCount
epwm.h	EPWM_setADCTriggerEventPrescale
ETFLG	
epwm.h	EPWM_getEventTriggerInterruptStatus
epwm.h	EPWM_getADCTriggerFlagStatus
ETCLR	
epwm.h	EPWM_clearEventTriggerInterruptFlag
epwm.h	EPWM_clearADCTriggerFlag
ETFRC	
epwm.h	EPWM_forceEventTriggerInterrupt
epwm.h	EPWM_forceADCTrigger
ETINTPS	
epwm.h	EPWM_setInterruptEventCount
epwm.h	EPWM_getInterruptEventCount
ETSOCPS	

Table 26-113. EPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
epwm.h	EPWM_setADCTriggerEventPrescale
epwm.h	EPWM_getADCTriggerEventCount
ETCNTINITCTL	
epwm.h	EPWM_enableInterruptEventCountInit
epwm.h	EPWM_disableInterruptEventCountInit
epwm.h	EPWM_forceInterruptEventCountInit
epwm.h	EPWM_enableADCTriggerEventCountInit
epwm.h	EPWM_disableADCTriggerEventCountInit
epwm.h	EPWM_forceADCTriggerEventCountInit
ETCNTINIT	
epwm.h	EPWM_enableInterruptEventCountInit
epwm.h	EPWM_disableInterruptEventCountInit
epwm.h	EPWM_forceInterruptEventCountInit
epwm.h	EPWM_setInterruptEventCountInitValue
epwm.h	EPWM_enableADCTriggerEventCountInit
epwm.h	EPWM_disableADCTriggerEventCountInit
epwm.h	EPWM_forceADCTriggerEventCountInit
epwm.h	EPWM_setADCTriggerEventCountInitValue
DCTRIPSEL	
epwm.h	EPWM_selectDigitalCompareTripInput
epwm.h	EPWM_enableDigitalCompareTripCombinationInput
DCACTL	
epwm.h	EPWM_setDigitalCompareEventSource
epwm.h	EPWM_setDigitalCompareEventSyncMode
epwm.h	EPWM_enableDigitalCompareADCTrigger
epwm.h	EPWM_disableDigitalCompareADCTrigger
epwm.h	EPWM_enableDigitalCompareSyncEvent
epwm.h	EPWM_disableDigitalCompareSyncEvent
epwm.h	EPWM_setDigitalCompareCBCLatchMode
epwm.h	EPWM_selectDigitalCompareCBCLatchClearEvent
epwm.h	EPWM_getDigitalCompareCBCLatchStatus
DCBCTL	
-	See DCACTL
DCFCTL	
epwm.h	EPWM_enableDigitalCompareBlankingWindow
epwm.h	EPWM_disableDigitalCompareBlankingWindow
epwm.h	EPWM_enableDigitalCompareWindowInverseMode
epwm.h	EPWM_disableDigitalCompareWindowInverseMode
epwm.h	EPWM_setDigitalCompareBlankingEvent
epwm.h	EPWM_setDigitalCompareFilterInput
epwm.h	EPWM_enableDigitalCompareEdgeFilter
epwm.h	EPWM_disableDigitalCompareEdgeFilter
epwm.h	EPWM_setDigitalCompareEdgeFilterMode
epwm.h	EPWM_setDigitalCompareEdgeFilterEdgeCount
epwm.h	EPWM_getDigitalCompareEdgeFilterEdgeCount

Table 26-113. EPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
epwm.h	EPWM_getDigitalCompareEdgeFilterEdgeStatus
DCCAPCTL	
epwm.h	EPWM_enableDigitalCompareCounterCapture
epwm.h	EPWM_disableDigitalCompareCounterCapture
epwm.h	EPWM_setDigitalCompareCounterShadowMode
epwm.h	EPWM_getDigitalCompareCaptureStatus
DCFOFFSET	
epwm.h	EPWM_setDigitalCompareWindowOffset
epwm.h	EPWM_getDigitalCompareBlankingWindowOffsetCount
DCFOFFSETCNT	
epwm.h	EPWM_getDigitalCompareBlankingWindowOffsetCount
DCFWINDOW	
epwm.h	EPWM_setDigitalCompareWindowLength
epwm.h	EPWM_getDigitalCompareBlankingWindowLengthCount
DCFWINDOWCNT	
epwm.h	EPWM_getDigitalCompareBlankingWindowLengthCount
DCCAP	
epwm.h	EPWM_enableDigitalCompareCounterCapture
epwm.h	EPWM_disableDigitalCompareCounterCapture
epwm.h	EPWM_setDigitalCompareCounterShadowMode
epwm.h	EPWM_getDigitalCompareCaptureStatus
epwm.h	EPWM_getDigitalCompareCaptureCount
DCAHTRIPSEL	
epwm.h	EPWM_enableDigitalCompareTripCombinationInput
epwm.h	EPWM_disableDigitalCompareTripCombinationInput
DCALTRIPSEL	
-	See DCAHTRIPSEL
DCBHTRIPSEL	
-	See DCAHTRIPSEL
DCBLTRIPSEL	
-	See DCAHTRIPSEL
LOCK	
epwm.h	EPWM_lockRegisters
HWVDELVAL	
epwm.h	EPWM_getValleyHWDelay
VCNTVAL	
epwm.h	EPWM_getValleyCount

26.17.4.2 HRPWM Registers to Driverlib Functions

Table 26-114. HRPWM Registers to Driverlib Functions

File	Driverlib Function
TBCTL	
-	
TBCTL2	
-	

Table 26-114. HRPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
EPWMSYNCINSEL	
-	
TBCTR	
-	
TBSTS	
-	
EPWMSYNCOUTEN	
-	
TBCTL3	
-	
CMPCTL	
-	
CMPCTL2	
-	
DBCTL	
-	
DBCTL2	
-	
AQCTL	
-	
AQTSRCSEL	
-	
PCCTL	
-	
VCAPCTL	
-	
VCNTCFG	
-	
HRCNFG	
hrpwm.h	HRPWM_setMEPEdgeSelect
hrpwm.h	HRPWM_setMEPControlMode
hrpwm.h	HRPWM_setCounterCompareShadowLoadEvent
hrpwm.h	HRPWM_setOutputSwapMode
hrpwm.h	HRPWM_setChannelBOutputPath
hrpwm.h	HRPWM_enableAutoConversion
hrpwm.h	HRPWM_disableAutoConversion
hrpwm.h	HRPWM_setDeadbandMEPEdgeSelect
hrpwm.h	HRPWM_setRisingEdgeDelayLoadMode
hrpwm.h	HRPWM_setFallingEdgeDelayLoadMode
HRPWR	
-	
HRMSTEP	
hrpwm.h	HRPWM_setMEPStep
HRCNFG2	
hrpwm.h	HRPWM_setDeadbandMEPEdgeSelect

Table 26-114. HRPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
hrpwm.h	HRPWM_setRisingEdgeDelayLoadMode
hrpwm.h	HRPWM_setFallingEdgeDelayLoadMode
HRPCTL	
hrpwm.h	HRPWM_enablePeriodControl
hrpwm.h	HRPWM_disablePeriodControl
hrpwm.h	HRPWM_enablePhaseShiftLoad
hrpwm.h	HRPWM_disablePhaseShiftLoad
hrpwm.h	HRPWM_setSyncPulseSource
TRREM	
hrpwm.h	HRPWM_setTranslatorRemainder
GLDCTL	
-	
GLDCFG	
-	
EPWMXLINK	
-	
AQCTLA	
-	
AQCTLA2	
-	
AQCTLB	
-	
AQCTLB2	
-	
AQSFRC	
-	
AQCSFRC	
-	
DBREDHR	
hrpwm.h	HRPWM_setRisingEdgeDelay
hrpwm.h	HRPWM_setHiResRisingEdgeDelayOnly
DBRED	
hrpwm.h	HRPWM_setRisingEdgeDelay
hrpwm.h	HRPWM_setHiResRisingEdgeDelayOnly
DBFEDHR	
hrpwm.h	HRPWM_setFallingEdgeDelay
hrpwm.h	HRPWM_setHiResFallingEdgeDelayOnly
DBFED	
hrpwm.h	HRPWM_setFallingEdgeDelay
hrpwm.h	HRPWM_setHiResFallingEdgeDelayOnly
TBPHS	
hrpwm.h	HRPWM_setPhaseShift
hrpwm.h	HRPWM_setHiResPhaseShiftOnly
TBPRDHR	
hrpwm.h	HRPWM_setTimeBasePeriod

Table 26-114. HRPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
hrpwm.h	HRPWM_setHiResTimeBasePeriodOnly
hrpwm.h	HRPWM_getTimeBasePeriod
hrpwm.h	HRPWM_getHiResTimeBasePeriodOnly
TBPRD	
hrpwm.h	HRPWM_setTimeBasePeriod
hrpwm.h	HRPWM_setHiResTimeBasePeriodOnly
hrpwm.h	HRPWM_getTimeBasePeriod
hrpwm.h	HRPWM_getHiResTimeBasePeriodOnly
CMPA	
hrpwm.h	HRPWM_setCounterCompareValue
hrpwm.h	HRPWM_setHiResCounterCompareValueOnly
hrpwm.h	HRPWM_getCounterCompareValue
hrpwm.h	HRPWM_getHiResCounterCompareValueOnly
CMPB	
hrpwm.h	HRPWM_setCounterCompareValue
hrpwm.h	HRPWM_setHiResCounterCompareValueOnly
hrpwm.h	HRPWM_getCounterCompareValue
hrpwm.h	HRPWM_getHiResCounterCompareValueOnly
CMPC	
-	
CMPD	
-	
GLDCTL2	
-	
SWVDELVAL	
-	
TZSEL	
-	
TZDCSEL	
-	
TZCTL	
-	
TZCTL2	
-	
TZCTLDCA	
-	
TZCTLDCB	
-	
TZEINT	
-	
TZFLG	
-	
TZCBCFLG	
-	
TZOSTFLG	

Table 26-114. HRPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
TZCLR	
-	
TZCBCCLR	
-	
TZOSTCLR	
-	
TZFRC	
-	
ETSEL	
-	
ETPS	
-	
ETFLG	
-	
ETCLR	
-	
ETFRC	
-	
ETINTPS	
-	
ETSOCPS	
-	
ETCNTINITCTL	
-	
ETCNTINIT	
-	
DCTRIPSEL	
-	
DCACTL	
-	
DCBCTL	
-	
DCFCTL	
-	
DCCAPCTL	
-	
DCFOFFSET	
-	
DCFOFFSETCNT	
-	
DCFWINDOW	
-	
DCFWINDOWCNT	
-	

Table 26-114. HRPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
DCCAP	
-	
DCAHTRIPSEL	
-	
DCALTRIPSEL	
-	
DCBHTRIPSEL	
-	
DCBLTRIPSEL	
-	
EPWMLOCK	
hrpwm.h	HRPWM_lockRegisters
HWVDELVAL	
-	
VCNTVAL	
-	

This page intentionally left blank.

Chapter 27

Enhanced Quadrature Encoder Pulse (eQEP)

The enhanced Quadrature Encoder Pulse (eQEP) module described here is a Type 2 eQEP. See the *C2000 Real-Time Control Peripheral Reference Guide* for a list of all devices with a module of the same type to determine the differences between types and for a list of device-specific differences within a type.

The enhanced quadrature encoder pulse (eQEP) module is used for direct interface with a linear or rotary incremental encoder to get position, direction, and speed information from a rotating machine for use in a high-performance motion and position-control system.

27.1 Introduction

An incremental encoder disk is patterned with a track of slots along the periphery, as shown in Figure 27-1. These slots create an alternating pattern of dark and light lines. The disk count is defined as the number of dark and light line pairs that occur per revolution (lines per revolution). As a rule, a second track is added to generate a signal that occurs once per revolution (index signal: QEPI), which can be used to indicate an absolute position. Encoder manufacturers identify the index pulse using different terms such as index, marker, home position, and zero reference

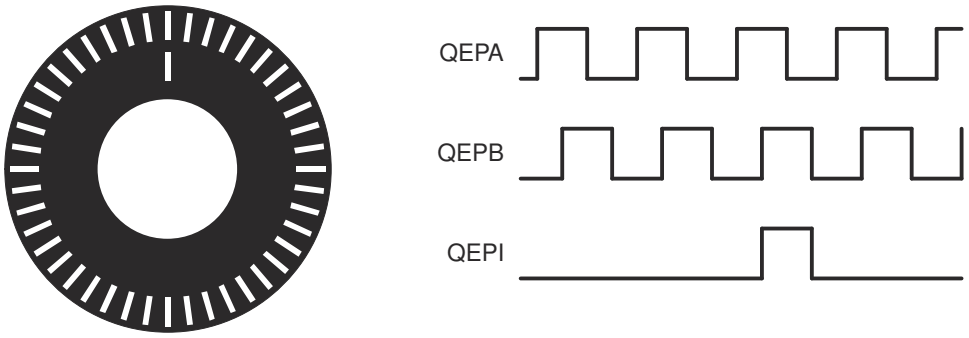
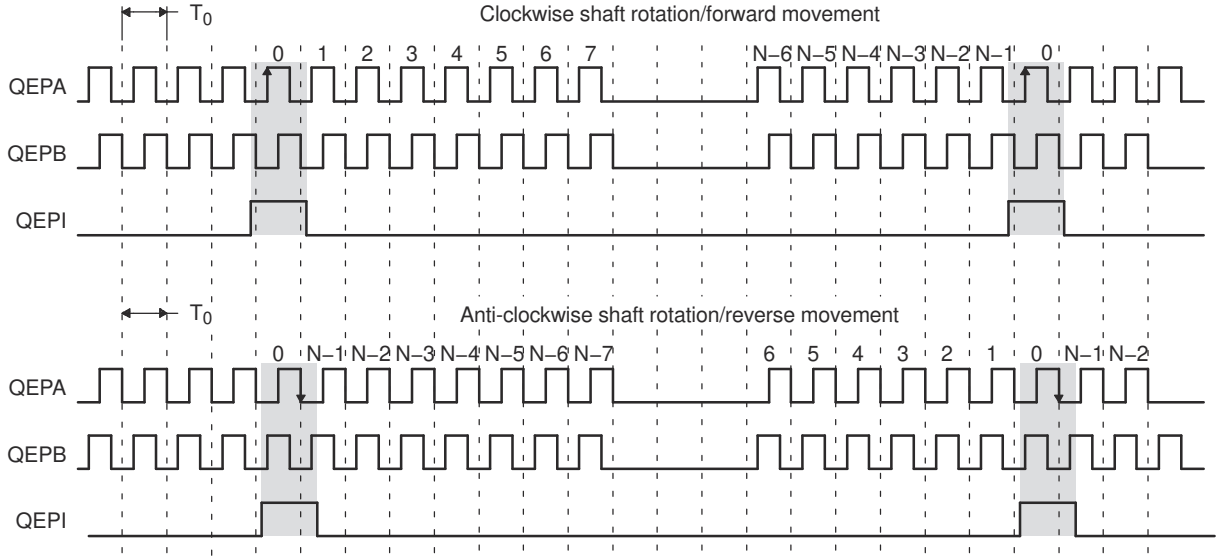


Figure 27-1. Optical Encoder Disk

To derive direction information, the lines on the disk are read out by two different photo-elements that "look" at the disk pattern with a mechanical shift of 1/4 the pitch of a line pair between them. This shift is detected with a reticle or mask that restricts the view of the photo-element to the desired part of the disk lines. As the disk rotates, the two photo-elements generate signals that are shifted 90° out of phase from each other. These are commonly called the quadrature QEPA and QEPB signals. The clockwise direction for most encoders is defined as the QEPA channel going positive before the QEPB channel and conversely, as shown in Figure 27-2.



Legend: N = lines per revolution

Figure 27-2. QEP Encoder Output Signal for Forward/Reverse Movement

The encoder wheel typically makes one revolution for every revolution of the motor, or the wheel can be at a geared rotation ratio with respect to the motor. Therefore, the frequency of the digital signal coming from the QEPA and QEPB outputs varies proportionally with the velocity of the motor. For example, a 2000-line encoder directly coupled to a motor running at 5000 revolutions-per-minute (rpm) results in a frequency of 166.6 kHz, so

by measuring the frequency of either the QEPA or QEPB output, the processor can determine the velocity of the motor.

Quadrature encoders from different manufacturers come with two forms of index pulse (gated index pulse or ungated index pulse) as shown in Figure 27-3. A nonstandard form of index pulse is ungated. In the ungated configuration, the index edges are not necessarily coincident with A and B signals. The gated index pulse is aligned to any of the four quadrature edges and width of the index pulse and can be equal to a quarter, half, or full period of the quadrature signal.

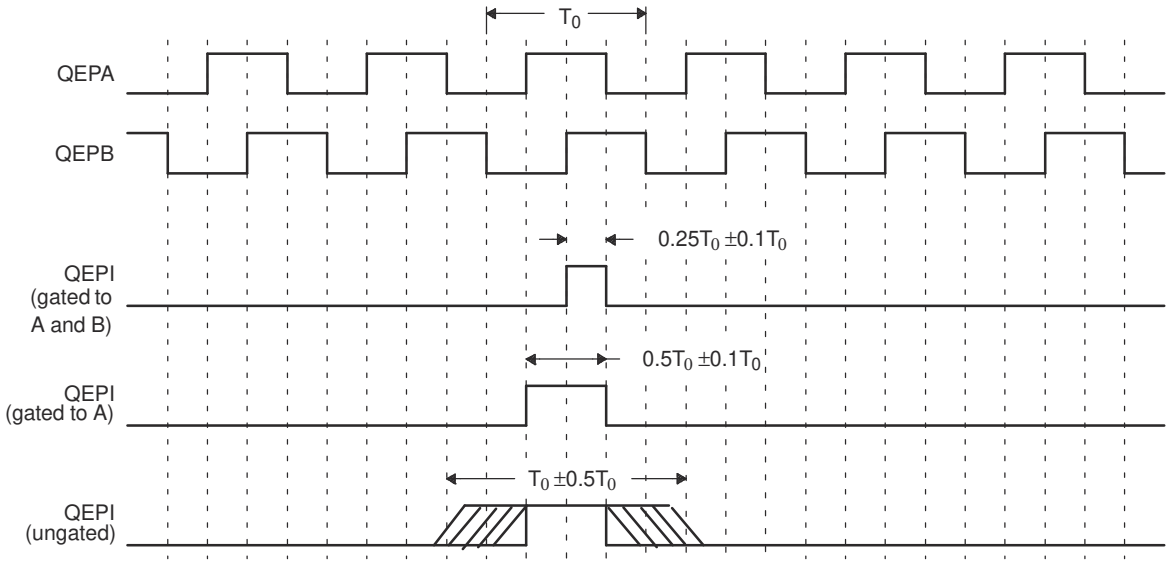


Figure 27-3. Index Pulse Example

Some typical applications of shaft encoders include robotics and computer input in the form of a mouse. Inside your mouse you can see where the mouse ball spins a pair of axles (a left/right, and an up/down axle). These axles are connected to optical shaft encoders that effectively tell the computer how fast and in what direction the mouse is moving.

General Issues: Estimating velocity from a digital position sensor is a cost-effective strategy in motor control. Two different first order approximations for velocity can be written as:

$$v(k) \approx \frac{x(k) - x(k-1)}{T} = \frac{\Delta X}{T} \tag{11}$$

$$v(k) \approx \frac{X}{t(k) - t(k-1)} = \frac{X}{\Delta T} \tag{12}$$

where:

- v(k) = Velocity at time instant k
- x(k) = Position at time instant k
- x(k-1) = Position at time instant k-1
- T = Fixed unit time or inverse of velocity calculation rate
- ΔX = Incremental position movement in unit time
- t(k) = Time instant "k"
- t(k-1) = Time instant "k-1"
- X = Fixed unit position
- ΔT = Incremental time elapsed for unit position movement

Equation 11 is the conventional approach to velocity estimation and requires a time base to provide a unit time event for velocity calculation. Unit time is basically the inverse of the velocity calculation rate.

The encoder count (position) is read once during each unit time event. The quantity [x(k) - x(k-1)] is formed by subtracting the previous reading from the current reading. Then the velocity estimate is computed by multiplying by the known constant 1/T (where T is the constant time between unit time events and is known in advance).

Estimation based on Equation 11 has an inherent accuracy limit directly related to the resolution of the position sensor and the unit time period T. For example, consider a 500 line-per-revolution quadrature encoder with a velocity calculation rate of 400 Hz. When used for position, the quadrature encoder gives a four-fold increase in resolution; in this case, 2000 counts-per-revolution. The minimum rotation that can be detected is, therefore, 0.0005 revolutions, which gives a velocity resolution of 12 rpm when sampled at 400 Hz. While this resolution can be satisfactory at moderate or high speeds, for example 1% error at 1200 rpm, this resolution clearly proves inadequate at low speeds. In fact, at speeds below 12 rpm, the speed estimate is erroneously zero much of the time.

At low speed, Equation 12 provides a more accurate approach. It requires a position sensor that outputs a fixed interval pulse train, such as the aforementioned quadrature encoder. The width of each pulse is defined by motor speed for a given sensor resolution. Equation 12 can be used to calculate motor speed by measuring the elapsed time between successive quadrature pulse edges. However, this method suffers from the opposite limitation, as does Equation 11. A combination of relatively large motor speeds and high sensor resolution makes the time interval ΔT small, and thus more greatly influenced by the timer resolution. This can introduce considerable error into high-speed estimates.

For systems with a large speed range (that is, speed estimation is needed at both low and high speeds), one approach is to use Equation 12 at low speed and have the DSP software switch over to Equation 11 when the motor speed rises above some specified threshold.

27.1.1 EQEP Related Collateral

Foundational Materials

- C2000 Academy - EQEP
- Interfacing with Quadrature Encoders (Video)
- *Real-Time Control Reference Guide*
  - Refer to the Encoders section

Getting Started Materials

- *C2000™ Position Manager PTO API Reference Guide Application Report*

Expert Materials

- *CW/CCW Support on the C2000 eQEP Module Application Report*

27.2 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

For proper operation of the eQEP module, input GPIO pins must be configured using the GPxQSELn registers for synchronous input mode (with or without qualification). The asynchronous mode cannot be used for eQEP input pins. The internal pullups can be configured in the GPyPUD register.

See the *General-Purpose Input/Output (GPIO)* chapter for more details on GPIO mux and settings.

27.3 Description

This section provides the eQEP inputs, memory map, and functional description.

27.3.1 EQEP Inputs

The eQEP inputs include two pins for quadrature-clock mode or direction-count mode, an index (or 0 marker), and a strobe input. The eQEP module requires that the QEPA, QEPB, and QEPI inputs are synchronized to SYSCLK prior to entering the module. The application code can enable the synchronous GPIO input feature on any eQEP-enabled GPIO pins (see the *General-Purpose Input/Output (GPIO)* chapter for more details).

- **QEPA/XCLK and QEPB/XDIR**

  These two pins can be used in quadrature-clock mode or direction-count mode.
  - Quadrature-clock Mode

    The eQEP encoders provide two square wave signals (A and B) 90 electrical degrees out of phase. This phase relationship is used to determine the direction of rotation of the input shaft and number of eQEP pulses from the index position to derive the relative position information. For forward or clockwise rotation, QEPA signal leads QEPB signal and conversely. The quadrature decoder uses these two inputs to generate quadrature-clock and direction signals.
  - Direction-count Mode

    In direction-count mode, direction and clock signals are provided directly from the external source. Some position encoders have this type of output instead of quadrature output. The QEPA pin provides the clock input and the QEPB pin provides the direction input.

- **QEPI: Index or Zero Marker**

  The eQEP encoder uses an index signal to assign an absolute start position from which position information is incrementally encoded using quadrature pulses. This pin is connected to the index output of the eQEP encoder to optionally reset the position counter for each revolution. This signal can be used to initialize or latch the position counter on the occurrence of a desired event on the index pin.

- **QEPS: Strobe Input**

  This general-purpose strobe signal can initialize or latch the position counter on the occurrence of a desired event on the strobe pin. This signal is typically connected to a sensor or limit switch to notify that the motor has reached a defined position.

Input signals to the eQEP (QEPA, QEPB, QEPI and QEPS) can come from multiple sources; that is, device pin, CMPSSx, or PWMXBARx. One typical used case is if SinCos transducers are used in the motor control system to estimate the position of motor shaft and Index signal is coming from traditional rotary encoder, source of the eQEP signals (QEPA, QEPB and QEPI) can be configured as output of CMPSSx which decodes the Sin, Cos and Index signals. Figure 27-4 illustrates the use case.

Selection of the source of Input signals (QEPA, QEPB, and QEPI) is user-configurable through the QEPSRCSEL register as shown in Table 27-1.

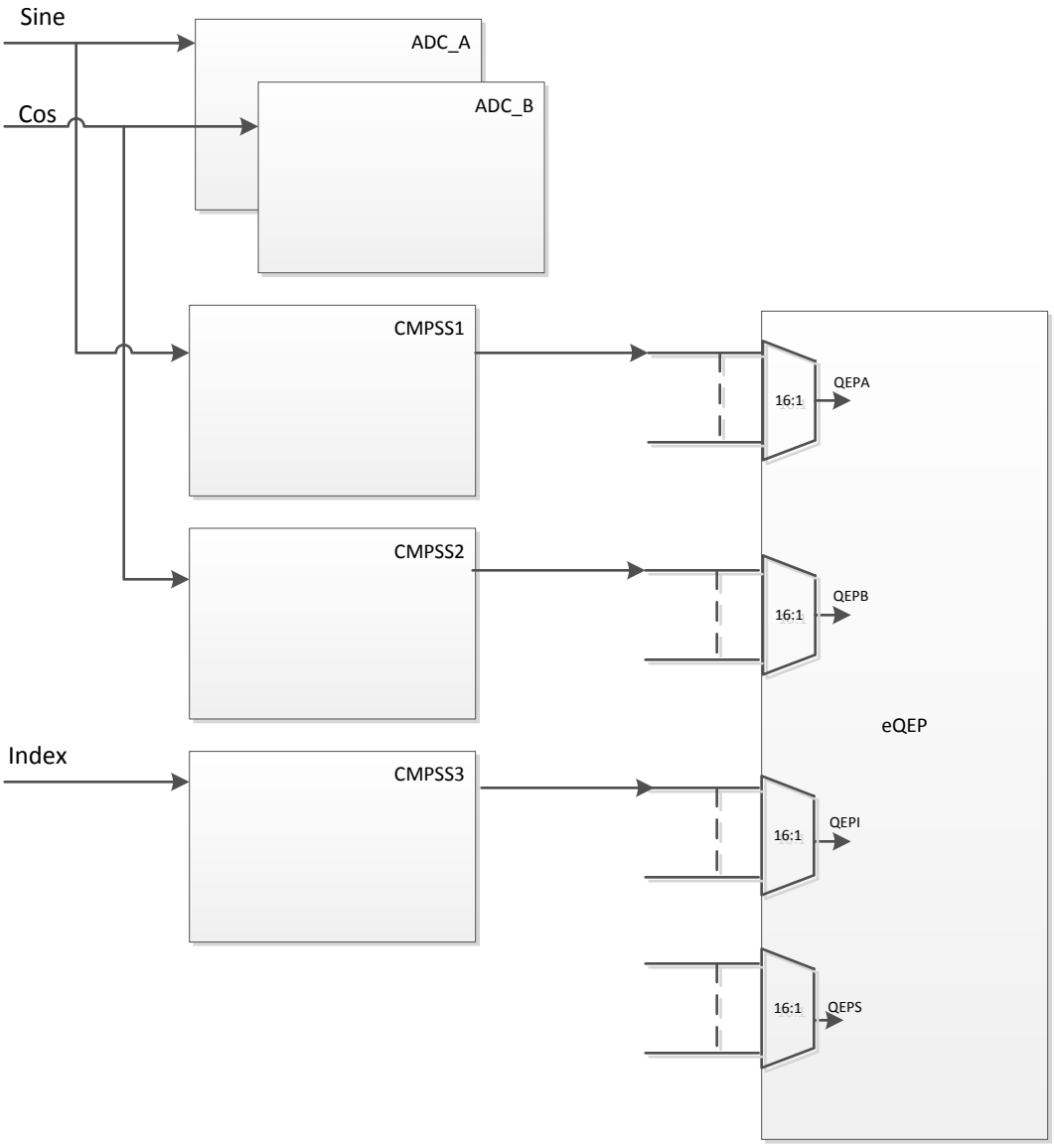


Figure 27-4. Using eQEP to Decode Signals from SinCos Transducer

Table 27-1. eQEP Input Source Select Table

QEPASEL, QEPBSEL, QEPISEL	Input Signal
0	INPUTXBAR
1	CMPSS1_CTRIPH
2	CMPSS2_CTRIPH
3	CMPSS3_CTRIPH
4	CMPSS4_CTRIPH
5	CMPSS5_CTRIPH
6	CMPSS6_CTRIPH
7	CMPSS7_CTRIPH
8	CMPSS8_CTRIPH
9	EPWMXBAR1
10	EPWMXBAR2
11	EPWMXBAR3
12	EPWMXBAR4
13	EPWMXBAR5
14	EPWMXBAR6
15	EPWMXBAR7

Note

Configuration of QEPSRCSEL register to select the source of QEPA, QEPB, and QEPI signals can lead to unexpected transition on these signals, which can cause an undesirable outcome if eQEP is already running. You need to make sure that the eQEP is disabled before configuring the QEPSRCSEL register for input signals.

27.3.2 Functional Description

The eQEP peripheral contains the following major functional units (as shown in Figure 27-5):

- Programmable input qualification for each pin (part of the GPIO MUX)
- Quadrature decoder unit (QDU)
- Position counter and control unit for position measurement (PCCU)
- Quadrature edge-capture unit for low-speed measurement (QCAP)
- Unit time base for speed/frequency measurement (UTIME)
- Watchdog timer for detecting stalls (QWDOG)
- Quadrature Mode Adapter (QMA)

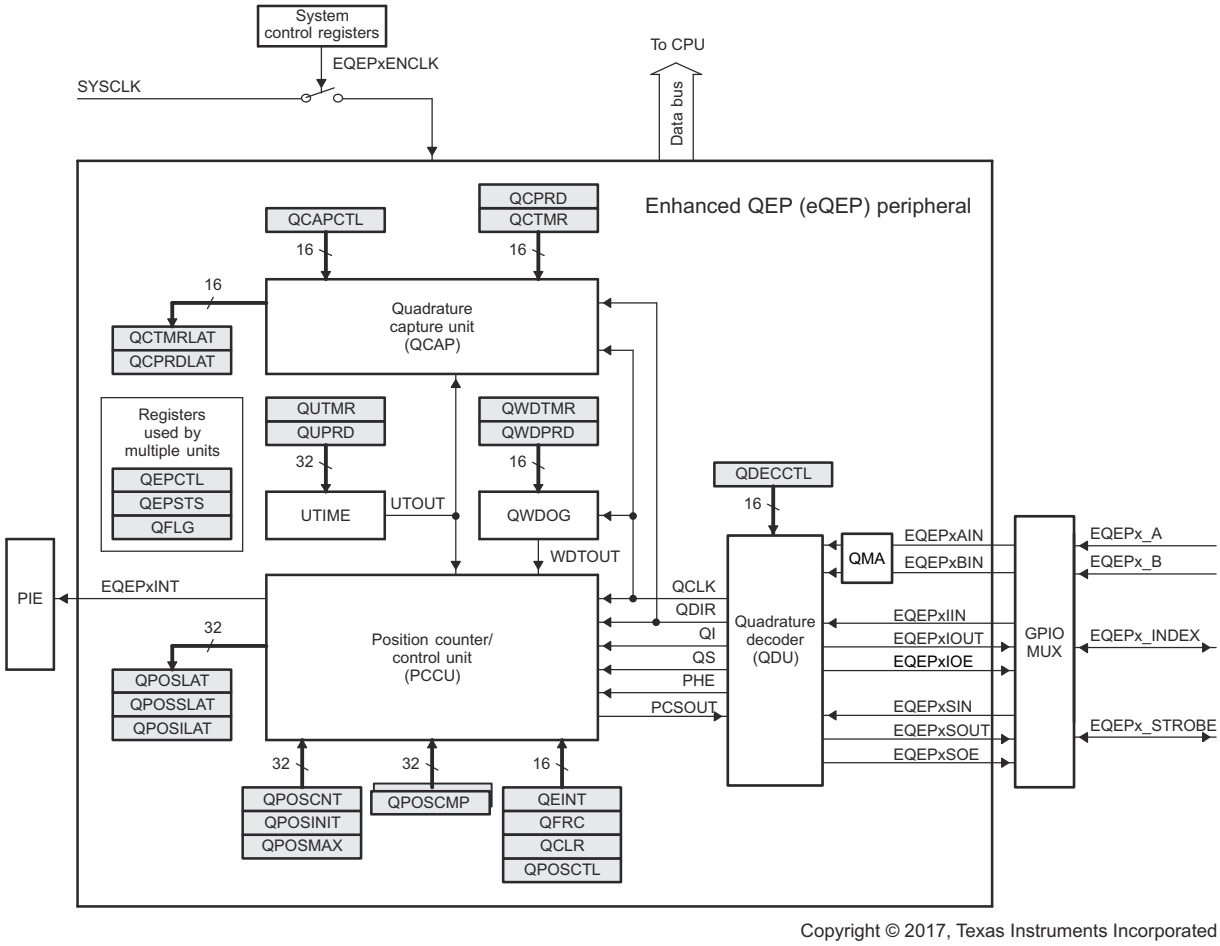


Figure 27-5. Functional Block Diagram of the eQEP Peripheral

27.3.3 eQEP Memory Map

Table 27-2 lists the registers with their memory locations, sizes, and reset values.

Table 27-2. EQEP Memory Map

Name	Offset	Size(x16)/ #shadow	Reset	Register Description
QPOSCNT	0x00	2/0	0x00000000	eQEP Position Counter
QPOSINIT	0x02	2/0	0x00000000	eQEP Initialization Position Count
QPOSMAX	0x04	2/0	0x00000000	eQEP Maximum Position Count
QPOSCMP	0x06	2/1	0x00000000	eQEP Position-compare
QPOSILAT	0x08	2/0	0x00000000	eQEP Index Position Latch
QPOSSLAT	0x0A	2/0	0x00000000	eQEP Strobe Position Latch
QPOSLAT	0x0C	2/0	0x00000000	eQEP Position Latch
QUTMR	0x0E	2/0	0x00000000	QEP Unit Timer
QUPRD	0x10	2/0	0x00000000	eQEP Unit Period Register
QWDTMR	0x12	1/0	0x0000	eQEP Watchdog Timer
QWDPRD	0x13	1/0	0x0000	eQEP Watchdog Period Register
QDECCTL	0x14	1/0	0x0000	eQEP Decoder Control Register
QEPCTL	0x15	1/0	0x0000	eQEP Control Register
QCAPCTL	0x16	1/0	0x0000	eQEP Capture Control Register
QPOSCTL	0x17	1/0	0x00000	eQEP Position-compare Control Register
QEINT	0x18	1/0	0x0000	eQEP Interrupt Enable Register
QFLG	0x19	1/0	0x0000	eQEP Interrupt Flag Register
QCLR	0x1A	1/0	0x0000	eQEP Interrupt Clear Register
QFRC	0x1B	1/0	0x0000	eQEP Interrupt Force Register
QEPSTS	0x1C	1/0	0x0000	eQEP Status Register
QCTMR	0x1D	1/0	0x0000	eQEP Capture Timer
QCPRD	0x1E	1/0	0x0000	eQEP Capture Period Register
QCTMRLAT	0x1F	1/0	0x0000	eQEP Capture Timer Latch
QCPRDLAT	0x20	1/0	0x0000	eQEP Capture Period Latch
reserved	0x21 to 0x2F	15/0		
REV	0x30	2/0	0x0000	eQEP Revision Number
QEPSTROBESEL	0x32	2/0	0x0000	eQEP Strobe select register
QMACTRL	0x34	2/0	0x0000	eQEP QMA Control register
QEPSRCSEL	0x36	2/0	0x0000	eQEP Source Select Register
reserved	0x38 to 0x3F	8/0		

27.4 Quadrature Decoder Unit (QDU)

Figure 27-6 shows a functional block diagram of the QDU.

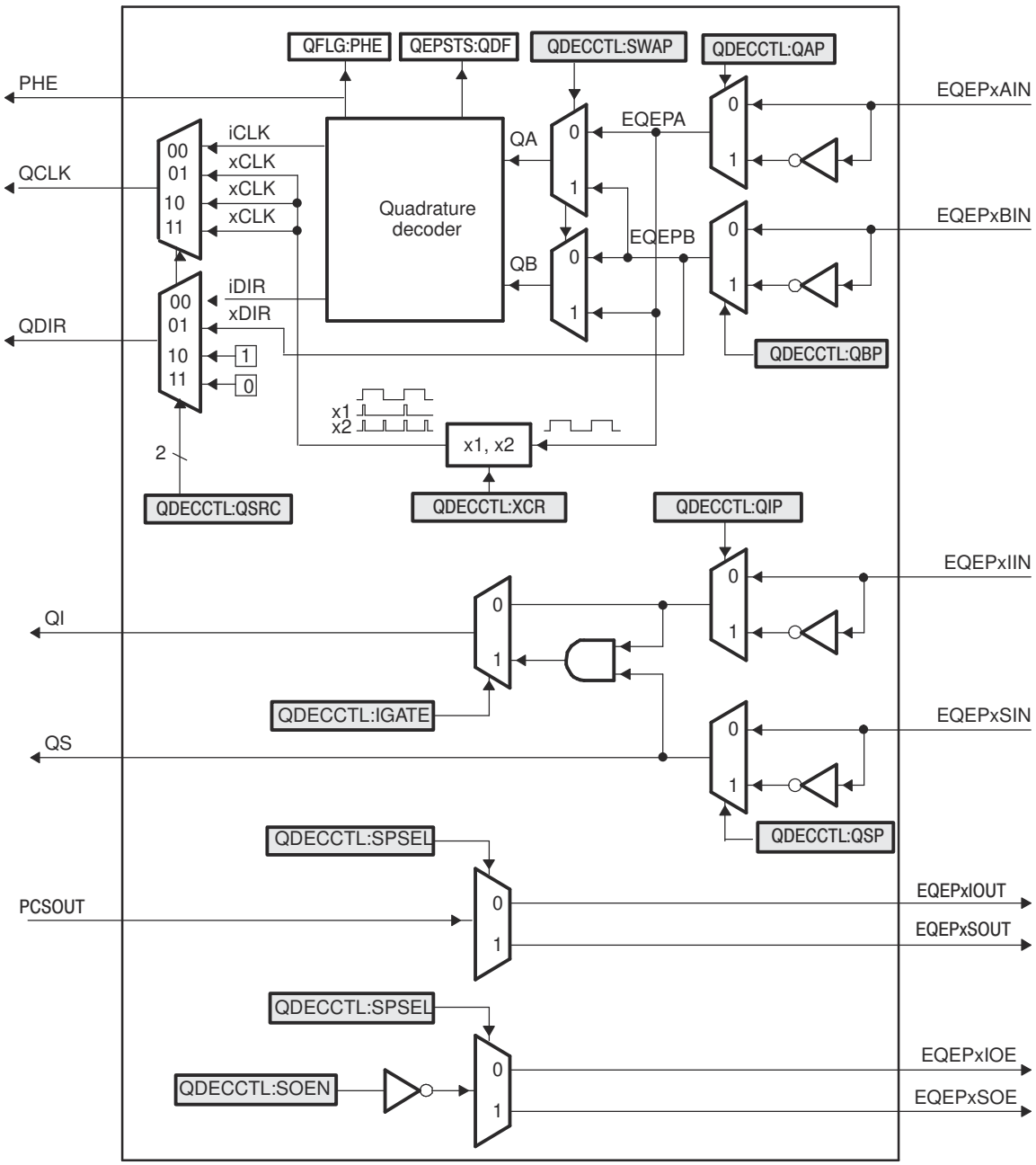


Figure 27-6. Functional Block Diagram of Decoder Unit

27.4.1 Position Counter Input Modes

Clock and direction input to the position counter is selected using QDECCTL[QSRC] bits, based on interface input requirement as follows:

- Quadrature-count mode
- Direction-count mode
- UP-count mode
- DOWN-count mode

27.4.1.1 Quadrature Count Mode

The quadrature decoder generates the direction and clock to the position counter in quadrature count mode.

Direction Decoding

The direction decoding logic of the eQEP circuit determines which one of the sequences (QEPA, QEPB) is the leading sequence and accordingly updates the direction information in the QEPSTS[QDF] bit. Table 27-3 and Figure 27-7 show the direction decoding logic in truth table and state machine form. Both edges of the QEPA and QEPB signals are sensed to generate count pulses for the position counter. Therefore, the frequency of the clock generated by the eQEP logic is four times that of each input sequence. Figure 27-8 shows the direction decoding and clock generation from the eQEP input signals.

Table 27-3. Quadrature Decoder Truth Table

Previous Edge	Present Edge	QDIR	QPOSCNT
QA↑	QB↑	UP	Increment
	QB↓	DOWN	Decrement
	QA↓	TOGGLE	Increment or Decrement
QA↓	QB↓	UP	Increment
	QB↑	DOWN	Decrement
	QA↑	TOGGLE	Increment or Decrement
QB↑	QA↑	DOWN	Decrement
	QA↓	UP	Increment
	QB↓	TOGGLE	Increment or Decrement
QB↓	QA↓	DOWN	Decrement
	QA↑	UP	Increment
	QB↑	TOGGLE	Increment or Decrement

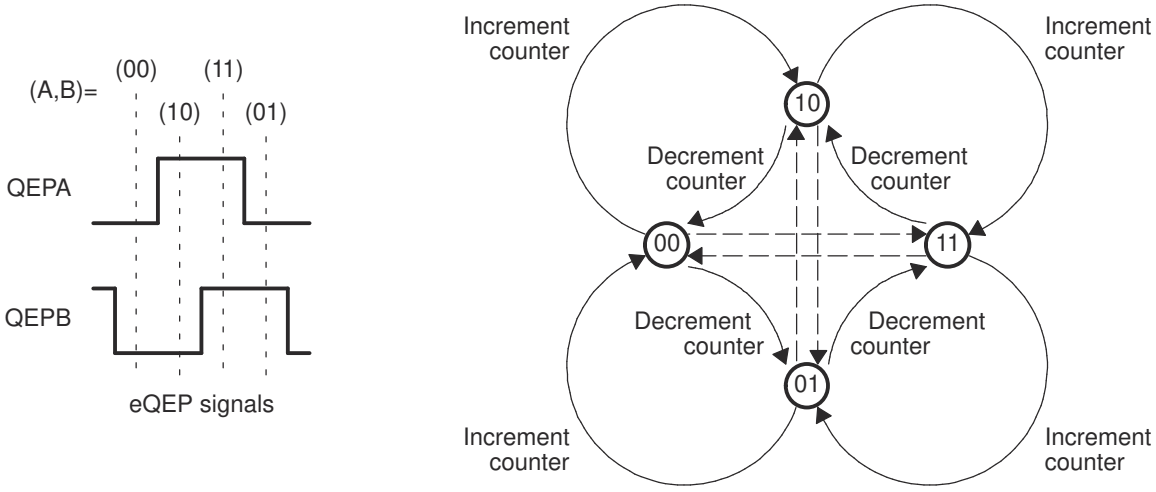


Figure 27-7. Quadrature Decoder State Machine

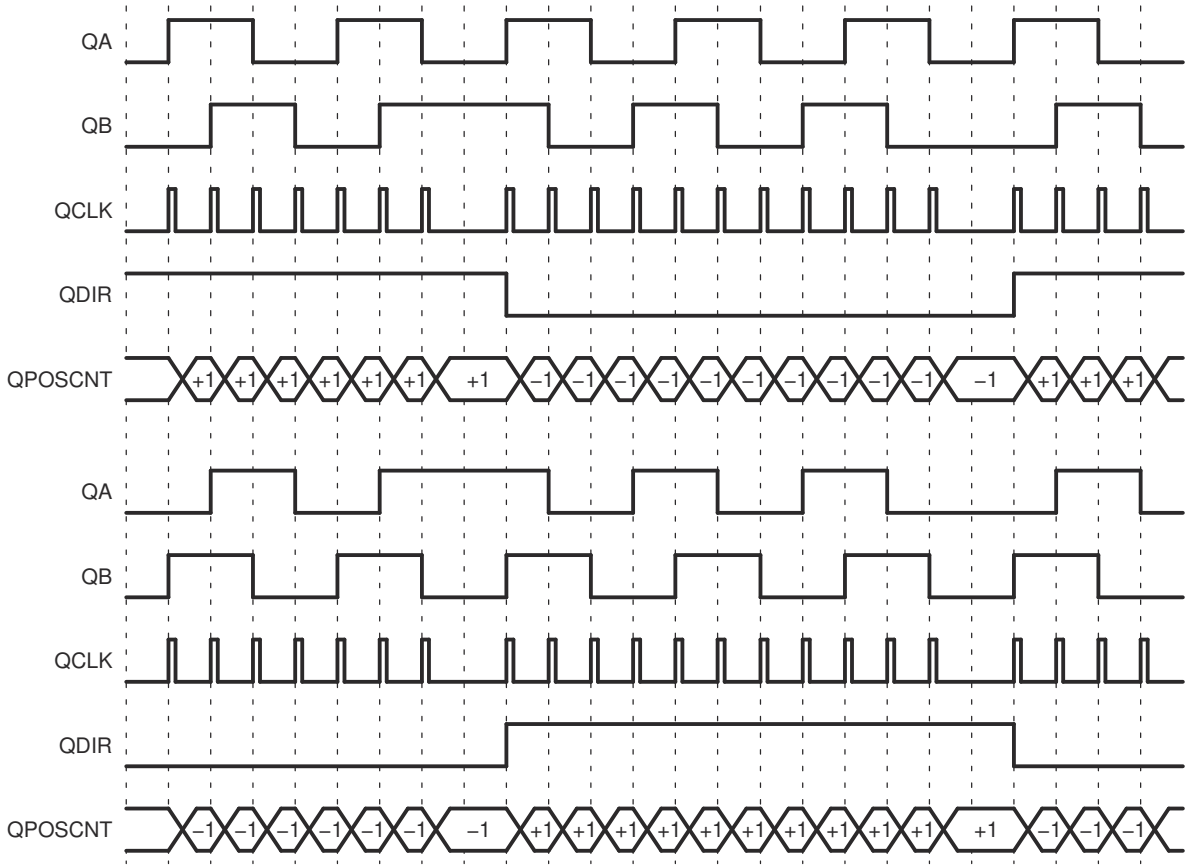


Figure 27-8. Quadrature-clock and Direction Decoding

Phase Error Flag In normal operating conditions, quadrature inputs QEPA and QEPB is 90 degrees out of phase. The phase error flag (PHE) is set in the QFLG register and the QPOSCNT value can be incorrect and offset by multiples of 1 or 3. That is, when edge transition is detected simultaneously on the QEPA and QEPB signals to optionally generate interrupts. State transitions marked by dashed lines in Figure 27-7 are invalid transitions that generate a phase error.

Count Multiplication The eQEP position counter provides 4x times the resolution of an input clock by generating a quadrature-clock (QCLK) on the rising/falling edges of both eQEP input clocks (QEPA and QEPB) as shown in Figure 27-8.

Reverse Count In normal quadrature count operation, QEPA input is applied to the QA input of the quadrature decoder and the QEPB input is applied to the QB input of the quadrature decoder. Reverse counting is enabled by setting the SWAP bit in the QDECCTL register. This swaps the input to the quadrature decoder; thereby, reversing the counting direction.

27.4.1.2 Direction-Count Mode

Some position encoders provide direction and clock outputs, instead of quadrature outputs. In such cases, direction-count mode can be used. The QEPA input provides the clock for the position counter and the QEPB input has the direction information. The position counter is incremented on every rising edge of a QEPA input when the direction input is high, and decremented when the direction input is low.

27.4.1.3 Up-Count Mode

The counter direction signal is hard-wired for up-count and the position counter is used to measure the frequency of the QEPA input. Clearing the QDECCTL[XCR] bit enables clock generation to the position counter on both edges of the QEPA input; thereby, increasing the measurement resolution by a factor of 2x. In up-count mode, we recommend that the application not configure QEPB as a GPIO mux option, or make sure that a signal edge is not generated on the QEPB input.

27.4.1.4 Down-Count Mode

The counter direction signal is hardwired for a down-count and the position counter is used to measure the frequency of the QEPA input. Clearing the QDECCTL[XCR] bit enables clock generation to the position counter on both edges of a QEPA input, thereby increasing the measurement resolution by a factor of 2x. In down-count mode, it is recommended that the application not configure QEPB as a GPIO mux option, or ensure that a signal edge is not generated on the QEPB input.

27.4.2 eQEP Input Polarity Selection

Each eQEP input can be inverted using QDECCTL[8:5] control bits. As an example, setting the QDECCTL[QIP] bit inverts the index input.

27.4.3 Position-Compare Sync Output

The enhanced eQEP peripheral includes a position-compare unit that is used to generate the position-compare sync signal on compare match between the position-counter register (QPOSCNT) and the position- compare register (QPOSCMP). This sync signal can be output using an index pin or strobe pin of the EQEP peripheral.

Setting the QDECCTL[SOEN] bit enables the position-compare sync output and the QDECCTL[SPSEL] bit selects either an eQEP index pin or an eQEP strobe pin.

27.5 Position Counter and Control Unit (PCCU)

The position-counter and control unit provides two configuration registers (QEPCTL and QPOSCTL) for setting up position-counter operational modes, position-counter initialization/latch modes and position-compare logic for sync signal generation.

27.5.1 Position Counter Operating Modes

Position-counter data can be captured in different manners. In some systems, the position counter is accumulated continuously for multiple revolutions and the position-counter value provides the position information with respect to the known reference. An example of this is the quadrature encoder mounted on the motor controlling the print head in the printer. Here the position counter is reset by moving the print head to the home position and then the position counter provides absolute position information with respect to home position.

In other systems, the position counter is reset on every revolution using index pulse, and the position counter provides a rotor angle with respect to the index pulse position.

The position counter can be configured to operate in following four modes

- Position-Counter Reset on Index Event
- Position-Counter Reset on Maximum Position
- Position-Counter Reset on the first Index Event
- Position-Counter Reset on Unit Time Out Event (Frequency Measurement)

In all the above operating modes, the position counter is reset to 0 on overflow and to the QPOSMAX register value on underflow. Overflow occurs when the position counter counts up after the QPOSMAX value. Underflow occurs when the position counter counts down after 0. The Interrupt flag is set to indicate overflow/underflow in QFLG register.

27.5.1.1 Position Counter Reset on Index Event (QEPCTL[PCRM]=00)

If the index event occurs during the forward movement, then the position counter is reset to 0 on the next eQEP clock. If the index event occurs during the reverse movement, then the position counter is reset to the value in the QPOSMAX register on the next eQEP clock.

The first index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (QEPSTS[FIMF]) and direction on the first index event marker (QEPSTS[FIDF]) in QEPSTS registers, the eQEP peripheral also remembers the quadrature edge on the first index marker so that same relative quadrature transition is used for index event reset operation.

For example, if the first reset operation occurs on the falling edge of QEPB during the forward direction, then all the subsequent reset must be aligned with the falling edge of QEPB for the forward rotation and on the rising edge of QEPB for the reverse rotation as shown in Figure 27-9.

The position-counter value is latched to the QPOSILAT register and direction information is recorded in the QEPSTS[QDLF] bit on every index event marker. The position-counter error flag (QEPSTS[PCEF]) and error interrupt flag (QFLG[PCE]) are set if the latched value is not equal to 0 or QPOSMAX. The position-counter error flag (QEPSTS[PCEF]) is updated on every index event marker and an interrupt flag (QFLG[PCE]) is set on error that can be cleared only through software.

The index event latch configuration QEPCTL[IEL] must be configured to 00 or 11 when pcrm=0 and the position counter error flag/interrupt flag are generated only in index event reset mode. The position counter value is latched into the IPOSLAT register on every index marker.

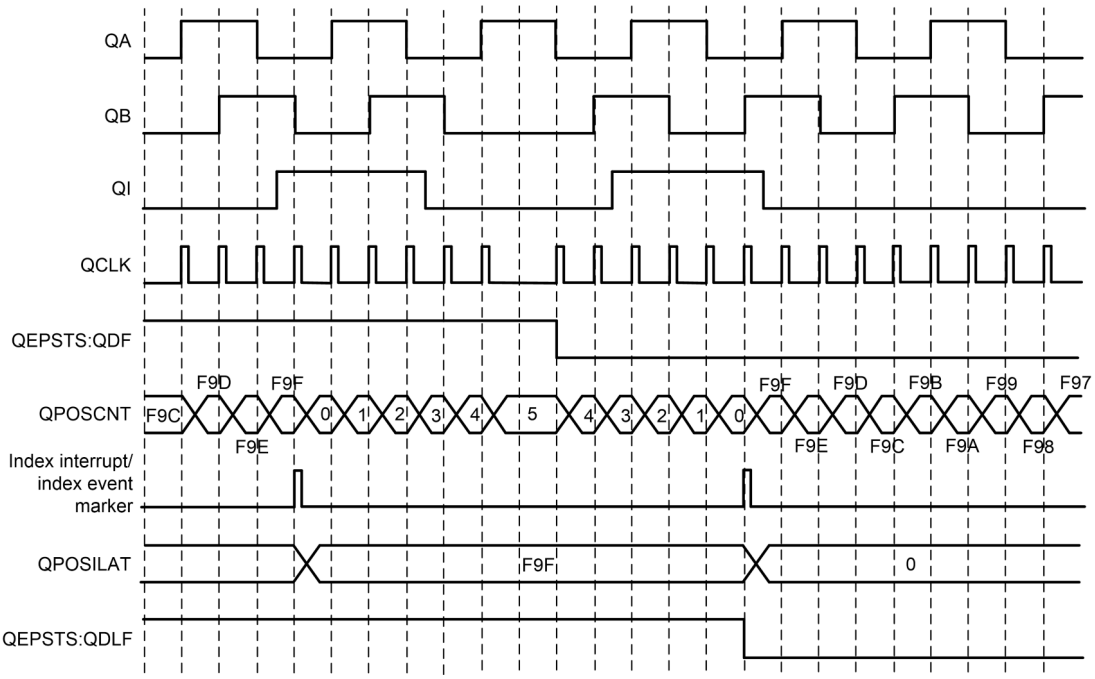


Figure 27-9. Position Counter Reset by Index Pulse for 1000-Line Encoder (QPOSMAX = 3999 or 0xF9F)

Note

In case of a boundary condition where the time period between the Index Event and the previous QCLK edge is less than SYSCLK period, then QPOSCNT gets reset to zero or QPOSMAX in the same SYSCLK cycle and does not wait for the next QCLK edge to occur.

27.5.1.2 Position Counter Reset on Maximum Position (QEPCTL[PCRM]=01)

If the position counter is equal to QPOSMAX, then the position counter is reset to 0 on the next eQEP clock for forward movement and position counter overflow flag is set. If the position counter is equal to ZERO, then the position counter is reset to QPOSMAX on the next QEP clock for reverse movement and position-counter underflow flag is set. Figure 27-10 shows the position-counter reset operation in this mode.

The first index marker fields (QEPSTS[FIDF] and QEPSTS[FIMF]) are not applicable in this mode.

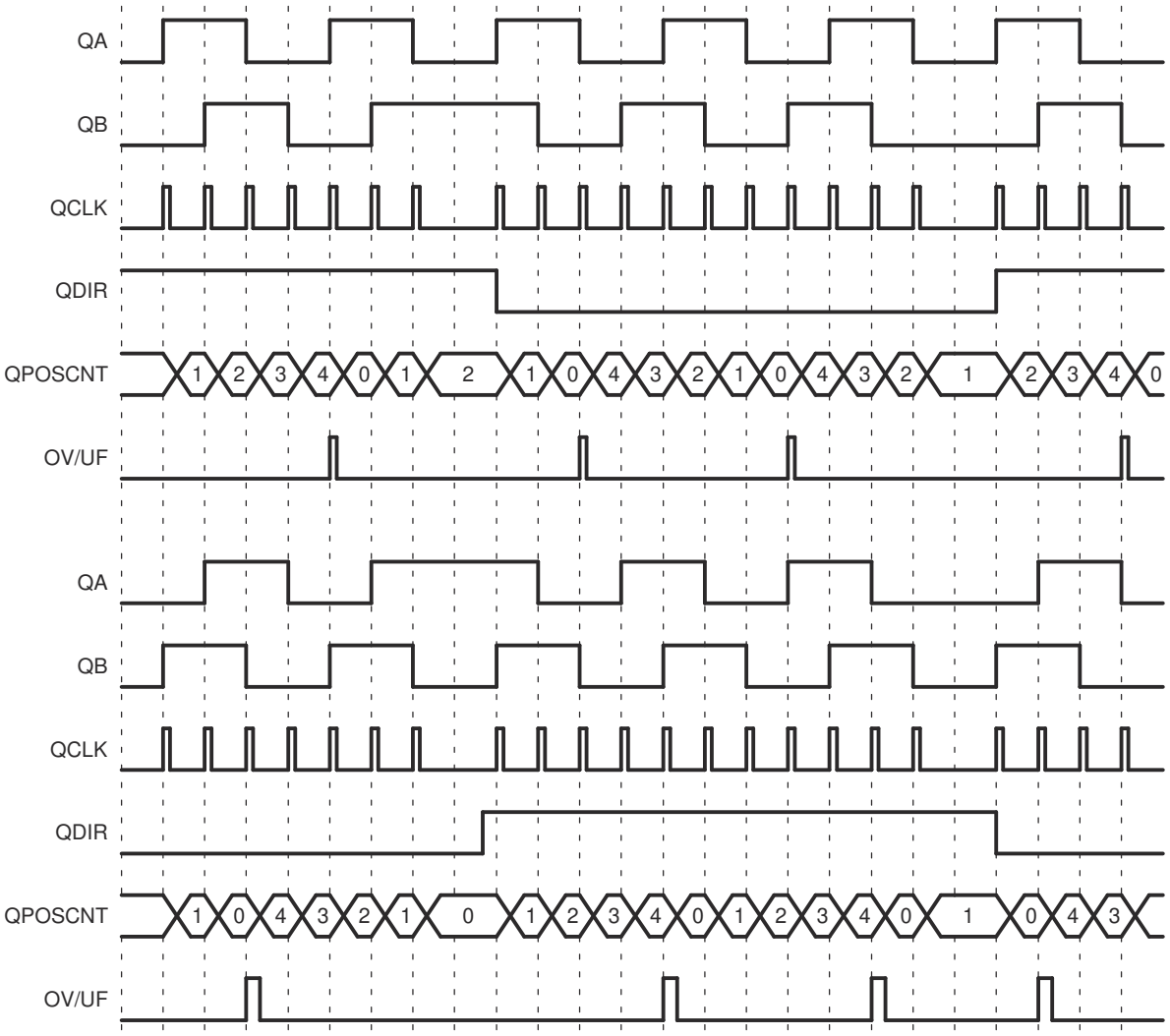


Figure 27-10. Position Counter Underflow/Overflow (QPOSMAX = 4)

27.5.1.3 Position Counter Reset on the First Index Event (QEPCTL[PCRM] = 10)

If the index event occurs during forward movement, then the position counter is reset to 0 on the next eQEP clock. If the index event occurs during the reverse movement, then the position counter is reset to the value in the QPOSMAX register on the next eQEP clock. Note that this is done only on the first occurrence and subsequently the position-counter value is not reset on an index event; rather, the position-counter value is reset based on the maximum position as described in Section 27.5.1.2.

The first index marker fields (QEPSTS[FIDF] and QEPSTS[FIMF]) are not applicable in this mode.

27.5.1.4 Position Counter Reset on Unit Time-out Event (QEPCTL[PCRM] = 11)

In this mode, QPOSCNT is set to 0 or QPOMAX, depending on the direction mode selected by QDECCTL[QSRC] bits on a unit time event. This is useful for frequency measurement.

27.5.2 Position Counter Latch

The eQEP index and strobe input can be configured to latch the position counter (QPOSCNT) into QPOSILAT and QPOSSLAT, respectively, on occurrence of a definite event on these pins.

27.5.2.1 Index Event Latch

In some applications, it is not desirable to reset the position counter on every index event and instead it can be required to operate the position counter in full 32-bit mode (QEPCTL[PCRM] = 01 and QEPCTL[PCRM] = 10 modes).

In such cases, the eQEP position counter can be configured to latch on the following events and direction information is recorded in the QEPSTS[QDLF] bit on every index event marker.

- Latch on Rising edge (QEPCTL[IEL]=01)
- Latch on Falling edge (QEPCTL[IEL]=10)
- Latch on Index Event Marker (QEPCTL[IEL]=11)

This is particularly useful as an error checking mechanism to check if the position counter accumulated the correct number of counts between index events. As an example, the 1000-line encoder must count 4000 times when moving in the same direction between the index events.

The index event latch interrupt flag (QFLG[IEL]) is set when the position counter is latched to the QPOSILAT register. The index event latch configuration bits (QEPCTL[IEL]) are ignored when QEPCTL[PCRM] = 00.

Latch on Rising Edge (QEPCTL[IEL]=01)	The position-counter value (QPOSCNT) is latched to the QPOSILAT register on every rising edge of an index input.
Latch on Falling Edge (QEPCTL[IEL] = 10)	The position-counter value (QPOSCNT) is latched to the QPOSILAT register on every falling edge of index input.
Latch on Index Event Marker/Software Index Marker (QEPCTL[IEL] = 11	The first index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (QEPSTS[FIMF]) and the direction on the first index event marker (QEPSTS[FIDF]) in the QEPSTS registers. The eQEP peripheral also remembers the quadrature edge on the first index marker so that the same relative quadrature transition is used for latching the position counter (QEPCTL[IEL]=11).

Figure 27-11 shows the position counter latch using an index event marker.

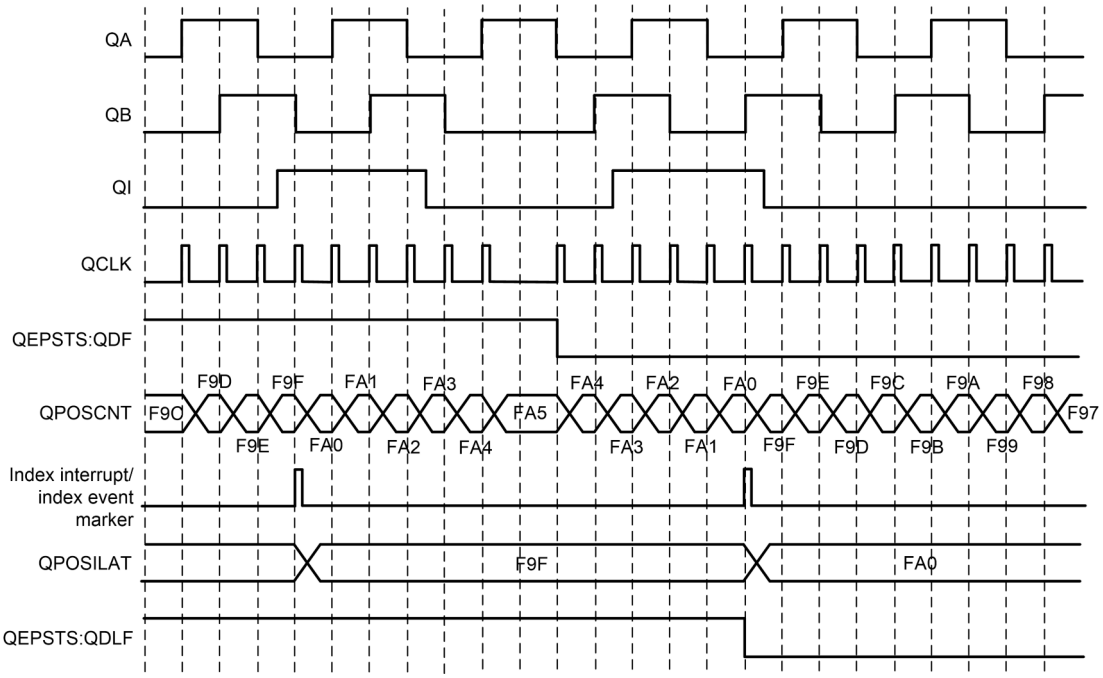


Figure 27-11. Software Index Marker for 1000-line Encoder (QEPCTL[IEL] = 1)

27.5.2.2 Strobe Event Latch

The position-counter value is latched to the QPOSSLAT register on the rising edge of the strobe input by clearing the QEPCTL[SEL] bit.

If the QEPCTL[SEL] bit is set, then the position-counter value is latched to the QPOSSLAT register on the rising edge of the strobe input for forward direction, and on the falling edge of the strobe input for reverse direction as shown in Figure 27-12.

The strobe event latch interrupt flag (QFLG[SEL) is set when the position counter is latched to the QPOSSLAT register.

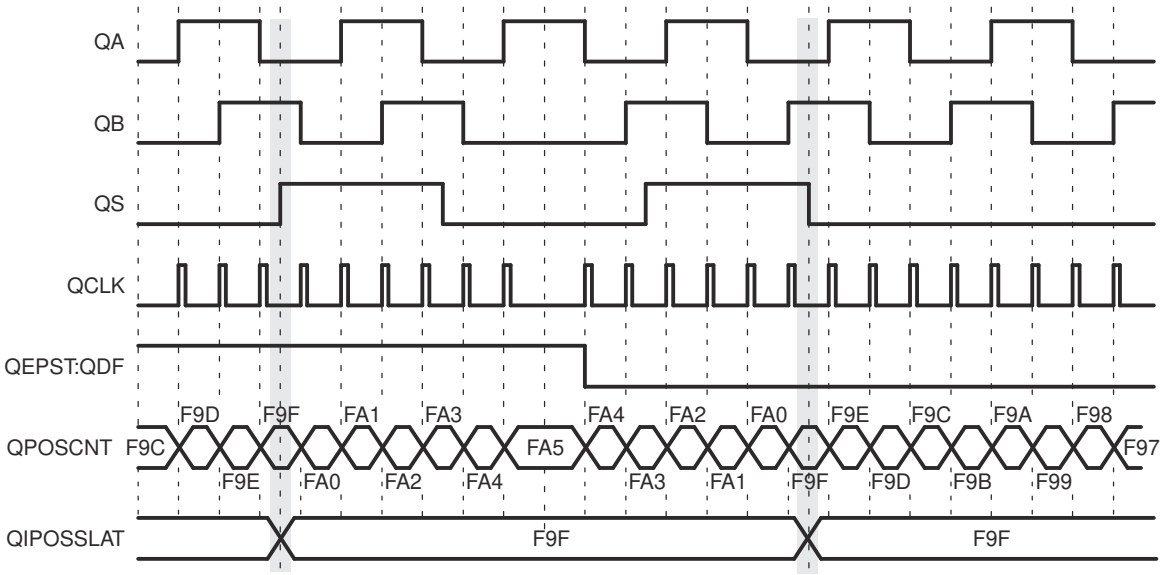


Figure 27-12. Strobe Event Latch (QEPCTL[SEL] = 1)

There is an added feature on Type 2.0 eQEP where position-counter value can also be latched on ADCSOCA and ADCSOSCB events by configuring the register QEPSTROBESEL.STROBESEL as shown in figure Figure 27-13

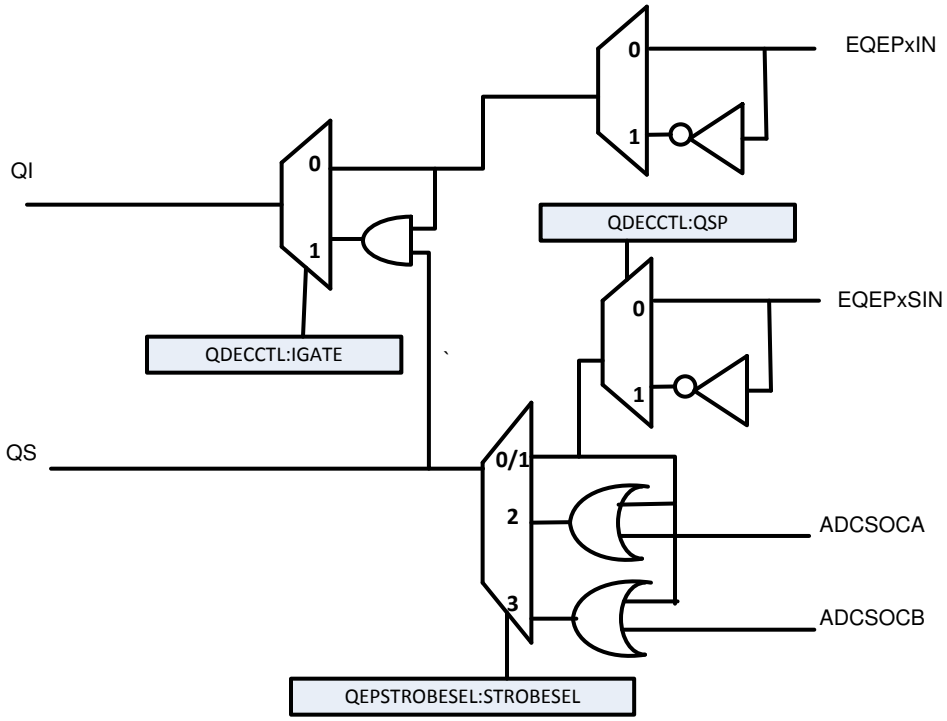


Figure 27-13. Latching Position Counter on ADCSOCA/ADCSOCB Event

27.5.3 Position Counter Initialization

The position counter can be initialized using the following events:

- Index event
- Strobe event
- Software initialization

Index Event Initialization (IEI)	The QEPI index input can be used to trigger the initialization of the position counter at the rising or falling edge of the index input. If the QEPCTL[IEI] bits are 10, then the position counter (QPOSCNT) is initialized with a value in the QPOSINIT register on the rising edge of index input. Conversely, if the QEPCTL[IEI] bits are 11, initialization is on the falling edge of the index input.
Strobe Event Initialization (SEI)	If the QEPCTL[SEI] bits are 10, then the position counter is initialized with a value in the QPOSINIT register on the rising edge of strobe input. If QEPCTL[SEL] bits are 11, then the position counter is initialized with a value in the QPOSINIT register on the rising edge of strobe input for forward direction and on the falling edge of strobe input for reverse direction.
Software Initialization (SWI)	The position counter can be initialized in software by writing a 1 to the QEPCTL[SWI] bit. This bit is not automatically cleared. While the bit is still set, if a 1 is written to the bit again, the position counter is re-initialized.

27.5.4 eQEP Position-compare Unit

The eQEP peripheral includes a position-compare unit that is used to generate a sync output and interrupt on a position-compare match. Figure 27-14 shows a diagram. The position-compare (QPOSCMP) register is shadowed and shadow mode can be enabled or disabled using the QPOSCTL[PSSHDW] bit. If the shadow mode is not enabled, the CPU writes directly to the active position compare register.

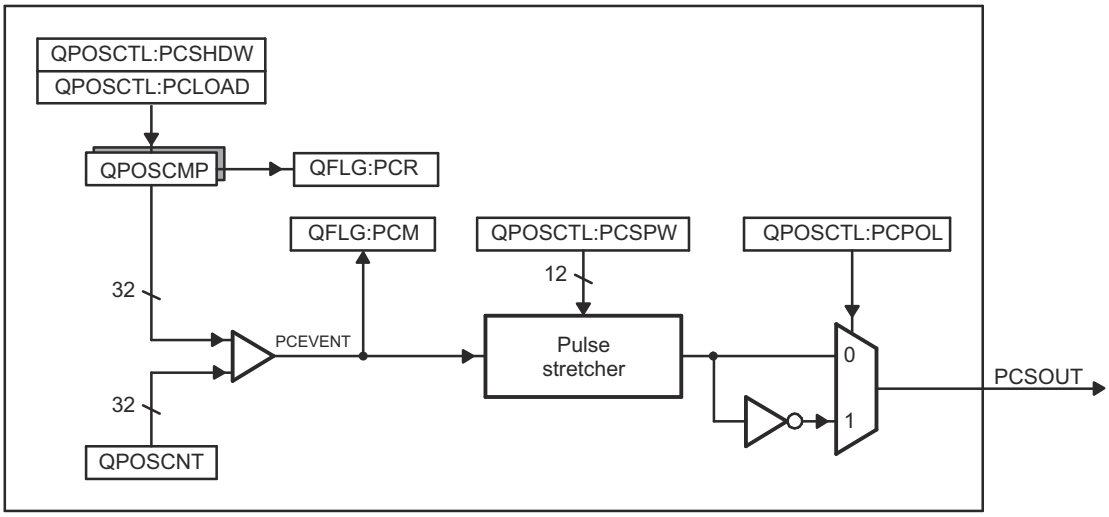


Figure 27-14. eQEP Position-compare Unit

In shadow mode, you can configure the position-compare unit (QPOSCTL[PCLOAD]) to load the shadow register value into the active register on the following events, and to generate the position-compare ready (QFLG[PCR]) interrupt after loading.

- Load on compare match
- Load on position-counter zero event

The position-compare match (QFLG[PCM]) is set when the position-counter value (QPOSCNT) matches with the active position-compare register (QPOSCMP) and the position-compare sync output of the programmable pulse width is generated on compare-match to trigger an external device.

For example, if QPOSCMP = 2, the position-compare unit generates a position-compare event on 1 to 2 transitions of the eQEP position counter for forward counting direction and on 3 to 2 transitions of the eQEP position counter for reverse counting direction (see Figure 27-15).

See the register section for the layout of the eQEP Position-Compare Control Register (QPOSCTL) and description of the QPOSCTL bit fields.

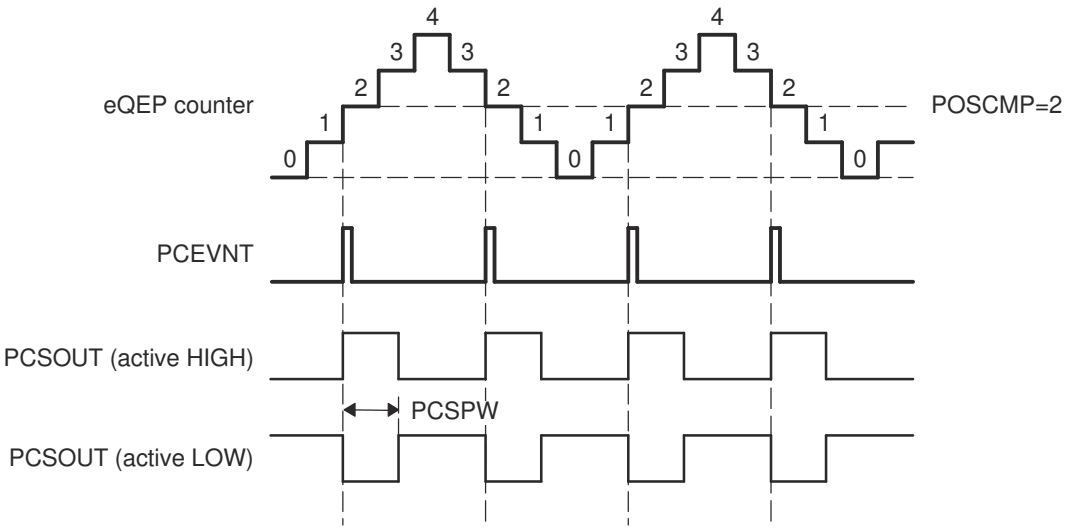


Figure 27-15. eQEP Position-compare Event Generation Points

The pulse stretcher logic in the position-compare unit generates a programmable position-compare sync pulse output on the position-compare match. In the event of a new position-compare match while a previous position-compare pulse is still active, then the pulse stretcher generates a pulse of specified duration from the new position-compare event as shown in Figure 27-16.

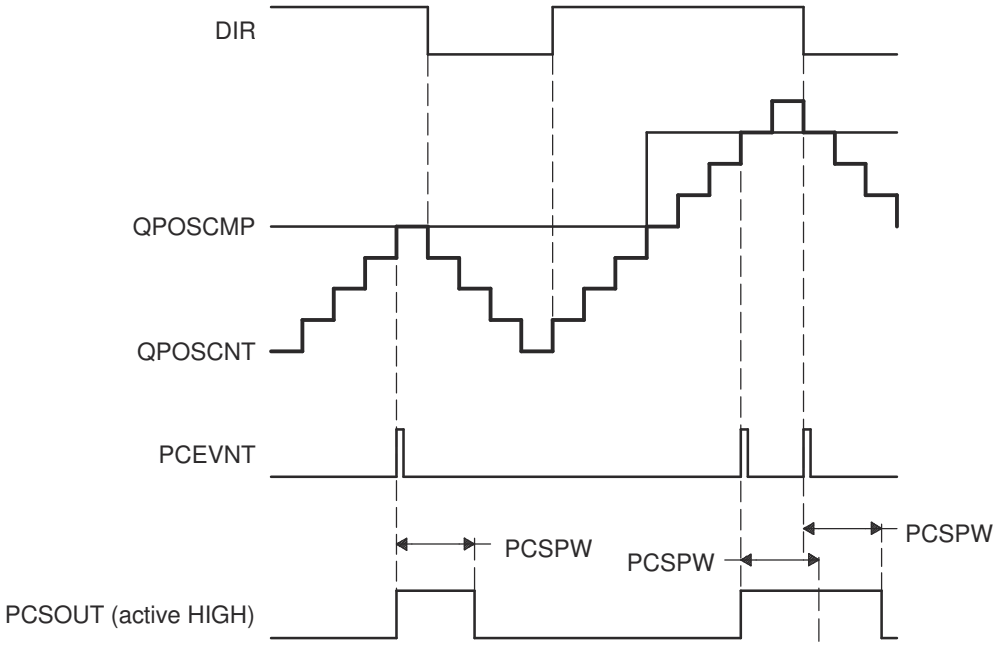


Figure 27-16. eQEP Position-compare Sync Output Pulse Stretcher

27.6 eQEP Edge Capture Unit

The eQEP peripheral includes an integrated edge capture unit to measure the elapsed time between the unit position events as shown in Figure 27-17. This feature is typically used for low-speed measurement using the following formula:

$$v(k) = \frac{X}{t(k) - t(k-1)} = \frac{X}{\Delta T} \quad (13)$$

where:

- X = Unit position is defined by integer multiple of quadrature edges (see Figure 27-18)
- ΔT = Elapsed time between unit position events
- v(k) = Velocity at time instant "k"

The eQEP capture timer (QCTMR) runs from prescaled SYSCLKOUT and the prescaler is programmed by the QCAPCTL[CCPS] bits. The capture timer (QCTMR) value is latched into the capture period register (QCPRD) on every unit position event and then the capture timer is reset, a flag is set in QEPSTS:UPEVNT to indicate that new value is latched into the QCPRD register. Software can check this status flag before reading the period register for low speed measurement, and clear the flag by writing 1.

Time measurement (ΔT) between unit position events is correct if the following conditions are met:

- No more than 65,535 counts have occurred between unit position events.
- No direction change between unit position events.

If the QEP capture timer overflows between unit position events, then the timer sets the QEP capture overflow flag (QEPSTS[COEF]) in the status register and the QCPRDLAT register is set to 0xFFFF. If direction change occurs between the unit position events, then the error flag is set in the status register (QEPSTS[CDEF]) and the QCPRDLAT register is set to 0xFFFF.

The Capture Timer (QCTMR) and Capture Period register (QCPRD) can be configured to latch on the following events:

- CPU read of QPOSCNT register
- Unit time-out event

If the QEPCTL[QCLM] bit is cleared, then the capture timer and capture period values are latched into the QCTMRLAT and QCPRDLAT registers, respectively, when the CPU reads the position counter (QPOSCNT).

If the QEPCTL[QCLM] bit is set, then the position counter, capture timer, and capture period values are latched into the QPOSLAT, QCTMRLAT and QCPRDLAT registers, respectively, on unit time out.

Figure 27-19 shows the capture unit operation along with the position counter.

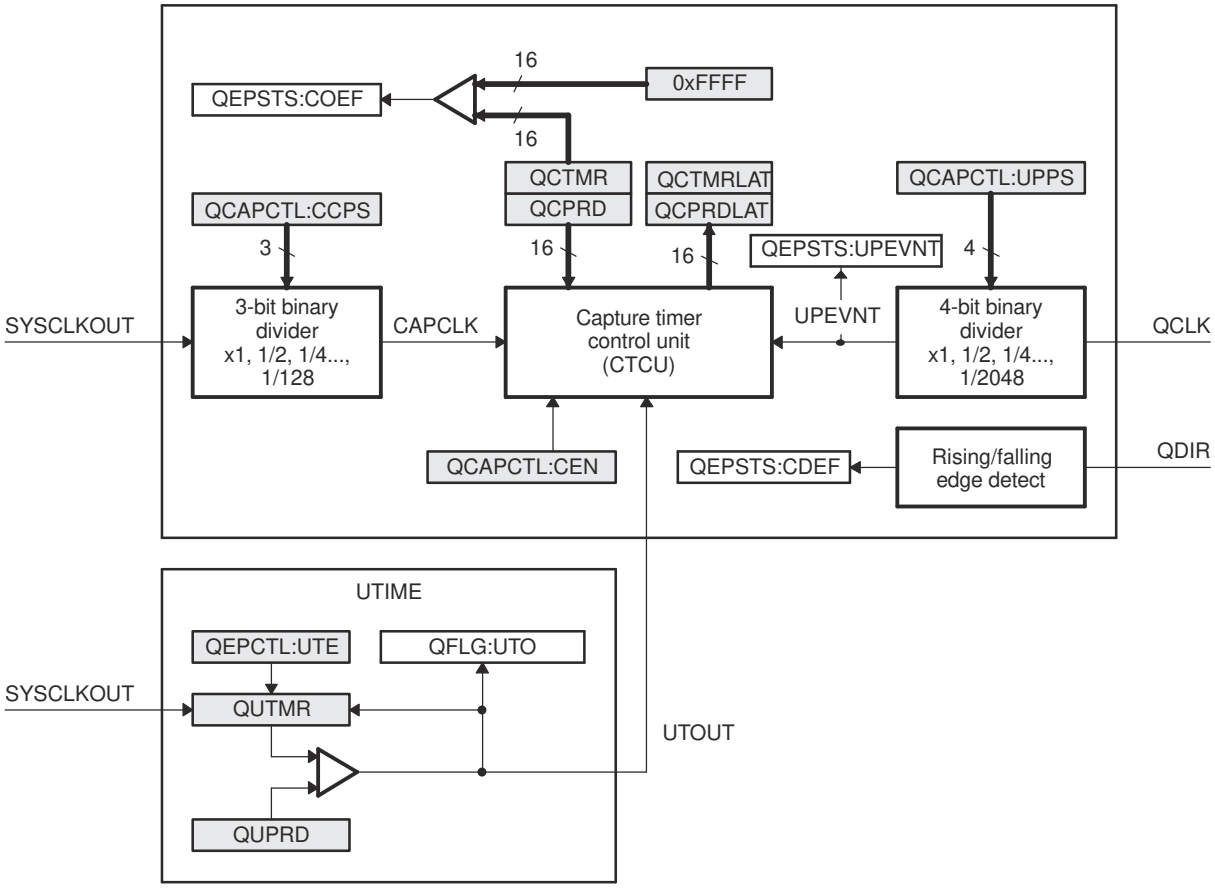
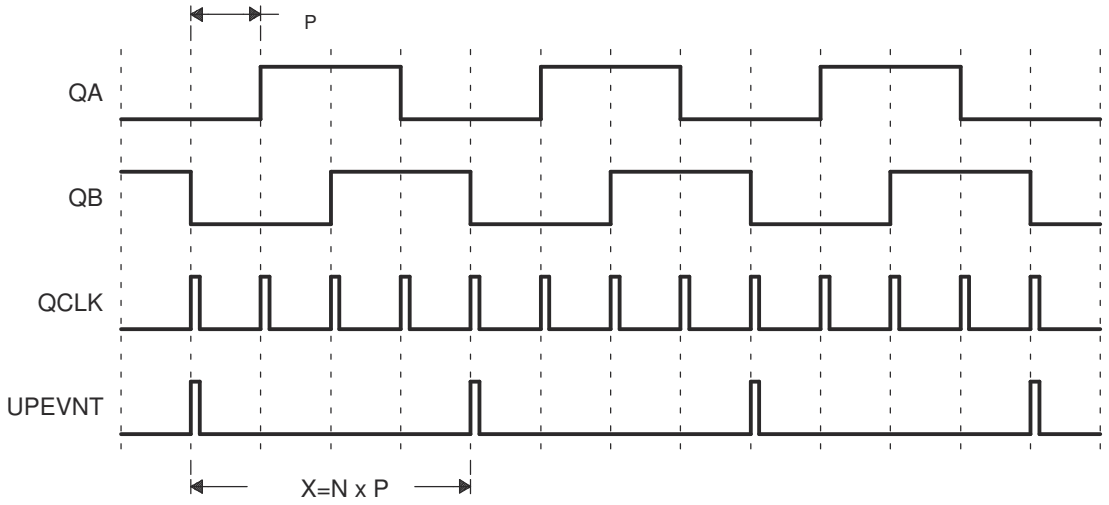


Figure 27-17. eQEP Edge Capture Unit

CAUTION

The QCAPCTL[UPPS] prescaler cannot be modified dynamically (such as switching the unit event prescaler from QCLK/4 to QCLK/8). Doing so can result in undefined behavior. The QCAPCTL[CPPS] prescaler can be modified dynamically (such as switching CAPCLK prescaling mode from SYSCLK/4 to SYSCLK/8) only after the capture unit is disabled.



N = Number of quadrature periods selected using QCAPCTL[UPPS] bits

Figure 27-18. Unit Position Event for Low Speed Measurement (QCAPCTL[UPPS] = 0010)

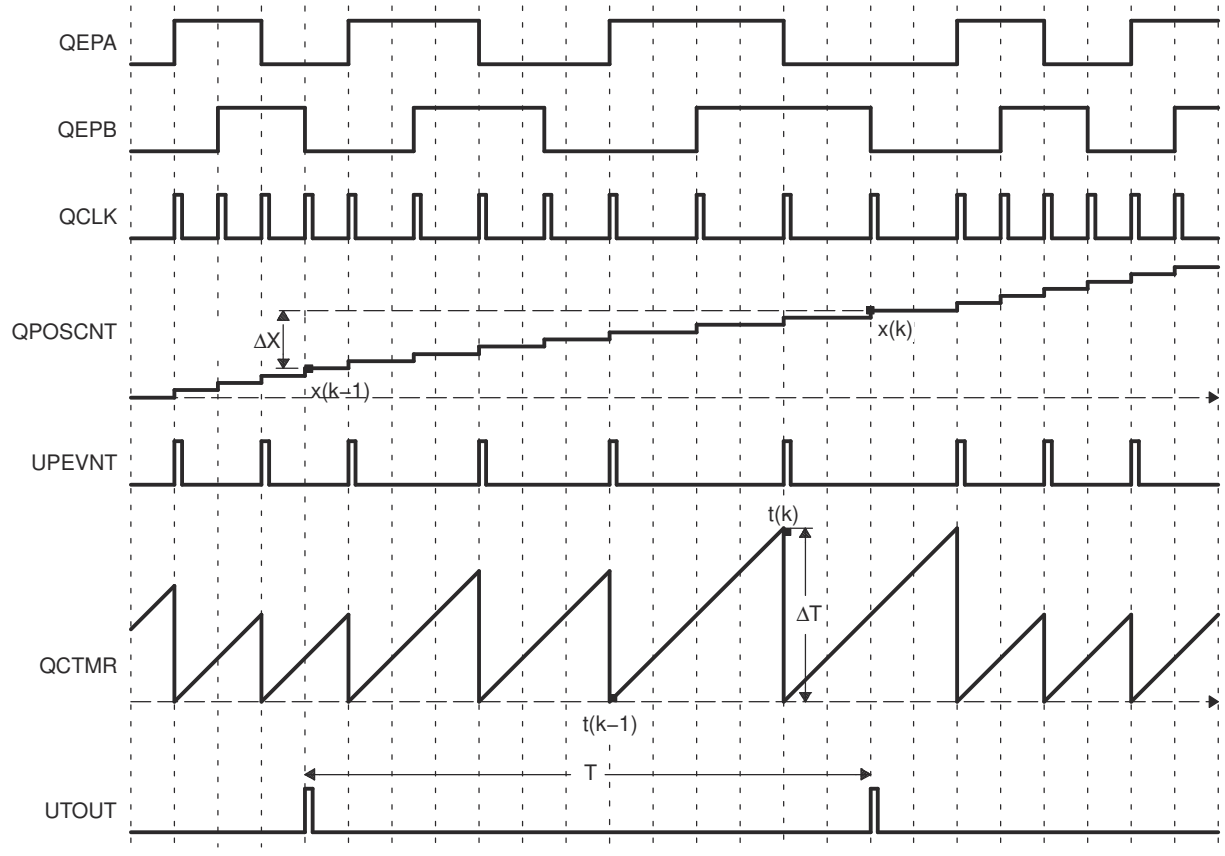


Figure 27-19. eQEP Edge Capture Unit - Timing Details

Velocity calculation equation:

$$v(k) = \frac{x(k) - x(k-1)}{T} = \frac{\Delta X}{T} \text{ o} \quad (14)$$

where:

- v(k) = Velocity at time instant k
- x(k) = Position at time instant k
- x(k-1) = Position at time instant k-1
- T = Fixed unit time or inverse of velocity calculation rate
- ΔX = Incremental position movement in unit time
- X = Fixed unit position
- ΔT = Incremental time elapsed for unit position movement
- t(k) = Time instant "k"
- t(k-1) = Time instant "k-1"

Unit time (T) and unit period (X) are configured using the QUPRD and QCAPCTL[UPPS] registers. Incremental position output and incremental time output is available in the QPOSLAT and QCPRDLAT registers.

Parameter	Relevant Register to Configure or Read the Information
T	Unit Period Register (QUPRD)
ΔX	Incremental Position = QPOSLAT(k) - QPOSLAT(K-1)
X	Fixed-unit position defined by sensor resolution and QCAPCTL[UPPS] bits
ΔT	Capture Period Latch (QCPRDLAT)

27.7 eQEP Watchdog

The eQEP peripheral contains a 16-bit watchdog timer (Figure 27-20) that monitors the quadrature clock to indicate proper operation of the motion-control system. The eQEP watchdog timer is clocked from SYSCLKOUT/64 and the quadrature clock event (pulse) resets the watchdog timer. If no quadrature clock event is detected until a period match (QWDPRD = QWDTMR), then the watchdog timer times out and the watchdog interrupt flag is set (QFLG[WTO]). The time-out value is programmable through the watchdog period register (QWDPRD).

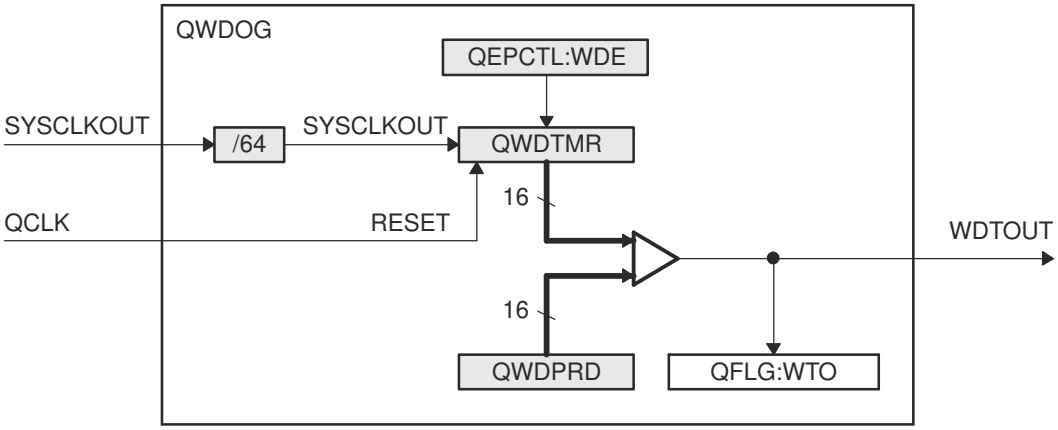


Figure 27-20. eQEP Watchdog Timer

27.8 eQEP Unit Timer Base

The eQEP peripheral includes a 32-bit timer (QUTMR) that is clocked by SYSCLKOUT to generate periodic interrupts for velocity calculations, see Figure 27-21. Whenever the unit timer (QUTMR) matches the unit period register (QUPRD), the eQEP peripheral resets the unit timer (QUTMR) and also generates the unit time out interrupt flag (QFLG[UTO]). The unit timer gets reset whenever timer value equals to configured period value.

The eQEP peripheral can be configured to latch the position counter, capture timer, and capture period values on a unit time out event so that latched values are used for velocity calculation as described in Section 27.6.

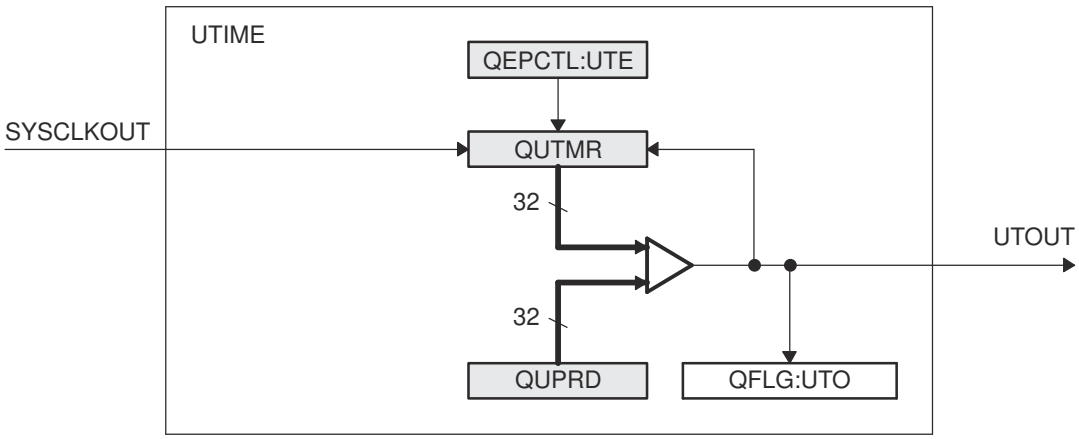


Figure 27-21. eQEP Unit Timer Base

27.9 QMA Module

The QEP Mode Adapter (QMA) is designed to extend the C2000™ eQEP module capabilities to support the additional modes described. Figure 27-22 depicts how the QMA module is integrated into to the eQEP module.

At reset, by default QMA logic is bypassed and the EQEPA and EQEPB inputs from the pins go directly into the eQEP module. When QMA module is enabled by configuring the QMACTRL[MODE] register, the EQEPA and EQEPB input are processed by this module and modified version of EQEPA and EQEPB signals are sent to the eQEP module. The QMA module requires the eQEP module to be configured in the Direction-Count mode and generates a clock signal on EQEPA input and direction signal on EQEPB input as needed for the proper operation of the intended mode.

- The xCLKMOD block inside the QMA module looks at the transitions on external EQEPA and EQEPB signals to generate the clock signal on the EQEPA input to the eQEP module.
- The xDIRMOD block inside the QMA module looks at the transitions on external EQEPA and EQEPB signals to generate the direction signal on the EQEPB input to the eQEP module.

The QMA module has error detection logic to detect illegal transitions on EQEPA and EQEPB input signals. The QMA module's error and interrupt are integrated inside the eQEP module as described in Section 27.10. In addition, the QMACTRL register configuration can be locked using the QMALOCK register. Refer to the register description for more details.

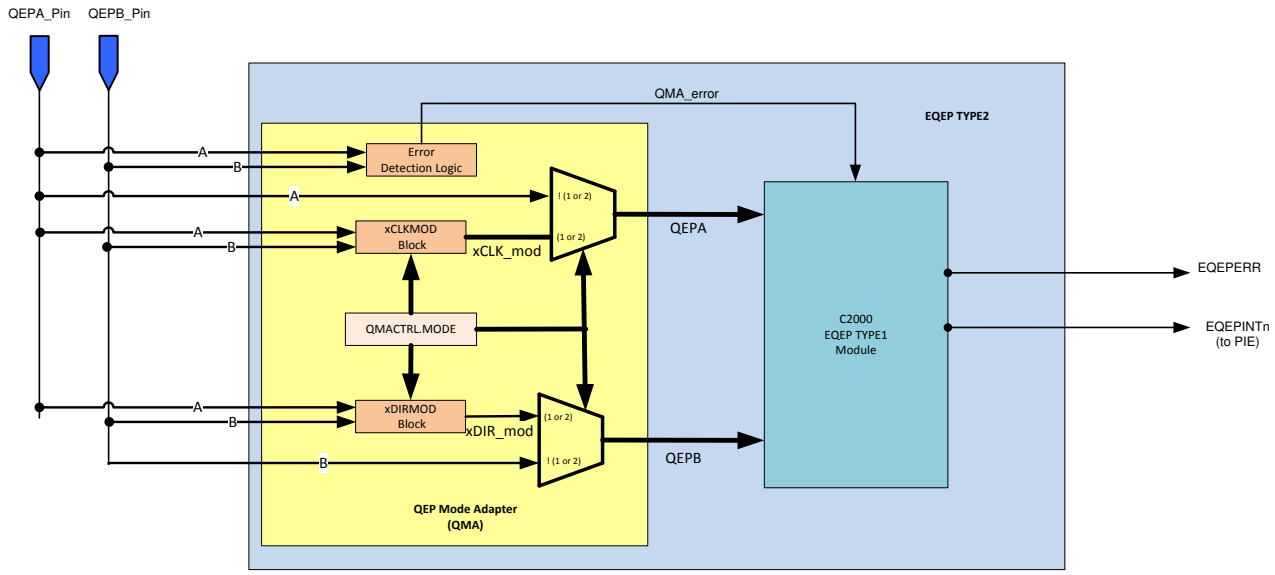


Figure 27-22. QMA Module Block Diagram

27.9.1 Modes of Operation

The QMA module can be operated in the following modes by configuring the QMACTRL register:

- QMA Mode-1 (QMACTRL[MODE]=1)
- QMA Mode-2 (QMACTRL[MODE]=2)

27.9.1.1 QMA Mode-1 (QMACTRL[MODE]=1)

This mode is used when the default state of EQEPA and EQEPB inputs is high. In this mode, outputs of QMA correspond to the following as shown in Figure 27-23:

- EQEPA Output of QMA is the AND of EQEPA and EQEPB inputs coming from the pin
- EQEPB Output of QMA is the direction signal generated by QMA based on EQEPA and EQEPB inputs

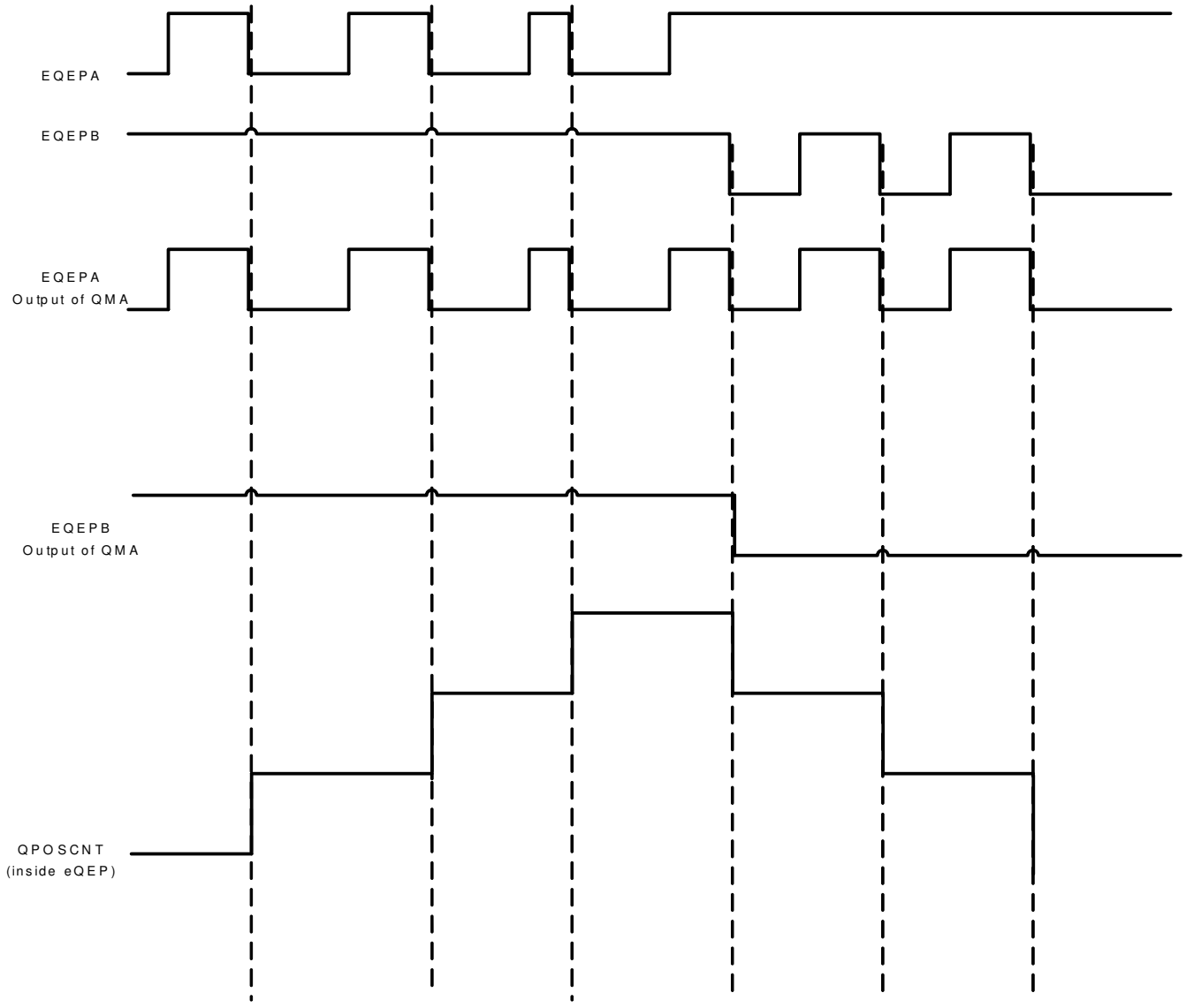


Figure 27-23. QMA Mode-1

27.9.1.2 QMA Mode-2 (QMACTRL[MODE]=2)

This mode is used when the default state of EQEPA and EQEPB inputs is low. In this mode, outputs of QMA correspond to the following as shown in Figure 27-24:

- EQEPA Output of QMA is the OR of EQEPA and EQEPB inputs coming from the pin
- EQEPB Output of QMA is the direction signal generated by QMA based on EQEPA and EQEPB inputs

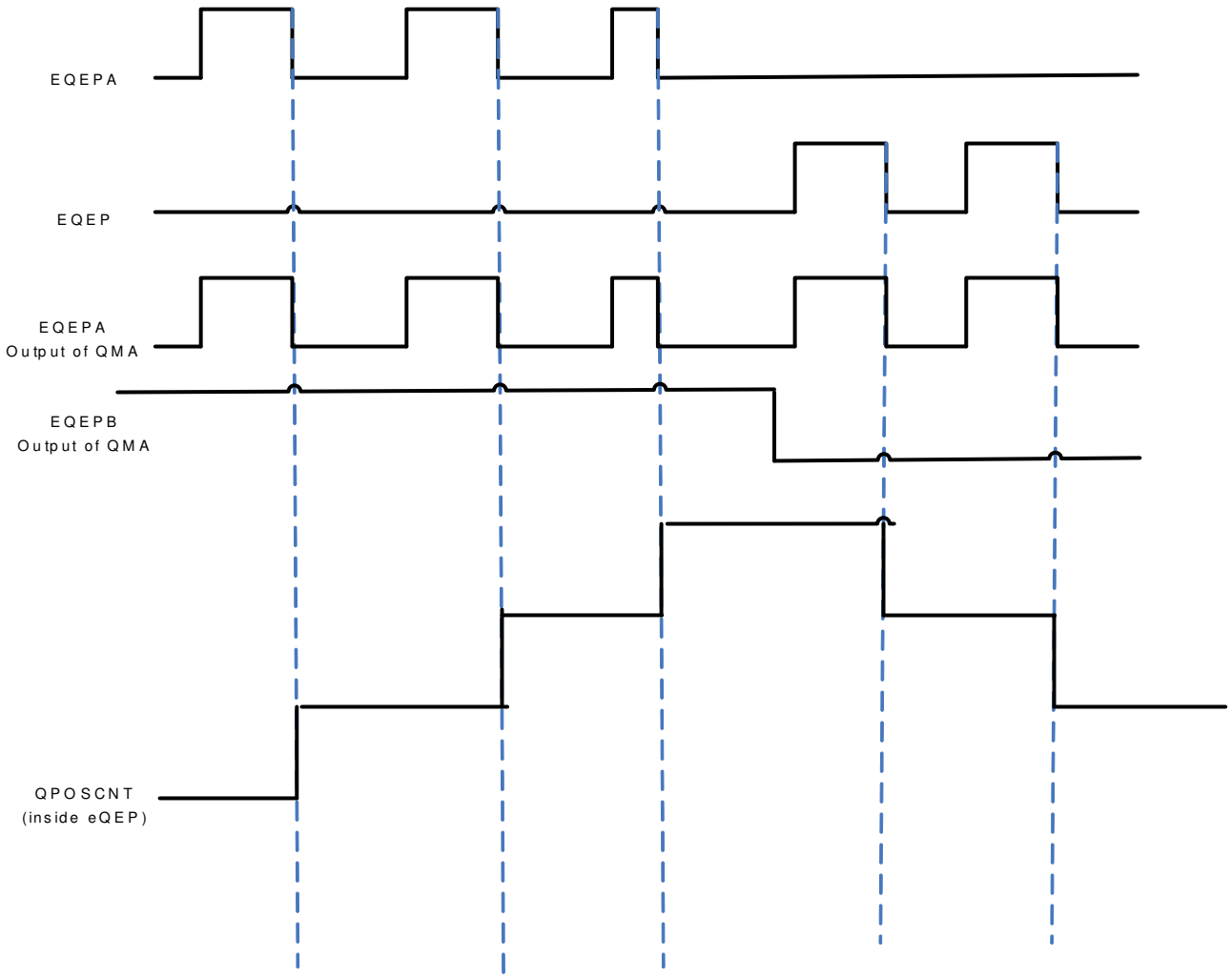


Figure 27-24. QMA Mode-2

27.9.2 Interrupt and Error Generation

The error detection logic detects illegal transitions on EQEPA and EQEPB signals and generates an error signal. This error signal can be used to generate eQEP interrupt and error output. Refer to Section 27.10 for details.

27.10 eQEP Interrupt Structure

Figure 27-25 shows how the interrupt mechanism works in the eQEP module.

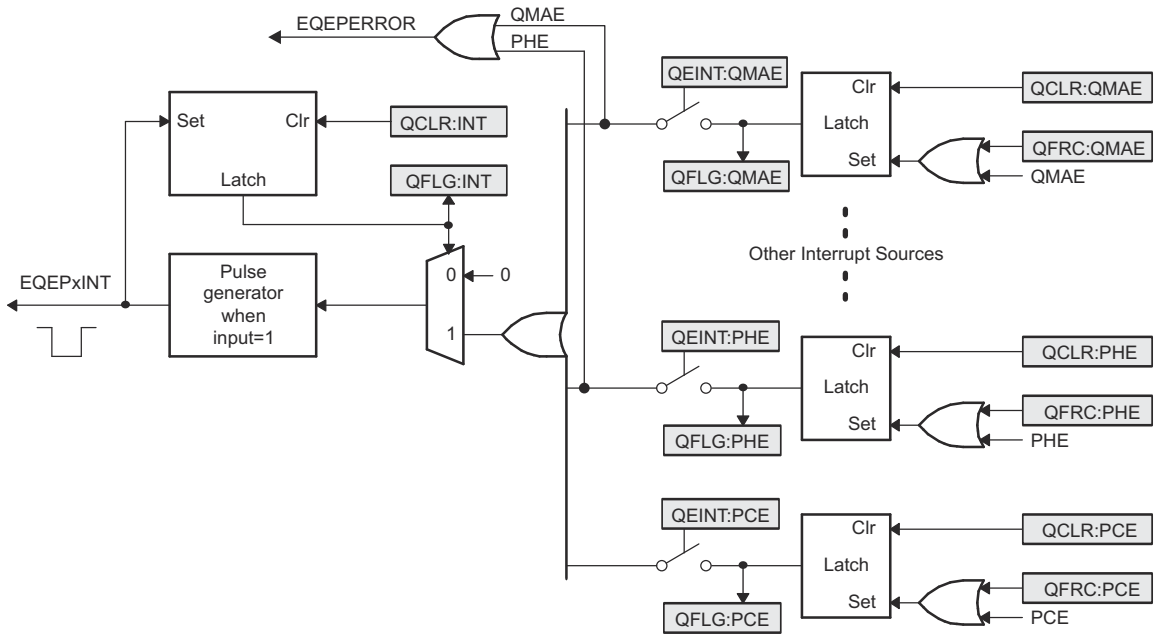


Figure 27-25. eQEP Interrupt Generation

Eleven interrupt events (PCE, PHE, QDC, WTO, PCU, PCO, PCR, PCM, SEL, IEL and UTO) can be generated. The interrupt control register (QEINT) is used to enable/disable individual interrupt event sources. The interrupt flag register (QFLG) indicates if any interrupt event has been latched and contains the global interrupt flag bit (INT).

An interrupt pulse is generated to PIE when:

1. Interrupt is enabled for eQEP event inside QEINT register
2. Interrupt flag for eQEP event inside QFLG register is set, and
3. Global interrupt status flag bit QFLG[INT] had been cleared for previously generated interrupt event. The interrupt service routine needs to clear the global interrupt flag bit and the serviced event, by way of the interrupt clear register (QCLR), before any other interrupt pulses are generated. If either flags inside the QFLG register are not cleared, further interrupt events do not generate an interrupt to PIE. You can force an interrupt event by way of the interrupt force register (QFRC), which is useful for test purposes.

27.11 Software

27.11.1 EQEP Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/eqep

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

27.11.1.1 Frequency Measurement Using eQEP

FILE: eqep_ex1_freq_cal.c

This example will calculate the frequency of an input signal using the eQEP module. ePWM1A is configured to generate this input signal with a frequency of 5 kHz. It will interrupt once every period and call the frequency calculation function. This example uses the IQMath library to simplify high-precision calculations.

In addition to the main example file, the following files must be included in this project:
- *eqep_ex1_calculation.c* - contains frequency calculation function
- *eqep_ex1_calculation.h* - includes initialization values for frequency structure

The configuration for this example is as follows
- Maximum frequency is configured to 10KHz (baseFreq)
- Minimum frequency is assumed at 50Hz for capture pre-scalar selection

SPEED_FR: High Frequency Measurement is obtained by counting the external input pulses for 10ms (unit timer set to 100Hz). \[SPEED_FR = \frac{Count\ Delta}{10ms} \]

SPEED_PR: Low Frequency Measurement is obtained by measuring time period of input edges. Time measurement is averaged over 64 edges for better results and the capture unit performs the time measurement using pre-scaled SYSCLK.

Note that the pre-scaler for capture unit clock is selected such that the capture timer does not overflow at the required minimum frequency. This example runs indefinitely until the user stops it.

For more information about the frequency calculation see the comments at the beginning of eqep_ex1_calculation.c and the XLS file provided with the project, eqep_ex1_calculation.xls.

27.11.1.2 Position and Speed Measurement Using eQEP

FILE: eqep_ex2_pos_speed.c

This example provides position and speed measurement using the capture unit and speed measurement using unit time out of the eQEP module. ePWM1 and a GPIO are configured to generate simulated eQEP signals. The ePWM module will interrupt once every period and call the position/speed calculation function. This example uses the IQMath library to simplify high-precision calculations.

In addition to the main example file, the following files must be included in this project:
- *eqep_ex2_calculation.c* - contains position/speed calculation function
- *eqep_ex2_calculation.h* - includes initialization values for position/speed structure

The configuration for this example is as follows
- Maximum speed is configured to 6000rpm (baseRPM)
- Minimum speed is assumed at 10rpm for capture pre-scalar selection
- Pole pair is configured to 2 (polePairs)
- Encoder resolution is configured to 4000 counts/revolution (mechScaler)
- Which means: 4000 / 4 = 1000 line/revolution quadrature encoder (simulated by ePWM1)
- ePWM1 (simulating QEP encoder signals) is configured for a 5kHz frequency or 300 rpm (= 4 * 5000 cnts/sec * 60 sec/min) / 4000 cnts/rev)

SPEEDRPM_FR: High Speed Measurement is obtained by counting the QEP input pulses for 10ms (unit timer set to 100Hz).

SPEEDRPM_FR = (Position Delta / 10ms) * 60 rpm

SPEEDRPM_PR: Low Speed Measurement is obtained by measuring time period of QEP edges. Time measurement is averaged over 64 edges for better results and the capture unit performs the time measurement using pre-scaled SYSCLK.

Note that the pre-scaler for capture unit clock is selected such that the capture timer does not overflow at the required minimum frequency. This example runs indefinitely until the user stops it.

For more information about the position/speed calculation see the comments at the beginning of eqep_ex2_calculation.c and the XLS file provided with the project, eqep_ex2_calculation.xls.

External Connections

- Connect GPIO20/eQEP1A to GPIO0/ePWM1A (simulates eQEP Phase A signal)
- Connect GPIO21/eQEP1B to GPIO1/ePWM1B (simulates eQEP Phase B signal)
- Connect GPIO23/eQEP1I to GPIO2 (simulates eQEP Index Signal)

Watch Variables

- *posSpeed.speedRPMFR* - Speed meas. in rpm using QEP position counter
- *posSpeed.speedRPMPR* - Speed meas. in rpm using capture unit
- *posSpeed.thetaMech* - Motor mechanical angle (Q15)
- *posSpeed.thetaElec* - Motor electrical angle (Q15)

27.11.1.3 ePWM frequency Measurement Using eQEP via xbar connection

FILE: eqep_ex3_epwm_xbar.c

This example will calculate the frequency of an PWM signal using the eQEP module. ePWM1A is configured to generate this input signal with a frequency of 5 kHz. This ePWM signal is connected to input of eQEP using Input CrossBar and EPWM XBAR. ePWM module will interrupt once every period and call the frequency calculation function. This example uses the IQMath library to simplify high-precision calculations.

In addition to the main example file, the following files must be included in this project:

- *eqep_ex1_calculation.c* - contains frequency calculation function
- *eqep_ex1_calculation.h* - includes initialization values for frequency structure

The configuration for this example is as follows

- Maximum frequency is configured to 10KHz (baseFreq)
- Minimum frequency is assumed at 50Hz for capture pre-scalar selection
- GPIO0 is connected to output of INPUT_XBAR1
- INPUT_XBAR1 is connected to output of PWMXBAR at TRIP4
- eQEPA source is configured as PWMXBAR.1 output (TRIP4)

SPEED_FR: High Frequency Measurement is obtained by counting the external input pulses for 10ms (unit timer set to 100Hz). \[SPEED_FR = \frac{Count\ Delta}{10ms} \]

SPEED_PR: Low Frequency Measurement is obtained by measuring time period of input edges. Time measurement is averaged over 64 edges for better results and the capture unit performs the time measurement using pre-scaled SYSCLK.

Note that the pre-scaler for capture unit clock is selected such that the capture timer does not overflow at the required minimum frequency. This example runs indefinitely until the user stops it.

For more information about the frequency calculation see the comments at the beginning of eqep_ex1_calculation.c and the XLS file provided with the project, eqep_ex1_calculation.xls.

Watch Variables

- *freq.freqHzFR* - Frequency measurement using position counter/unit time out
- *freq.freqHzPR* - Frequency measurement using capture unit

27.11.1.4 Frequency Measurement Using eQEP via unit timeout interrupt

FILE: eqep_ex4_freq_cal_interrupt.c

This example will calculate the frequency of an input signal using the eQEP module. ePWM1A is configured to generate this input signal with a frequency of 5 kHz. EQEP unit timeout is set which will generate an interrupt every *UNIT_PERIOD* microseconds and frequency calculation occurs continuously

The configuration for this example is as follows
- PWM frequency is specified as 5000Hz
- UNIT_PERIOD is specified as 10000 us
- Min frequency is (1/(2*10ms)) i.e 50Hz
- Highest frequency can be (2^32)/ ((2*10ms))
- Resolution of frequency measurement is 50hz

freq : Frequency Measurement is obtained by counting the external input pulses for UNIT_PERIOD (unit timer set to 10 ms).

ExternalConnections
- Connect GPIO20/eQEP1A to GPIO0/ePWM1A

Watch Variables
- *freq* - Frequency measurement using position counter/unit time out
- *pass* - If measured frequency matches with PWM frequency then pass = 1 else 0

27.11.1.5 Motor speed and direction measurement using eQEP via unit timeout interrupt

FILE: eqep_ex5_speed_dir_motor.c

This example can be used to sense the speed and direction of motor using eQEP in quadrature encoder mode. ePWM1A is configured to simulate motor encoder signals with frequency of 5 kHz on both A and B pins with 90 degree phase shift (so as to run this example without motor). EQEP unit timeout is set which will generate an interrupt every *UNIT_PERIOD* microseconds and speed calculation occurs continuously based on the direction of motor

The configuration for this example is as follows
- PWM frequency is specified as 5000Hz
- UNIT_PERIOD is specified as 10000 us
- Simulated quadrature signal frequency is 20000Hz (4 * 5000)
- Encoder holes assumed as 1000
- Thus Simulated motor speed is 300rpm (5000 * (60 / 1000))

freq : Simulated quadrature signal frequency measured by counting the external input pulses for UNIT_PERIOD (unit timer set to 10 ms). *speed* : Measure motor speed in rpm *dir* : Indicates clockwise (1) or anticlockwise (-1)

External Connections (if motor encoder signals are simulated by ePWM)

- Connect GPIO20/eQEP1A to GPIO0/ePWM1A
- Connect GPIO21/eQEP1B to GPIO1/ePWM1B With motor
- Connect GPIO20/eQEP1A to encoder A output
- Connect GPIO21/eQEP1B to encoder B output

Watch Variables
- *freq* : Simulated motor frequency measurement is obtained by counting the external input pulses for UNIT_PERIOD (unit timer set to 10 ms).
- *speed* : Measure motor speed in rpm
- *dir* : Indicates clockwise (1) or anticlockwise (-1)
- *pass* - If measured qudrature frequency matches with i.e. input quadrature frequency (4 * PWM frequency) then pass = 1 else fail = 1 (** only when "MOTOR" is commented out)

27.12 eQEP Registers

This section describes the Enhanced Quadrature Encoder Pulse Registers.

27.12.1 EQEP Base Address Table (C28)

Table 27-4. EQEP Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
EQep1Regs	EQEP_REGS	EQEP1_BASE	0x0000_5100	YES	YES	YES	YES	YES
EQep2Regs	EQEP_REGS	EQEP2_BASE	0x0000_5140	YES	YES	YES	YES	YES
EQep3Regs	EQEP_REGS	EQEP3_BASE	0x0000_5180	YES	YES	YES	YES	YES

27.12.2 EQEP_REGS Registers

Table 27-5 lists the memory-mapped registers for the EQEP_REGS registers. All register offset addresses not listed in Table 27-5 should be considered as reserved locations and the register contents should not be modified.

Table 27-5. EQEP_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	QPOSCNT	Position Counter		Go
2h	QPOSINIT	Position Counter Init		Go
4h	QPOSMAX	Maximum Position Count		Go
6h	QPOSCMP	Position Compare		Go
8h	QPOSILAT	Index Position Latch		Go
Ah	QPOSSLAT	Strobe Position Latch		Go
Ch	QPOSLAT	Position Latch		Go
Eh	QUTMR	QEP Unit Timer		Go
10h	QUPRD	QEP Unit Period		Go
12h	QWDTMR	QEP Watchdog Timer		Go
13h	QWDPRD	QEP Watchdog Period		Go
14h	QDECCTL	Quadrature Decoder Control		Go
15h	QEPCTL	QEP Control		Go
16h	QCAPCTL	Qaudrature Capture Control		Go
17h	QPOSCTL	Position Compare Control		Go
18h	QEINT	QEP Interrupt Control		Go
19h	QFLG	QEP Interrupt Flag		Go
1Ah	QCLR	QEP Interrupt Clear		Go
1Bh	QFRC	QEP Interrupt Force		Go
1Ch	QEPSTS	QEP Status		Go
1Dh	QCTMR	QEP Capture Timer		Go
1Eh	QCPRD	QEP Capture Period		Go
1Fh	QCTMRLAT	QEP Capture Latch		Go
20h	QCPRDLAT	QEP Capture Period Latch		Go
30h	REV	QEP Revision Number		Go
32h	QEPSTROBESEL	QEP Strobe select register		Go
34h	QMACTRL	QMA Control register		Go
36h	QEPSRCSEL	QEP Source Select Register		Go

Complex bit access types are encoded to fit into small table cells. Table 27-6 shows the codes that are used for access types in this section.

Table 27-6. EQEP_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear

Table 27-6. EQEP_REGS Access Type Codes (continued)

Access Type	Code	Description
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

27.12.2.1 QPOSCNT Register (Offset = 0h) [Reset = 0h]

QPOSCNT is shown in Figure 27-26 and described in Table 27-7.

Return to the Summary Table.

Position Counter

Figure 27-26. QPOSCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSCNT																															
R/W-0h																															

Table 27-7. QPOSCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSCNT	R/W	0h	Position Counter This 32-bit position counter register counts up/down on every eQEP pulse based on direction input. This counter acts as a position integrator whose count value is proportional to position from a give reference point. This Register acts as a Read ONLY register while counter is counting up/down. Note: It is recommended to only write to the position counter register (QPOSCNT) during initialization, i.e. when the eQEP position counter is disabled (QPEN bit of QEPCTL is zero). Once the position counter is enabled (QPEN bit is one), writing to the eQEP position counter register (QPOSCNT) may cause unexpected results. Reset type: SYSRSn

27.12.2.2 QPOSINIT Register (Offset = 2h) [Reset = 0h]

QPOSINIT is shown in Figure 27-27 and described in Table 27-8.

Return to the Summary Table.

Position Counter Init

Figure 27-27. QPOSINIT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSINIT																															
R/W-0h																															

Table 27-8. QPOSINIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSINIT	R/W	0h	Position Counter Init This register contains the position value that is used to initialize the position counter based on external strobe or index event. The position counter can be initialized through software. Writes to this register should always be full 32-bit writes. Reset type: SYSRSn

27.12.2.3 QPOSMAX Register (Offset = 4h) [Reset = 0h]

QPOSMAX is shown in Figure 27-28 and described in Table 27-9.

Return to the Summary Table.

Maximum Position Count

Figure 27-28. QPOSMAX Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSMAX																															
R/W-0h																															

Table 27-9. QPOSMAX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSMAX	R/W	0h	Maximum Position Count This register contains the maximum position counter value. Writes to this register should always be full 32-bit writes. Reset type: SYSRSn

27.12.2.4 QPOSCMP Register (Offset = 6h) [Reset = 0h]

QPOSCMP is shown in Figure 27-29 and described in Table 27-10.

Return to the Summary Table.

Position Compare

Figure 27-29. QPOSCMP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSCMP																															
R/W-0h																															

Table 27-10. QPOSCMP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSCMP	R/W	0h	Position Compare The position-compare value in this register is compared with the position counter (QPOSCNT) to generate sync output and/or interrupt on compare match. Writes to this register should always be full 32-bit writes. Reset type: SYSRSn

27.12.2.5 QPOSILAT Register (Offset = 8h) [Reset = 0h]

QPOSILAT is shown in Figure 27-30 and described in Table 27-11.

Return to the Summary Table.

Index Position Latch

Figure 27-30. QPOSILAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSILAT																															
R-0h																															

Table 27-11. QPOSILAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSILAT	R	0h	Index Position Latch The position-counter value is latched into this register on an index event as defined by the QEPCTL[IEL] bits. Reset type: SYSRSn

27.12.2.6 QPOSSLAT Register (Offset = Ah) [Reset = 0h]

QPOSSLAT is shown in Figure 27-31 and described in Table 27-12.

Return to the Summary Table.

Strobe Position Latch

Figure 27-31. QPOSSLAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSSLAT																															
R-0h																															

Table 27-12. QPOSSLAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSSLAT	R	0h	Strobe Position Latch The position-counter value is latched into this register on a strobe event as defined by the QEPCTL[SEL] bits. Reset type: SYSRSn

27.12.2.7 QPOSLAT Register (Offset = Ch) [Reset = 0h]

QPOSLAT is shown in Figure 27-32 and described in Table 27-13.

Return to the Summary Table.

Position Latch

Figure 27-32. QPOSLAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSLAT																															
R-0h																															

Table 27-13. QPOSLAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSLAT	R	0h	Position Latch The position-counter value is latched into this register on a unit time out event. Reset type: SYSRSn

27.12.2.8 QUTMR Register (Offset = Eh) [Reset = 0h]

QUTMR is shown in Figure 27-33 and described in Table 27-14.

Return to the Summary Table.

QEP Unit Timer

Figure 27-33. QUTMR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUTMR																															
R/W-0h																															

Table 27-14. QUTMR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QUTMR	R/W	0h	QEP Unit Timer This register acts as time base for unit time event generation. When this timer value matches the unit time period value a unit time event is generated. Writes to this register should always be full 32-bit writes. Reset type: SYSRSn

27.12.2.9 QUPRD Register (Offset = 10h) [Reset = 0h]

QUPRD is shown in Figure 27-34 and described in Table 27-15.

Return to the Summary Table.

QEP Unit Period

Figure 27-34. QUPRD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUPRD																															
R/W-0h																															

Table 27-15. QUPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QUPRD	R/W	0h	QEP Unit Period This register contains the period count for the unit timer to generate periodic unit time events. These events latch the eQEP position information at periodic intervals and optionally generate an interrupt. Writes to this register should always be full 32-bit writes. Reset type: SYSRSn

27.12.2.10 QWDTMR Register (Offset = 12h) [Reset = 0h]

QWDTMR is shown in Figure 27-35 and described in Table 27-16.

Return to the Summary Table.

QEP Watchdog Timer

Figure 27-35. QWDTMR Register

15	14	13	12	11	10	9	8
QWDTMR							
R/W-0h							
7	6	5	4	3	2	1	0
QWDTMR							
R/W-0h							

Table 27-16. QWDTMR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QWDTMR	R/W	0h	QEP Watchdog Timer This register acts as time base for the watchdog to detect motor stalls. When this timer value matches with the watchdog's period value a watchdog timeout interrupt is generated. This register is reset upon edge transition in quadrature-clock indicating the motion. Reset type: SYSRSn

27.12.2.11 QWDPRD Register (Offset = 13h) [Reset = 0h]

QWDPRD is shown in Figure 27-36 and described in Table 27-17.

Return to the Summary Table.

QEP Watchdog Period

Figure 27-36. QWDPRD Register

15	14	13	12	11	10	9	8
QWDPRD							
R/W-0h							
7	6	5	4	3	2	1	0
QWDPRD							
R/W-0h							

Table 27-17. QWDPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QWDPRD	R/W	0h	QEP Watchdog Period This register contains the time-out count for the eQEP peripheral watch dog timer. When the watchdog timer value matches the watchdog period value, a watchdog timeout interrupt is generated. Reset type: SYSRSn

27.12.2.12 QDECCTL Register (Offset = 14h) [Reset = 0h]

QDECCTL is shown in Figure 27-37 and described in Table 27-18.

Return to the Summary Table.

Quadrature Decoder Control

Figure 27-37. QDECCTL Register

15	14	13	12	11	10	9	8
QSRC		SOEN	SPSEL	XCR	SWAP	IGATE	QAP
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
QBP	QIP	QSP	RESERVED				QIDIRE
R/W-0h	R/W-0h	R/W-0h	R-0h				R/W-0h

Table 27-18. QDECCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	QSRC	R/W	0h	Position-counter source selection Reset type: SYSRSn 0h (R/W) = Quadrature count mode (QCLK = iCLK, QDIR = iDIR) 1h (R/W) = Direction-count mode (QCLK = xCLK, QDIR = xDIR) 2h (R/W) = UP count mode for frequency measurement (QCLK = xCLK, QDIR = 1) 3h (R/W) = DOWN count mode for frequency measurement (QCLK = xCLK, QDIR = 0)
13	SOEN	R/W	0h	Sync output-enable Reset type: SYSRSn 0h (R/W) = Disable position-compare sync output 1h (R/W) = Enable position-compare sync output
12	SPSEL	R/W	0h	Sync output pin selection Reset type: SYSRSn 0h (R/W) = Index pin is used for sync output 1h (R/W) = Strobe pin is used for sync output
11	XCR	R/W	0h	External Clock Rate Reset type: SYSRSn 0h (R/W) = 2x resolution: Count the rising/falling edge 1h (R/W) = 1x resolution: Count the rising edge only
10	SWAP	R/W	0h	CLK/DIR Signal Source for Position Counter Reset type: SYSRSn 0h (R/W) = Quadrature-clock inputs are not swapped 1h (R/W) = Quadrature-clock inputs are swapped
9	IGATE	R/W	0h	Index pulse gating option Reset type: SYSRSn 0h (R/W) = Disable gating of Index pulse 1h (R/W) = Gate the index pin with strobe
8	QAP	R/W	0h	QEPA input polarity Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Negates QEPA input
7	QBP	R/W	0h	QEPB input polarity Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Negates QEPB input
6	QIP	R/W	0h	QEPI input polarity Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Negates QEPI input

Table 27-18. QDECCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	QSP	R/W	0h	QEPS input polarity Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Negates QEPS input
4-1	RESERVED	R	0h	Reserved
0	QIDIRE	R/W	0h	0 - Compatible mode, Behavior same as existing devices 1 - Enhancement for Direction change during Index will be enabled: On QEPI direction change, the incoming posedge of QA can erroneously update/reset the position counter of the eQEP. This bit only needs to be enabled if the application requires a direction change occurring at the same time as an incoming QEPI signal, or when erroneous PC resets are observed. Reset type: SYSRSn

27.12.2.13 QEPCTL Register (Offset = 15h) [Reset = 0h]

QEPCTL is shown in Figure 27-38 and described in Table 27-19.

Return to the Summary Table.

QEP Control

Figure 27-38. QEPCTL Register

15	14	13	12	11	10	9	8
FREE_SOFT		PCRM		SEI		IEI	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
SWI	SEL	IEL		QPEN	QCLM	UTE	WDE
R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 27-19. QEPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	FREE_SOFT	R/W	0h	Emulation mode Reset type: SYSRSn 0h (R/W) = QPOSCNT behavior Position counter stops immediately on emulation suspend 0h (R/W) = QWDTMR behavior Watchdog counter stops immediately 0h (R/W) = QUTMR behavior Unit timer stops immediately 0h (R/W) = QCTMR behavior Capture Timer stops immediately 1h (R/W) = QPOSCNT behavior Position counter continues to count until the rollover 1h (R/W) = QWDTMR behavior Watchdog counter counts until WD period match roll over 1h (R/W) = QUTMR behavior Unit timer counts until period rollover 1h (R/W) = QCTMR behavior Capture Timer counts until next unit period event 2h (R/W) = QPOSCNT behavior Position counter is unaffected by emulation suspend 2h (R/W) = QWDTMR behavior Watchdog counter is unaffected by emulation suspend 2h (R/W) = QUTMR behavior Unit timer is unaffected by emulation suspend 2h (R/W) = QCTMR behavior Capture Timer is unaffected by emulation suspend 3h (R/W) = Same as FREE_SOFT_2
13-12	PCRM	R/W	0h	Postion counter reset Reset type: SYSRSn 0h (R/W) = Position counter reset on an index event 1h (R/W) = Position counter reset on the maximum position 2h (R/W) = Position counter reset on the first index event 3h (R/W) = Position counter reset on a unit time event
11-10	SEI	R/W	0h	Strobe event initialization of position counter Reset type: SYSRSn 0h (R/W) = Does nothing (action disabled) 1h (R/W) = Does nothing (action disabled) 2h (R/W) = Initializes the position counter on rising edge of the QEPS signal 3h (R/W) = Clockwise Direction: Initializes the position counter on the rising edge of QEPS strobe Counter Clockwise Direction: Initializes the position counter on the falling edge of QEPS strobe

Table 27-19. QEPCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9-8	IEI	R/W	0h	Index event init of position count Reset type: SYSRSn 0h (R/W) = Do nothing (action disabled) 1h (R/W) = Do nothing (action disabled) 2h (R/W) = Initializes the position counter on the rising edge of the QEPI signal (QPOSCNT = QPOSINIT) 3h (R/W) = Initializes the position counter on the falling edge of QEPI signal (QPOSCNT = QPOSINIT)
7	SWI	R/W	0h	Software init position counter Reset type: SYSRSn 0h (R/W) = Do nothing (action disabled) 1h (R/W) = Initialize position counter (QPOSCNT=QPOSINIT). This bit is not cleared automatically
6	SEL	R/W	0h	Strobe event latch of position counter Reset type: SYSRSn 0h (R/W) = The position counter is latched on the rising edge of QEPS strobe (QPOSSLAT = POSCCNT). Latching on the falling edge can be done by inverting the strobe input using the QSP bit in the QDECCTL register 1h (R/W) = Clockwise Direction: Position counter is latched on rising edge of QEPS strobe Counter Clockwise Direction: Position counter is latched on falling edge of QEPS strobe
5-4	IEL	R/W	0h	Index event latch of position counter (software index marker) Reset type: SYSRSn 0h (R/W) = Reserved 1h (R/W) = Latches position counter on rising edge of the index signal 2h (R/W) = Latches position counter on falling edge of the index signal 3h (R/W) = Software index marker. Latches the position counter and quadrature direction flag on index event marker. The position counter is latched to the QPOSILAT register and the direction flag is latched in the QEPSTS[QDLF] bit. This mode is useful for software index marking.
3	QPEN	R/W	0h	Quadrature position counter enable/software reset Reset type: SYSRSn 0h (R/W) = Reset the eQEP peripheral internal operating flags/read-only registers. Control/configuration registers are not disturbed by a software reset. When QPEN is disabled, some flags in the QFLG register do not get reset or cleared and show the actual state of that flag. 1h (R/W) = eQEP position counter is enabled
2	QCLM	R/W	0h	QEP capture latch mode Reset type: SYSRSn 0h (R/W) = Latch on position counter read by CPU. Capture timer and capture period values are latched into QCTMRLAT and QCPRDLAT registers when CPU reads the QPOSCNT register. 1h (R/W) = Latch on unit time out. Position counter, capture timer and capture period values are latched into QPOSLAT, QCTMRLAT and QCPRDLAT registers on unit time out.
1	UTE	R/W	0h	QEP unit timer enable Reset type: SYSRSn 0h (R/W) = Disable eQEP unit timer 1h (R/W) = Enable unit timer
0	WDE	R/W	0h	QEP watchdog enable Reset type: SYSRSn 0h (R/W) = Disable the eQEP watchdog timer 1h (R/W) = Enable the eQEP watchdog timer

27.12.2.14 QCAPCTL Register (Offset = 16h) [Reset = 0h]

QCAPCTL is shown in Figure 27-39 and described in Table 27-20.

Return to the Summary Table.

Qaudrature Capture Control

Figure 27-39. QCAPCTL Register

15	14	13	12	11	10	9	8
CEN	RESERVED						
R/W-0h	R-0h						
7	6	5	4	3	2	1	0
RESERVED	CCPS			UPPS			
R-0h	R/W-0h			R/W-0h			

Table 27-20. QCAPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	CEN	R/W	0h	Enable eQEP capture Reset type: SYSRSn 0h (R/W) = eQEP capture unit is disabled 1h (R/W) = eQEP capture unit is enabled
14-7	RESERVED	R	0h	Reserved
6-4	CCPS	R/W	0h	eQEP capture timer clock prescaler Reset type: SYSRSn 0h (R/W) = CAPCLK = SYSCLKOUT/1 1h (R/W) = CAPCLK = SYSCLKOUT/2 2h (R/W) = CAPCLK = SYSCLKOUT/4 3h (R/W) = CAPCLK = SYSCLKOUT/8 4h (R/W) = CAPCLK = SYSCLKOUT/16 5h (R/W) = CAPCLK = SYSCLKOUT/32 6h (R/W) = CAPCLK = SYSCLKOUT/64 7h (R/W) = CAPCLK = SYSCLKOUT/128
3-0	UPPS	R/W	0h	Unit position event prescaler Reset type: SYSRSn 0h (R/W) = UPEVNT = QCLK/1 1h (R/W) = UPEVNT = QCLK/2 2h (R/W) = UPEVNT = QCLK/4 3h (R/W) = UPEVNT = QCLK/8 4h (R/W) = UPEVNT = QCLK/16 5h (R/W) = UPEVNT = QCLK/32 6h (R/W) = UPEVNT = QCLK/64 7h (R/W) = UPEVNT = QCLK/128 8h (R/W) = UPEVNT = QCLK/256 9h (R/W) = UPEVNT = QCLK/512 Ah (R/W) = UPEVNT = QCLK/1024 Bh (R/W) = UPEVNT = QCLK/2048 Ch (R/W) = Reserved Dh (R/W) = Reserved Eh (R/W) = Reserved Fh (R/W) = Reserved

27.12.2.15 QPOSCTL Register (Offset = 17h) [Reset = 0h]

QPOSCTL is shown in Figure 27-40 and described in Table 27-21.

Return to the Summary Table.

Position Compare Control

Figure 27-40. QPOSCTL Register

15	14	13	12	11	10	9	8
PCSHDW	PCLOAD	PCPOL	PCE	PCSPW			
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h			
7	6	5	4	3	2	1	0
PCSPW							
R/W-0h							

Table 27-21. QPOSCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	PCSHDW	R/W	0h	Position compare of shadow enable Reset type: SYSRSn 0h (R/W) = Shadow disabled, load Immediate 1h (R/W) = Shadow enabled
14	PCLOAD	R/W	0h	Position compare of shadow load Reset type: SYSRSn 0h (R/W) = Load on QPOSCNT = 0 1h (R/W) = Load when QPOSCNT = QPOSCMP
13	PCPOL	R/W	0h	Polarity of sync output Reset type: SYSRSn 0h (R/W) = Active HIGH pulse output 1h (R/W) = Active LOW pulse output
12	PCE	R/W	0h	Position compare enable/disable Reset type: SYSRSn 0h (R/W) = Disable position compare unit 1h (R/W) = Enable position compare unit
11-0	PCSPW	R/W	0h	Select-position-compare sync output pulse width Reset type: SYSRSn 0h (R/W) = 1 * 4 * SYSCLKOUT cycles 1h (R/W) = 2 * 4 * SYSCLKOUT cycles FFFh (R/W) = 4096 * 4 * SYSCLKOUT cycles

27.12.2.16 QEINT Register (Offset = 18h) [Reset = 0h]

QEINT is shown in Figure 27-41 and described in Table 27-22.

Return to the Summary Table.

QEP Interrupt Control

Figure 27-41. QEINT Register

15	14	13	12	11	10	9	8
RESERVED			QMAE	UTO	IEL	SEL	PCM
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	QPE	PCE	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

Table 27-22. QEINT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12	QMAE	R/W	0h	QMA Error Interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
11	UTO	R/W	0h	Unit time out interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
10	IEL	R/W	0h	Index event latch interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
9	SEL	R/W	0h	Strobe event latch interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
8	PCM	R/W	0h	Position-compare match interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
7	PCR	R/W	0h	Position-compare ready interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
6	PCO	R/W	0h	Position counter overflow interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
5	PCU	R/W	0h	Position counter underflow interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
4	WTO	R/W	0h	Watchdog time out interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
3	QDC	R/W	0h	Quadrature direction change interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled

Table 27-22. QEINT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	QPE	R/W	0h	Quadrature phase error interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
1	PCE	R/W	0h	Position counter error interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
0	RESERVED	R	0h	Reserved

27.12.2.17 QFLG Register (Offset = 19h) [Reset = 0h]

QFLG is shown in Figure 27-42 and described in Table 27-23.

Return to the Summary Table.

QEP Interrupt Flag

Figure 27-42. QFLG Register

15	14	13	12	11	10	9	8
RESERVED			QMAE	UTO	IEL	SEL	PCM
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	INT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 27-23. QFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12	QMAE	R	0h	QMA Error interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = Interrupt was generated
11	UTO	R	0h	Unit time out interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = Set by eQEP unit timer period match
10	IEL	R	0h	Index event latch interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = This bit is set after latching the QPOSCNT to QPOSILAT
9	SEL	R	0h	Strobe event latch interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = This bit is set after latching the QPOSCNT to QPOSSLAT
8	PCM	R	0h	eQEP compare match event interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = This bit is set on position-compare match
7	PCR	R	0h	Position-compare ready interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = This bit is set after transferring the shadow register value to the active position compare register
6	PCO	R	0h	Position counter overflow interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = This bit is set on position counter overflow.
5	PCU	R	0h	Position counter underflow interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = This bit is set on position counter underflow.
4	WTO	R	0h	Watchdog timeout interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = Set by watchdog timeout
3	QDC	R	0h	Quadrature direction change interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = Interrupt was generated

Table 27-23. QFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	PHE	R	0h	Quadrature phase error interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = Set on simultaneous transition of QEPA and QEPB
1	PCE	R	0h	Position counter error interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = Position counter error
0	INT	R	0h	Global interrupt status flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = Interrupt was generated

27.12.2.18 QCLR Register (Offset = 1Ah) [Reset = 0h]

QCLR is shown in Figure 27-43 and described in Table 27-24.

Return to the Summary Table.

QEP Interrupt Clear

Figure 27-43. QCLR Register

15	14	13	12	11	10	9	8
RESERVED			QMAE	UTO	IEL	SEL	PCM
R-0h			R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	INT
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 27-24. QCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12	QMAE	R-0/W1S	0h	Clear QMA Error interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
11	UTO	R-0/W1S	0h	Clear unit time out interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
10	IEL	R-0/W1S	0h	Clear index event latch interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
9	SEL	R-0/W1S	0h	Clear strobe event latch interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
8	PCM	R-0/W1S	0h	Clear eQEP compare match event interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
7	PCR	R-0/W1S	0h	Clear position-compare ready interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
6	PCO	R-0/W1S	0h	Clear position counter overflow interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
5	PCU	R-0/W1S	0h	Clear position counter underflow interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
4	WTO	R-0/W1S	0h	Clear watchdog timeout interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
3	QDC	R-0/W1S	0h	Clear quadrature direction change interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag

Table 27-24. QCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	PHE	R-0/W1S	0h	Clear quadrature phase error interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
1	PCE	R-0/W1S	0h	Clear position counter error interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
0	INT	R-0/W1S	0h	Global interrupt clear flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag

27.12.2.19 QFRC Register (Offset = 1Bh) [Reset = 0h]

QFRC is shown in Figure 27-44 and described in Table 27-25.

Return to the Summary Table.

QEP Interrupt Force

Figure 27-44. QFRC Register

15	14	13	12	11	10	9	8
RESERVED			QMAE	UTO	IEL	SEL	PCM
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

Table 27-25. QFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12	QMAE	R/W	0h	Force QMA error interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
11	UTO	R/W	0h	Force unit time out interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
10	IEL	R/W	0h	Force index event latch interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
9	SEL	R/W	0h	Force strobe event latch interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
8	PCM	R/W	0h	Force position-compare match interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
7	PCR	R/W	0h	Force position-compare ready interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
6	PCO	R/W	0h	Force position counter overflow interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
5	PCU	R/W	0h	Force position counter underflow interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
4	WTO	R/W	0h	Force watchdog time out interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
3	QDC	R/W	0h	Force quadrature direction change interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt

Table 27-25. QFRC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	PHE	R/W	0h	Force quadrature phase error interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
1	PCE	R/W	0h	Force position counter error interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
0	RESERVED	R	0h	Reserved

27.12.2.20 QEPSTS Register (Offset = 1Ch) [Reset = 0h]

QEPSTS is shown in Figure 27-45 and described in Table 27-26.

Return to the Summary Table.

QEP Status

Figure 27-45. QEPSTS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
UPEVNT	FIDF	QDF	QDLF	COEF	CDEF	FIMF	PCEF
R/W1C-0h	R-0h	R-0h	R-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R-0h

Table 27-26. QEPSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	UPEVNT	R/W1C	0h	Unit position event flag Reset type: SYSRSn 0h (R/W) = No unit position event detected 1h (R/W) = Unit position event detected. Write 1 to clear
6	FIDF	R	0h	Direction on the first index marker Status of the direction is latched on the first index event marker. Reset type: SYSRSn 0h (R/W) = Counter-clockwise rotation (or reverse movement) on the first index event 1h (R/W) = Clockwise rotation (or forward movement) on the first index event
5	QDF	R	0h	Quadrature direction flag Reset type: SYSRSn 0h (R/W) = Counter-clockwise rotation (or reverse movement) 1h (R/W) = Clockwise rotation (or forward movement)
4	QDLF	R	0h	eQEP direction latch flag Reset type: SYSRSn 0h (R/W) = Counter-clockwise rotation (or reverse movement) on index event marker 1h (R/W) = Clockwise rotation (or forward movement) on index event marker
3	COEF	R/W1C	0h	Capture overflow error flag Reset type: SYSRSn 0h (R/W) = Overflow has not occurred. 1h (R/W) = Overflow occurred in eQEP Capture timer (QEPCTMR). This bit is cleared by writing a '1'.
2	CDEF	R/W1C	0h	Capture direction error flag Reset type: SYSRSn 0h (R/W) = Capture direction error has not occurred. 1h (R/W) = Direction change occurred between the capture position event. This bit is cleared by writing a '1'.
1	FIMF	R/W1C	0h	First index marker flag Reset type: SYSRSn 0h (R/W) = First index pulse has not occurred. 1h (R/W) = Set by first occurrence of index pulse. This bit is cleared by writing a '1'.
0	PCEF	R	0h	Position counter error flag. This bit is not sticky and it is updated for every index event. Reset type: SYSRSn 0h (R/W) = No error occurred during the last index transition 1h (R/W) = Position counter error

27.12.2.21 QCTMR Register (Offset = 1Dh) [Reset = 0h]

QCTMR is shown in Figure 27-46 and described in Table 27-27.

Return to the Summary Table.

QEP Capture Timer

Figure 27-46. QCTMR Register

15	14	13	12	11	10	9	8
QCTMR							
R/W-0h							
7	6	5	4	3	2	1	0
QCTMR							
R/W-0h							

Table 27-27. QCTMR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QCTMR	R/W	0h	This register provides time base for edge capture unit. Reset type: SYSRSn

27.12.2.22 QCPRD Register (Offset = 1Eh) [Reset = 0h]

QCPRD is shown in Figure 27-47 and described in Table 27-28.

Return to the Summary Table.

QEP Capture Period

Figure 27-47. QCPRD Register

15	14	13	12	11	10	9	8
QCPRD							
R/W-0h							
7	6	5	4	3	2	1	0
QCPRD							
R/W-0h							

Table 27-28. QCPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QCPRD	R/W	0h	This register holds the period count value between the last successive eQEP position events Reset type: SYSRSn

27.12.2.23 QCTMRLAT Register (Offset = 1Fh) [Reset = 0h]

QCTMRLAT is shown in Figure 27-48 and described in Table 27-29.

Return to the Summary Table.

QEP Capture Latch

Figure 27-48. QCTMRLAT Register

15	14	13	12	11	10	9	8
QCTMRLAT							
R-0h							
7	6	5	4	3	2	1	0
QCTMRLAT							
R-0h							

Table 27-29. QCTMRLAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QCTMRLAT	R	0h	The eQEP capture timer value can be latched into this register on two events viz., unit timeout event, reading the eQEP position counter. Reset type: SYSRSn

27.12.2.24 QCPRDLAT Register (Offset = 20h) [Reset = 0h]

QCPRDLAT is shown in Figure 27-49 and described in Table 27-30.

Return to the Summary Table.

QEP Capture Period Latch

Figure 27-49. QCPRDLAT Register

15	14	13	12	11	10	9	8
QCPRDLAT							
R-0h							
7	6	5	4	3	2	1	0
QCPRDLAT							
R-0h							

Table 27-30. QCPRDLAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QCPRDLAT	R	0h	eQEP capture period value can be latched into this register on two events viz., unit timeout event, reading the eQEP position counter. Reset type: SYSRSn

27.12.2.25 REV Register (Offset = 30h) [Reset = 9h]

REV is shown in Figure 27-50 and described in Table 27-31.

Return to the Summary Table.

QEP Revision Number

Figure 27-50. REV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED										MINOR			MAJOR		
R-0-0h										R-1h			R-1h		

Table 27-31. REV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5-3	MINOR	R	1h	This field specifies the Minor Revision number for the eQEP IP. Reset type: N/A
2-0	MAJOR	R	1h	This field specifies the Major Revision number for the eQEP IP. Reset type: N/A

27.12.2.26 QEPSTROBESEL Register (Offset = 32h) [Reset = 0h]

QEPSTROBESEL is shown in Figure 27-51 and described in Table 27-32.

Return to the Summary Table.

QEP Strobe select register

Figure 27-51. QEPSTROBESEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						STROBESEL	
R-0-0h						R/W-0h	

Table 27-32. QEPSTROBESEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R-0	0h	Reserved
1-0	STROBESEL	R/W	0h	Strobe source select: Reset type: SYSRSn 0h (R/W) = QEP Strobe after polarity mux 1h (R/W) = QEP Strobe after polarity mux 2h (R/W) = QEP Strobe after polarity mux ORed with ADCSOCA 3h (R/W) = QEP Strobe after polarity mux ORed with ADCSOCB

27.12.2.27 QMACTRL Register (Offset = 34h) [Reset = 0h]

QMACTRL is shown in Figure 27-52 and described in Table 27-33.

Return to the Summary Table.

QMA Control register

Figure 27-52. QMACTRL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													MODE		
R-0-0h													R/W-0h		

Table 27-33. QMACTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R-0	0h	Reserved
2-0	MODE	R/W	0h	Select Mode for QMA mode: 000 : QMA Module is bypassed. 001 : QMA Mode-1 operation selected 010 : QMA Mode-2 operation selected 011 : QMA Module is bypassed (reserved) 1xx : QMA Module is bypassed (reserved) Reset type: SYSRSn

27.12.2.28 QEPSRCSEL Register (Offset = 36h) [Reset = 0h]

QEPSRCSEL is shown in Figure 27-53 and described in Table 27-34.

Return to the Summary Table.

QEP Source Select Register

Figure 27-53. QEPSRCSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QEPSSEL				QEPISEL				QEPBSEL				QEPASEL			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 27-34. QEPSRCSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-12	QEPSSEL	R/W	0h	QEP Strobe source select: 0x0: From device Pins (Default). Others: Tied to zero. Reset type: SYSRSn
11-8	QEPISEL	R/W	0h	QEP Index source select: 0x0: Device Pin (Default) 0x1: CMPSS1.CTRIPH 0x2: CMPSS2.CTRIPH 0x3: CMPSS3.CTRIPH 0x4: CMPSS4.CTRIPH 0x5: CMPSS5.CTRIPH 0x6: CMPSS6.CTRIPH 0x7: CMPSS7.CTRIPH 0x8: CMPSS8.CTRIPH 0x9: PWMXBAR.1 0xA:PWMXBAR.2 0xB:PWMXBAR.3 0xC:PWMXBAR.4 0xD:PWMXBAR.5 0xE:PWMXBAR.6 0xF:PWMXBAR.7 Note: eQEP needs to be disabled before configuring these bits as it can lead to unexpected behavior if eQEP is running. Reset type: SYSRSn
7-4	QEPBSEL	R/W	0h	QEPB source select: 0x0: Device Pin (Default) 0x1: CMPSS1.CTRIPH 0x2: CMPSS2.CTRIPH 0x3: CMPSS3.CTRIPH 0x4: CMPSS4.CTRIPH 0x5: CMPSS5.CTRIPH 0x6: CMPSS6.CTRIPH 0x7: CMPSS7.CTRIPH 0x8: CMPSS8.CTRIPH 0x9: PWMXBAR.1 0xA:PWMXBAR.2 0xB:PWMXBAR.3 0xC:PWMXBAR.4 0xD:PWMXBAR.5 0xE:PWMXBAR.6 0xF:PWMXBAR.7 Note: eQEP needs to be disabled before configuring these bits as it can lead to unexpected behavior if eQEP is running. Reset type: SYSRSn

Table 27-34. QEPSRCSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	QEPASEL	R/W	0h	QEPA source select: 0x0: Device Pin (Default) 0x1: CMPSS1.CTRIPH 0x2: CMPSS2.CTRIPH 0x3: CMPSS3.CTRIPH 0x4: CMPSS4.CTRIPH 0x5: CMPSS5.CTRIPH 0x6: CMPSS6.CTRIPH 0x7: CMPSS7.CTRIPH 0x8: CMPSS8.CTRIPH 0x9: PWMXBAR.1 0xA:PWMXBAR.2 0xB:PWMXBAR.3 0xC:PWMXBAR.4 0xD:PWMXBAR.5 0xE:PWMXBAR.6 0xF:PWMXBAR.7 Note: eQEP needs to be disabled before configuring these bits as it can lead to unexpected behavior if eQEP is running. Reset type: SYSRSn

27.12.3 EQEP Registers to Driverlib Functions

Table 27-35. EQEP Registers to Driverlib Functions

File	Driverlib Function
QPOSCNT	
eqep.h	EQEP_getPosition
eqep.h	EQEP_setPosition
QPOSINIT	
eqep.h	EQEP_setInitialPosition
QPOSMAX	
eqep.h	EQEP_setPositionCounterConfig
QPOSCMP	
eqep.c	EQEP_setCompareConfig
QPOSILAT	
eqep.h	EQEP_getIndexPositionLatch
QPOSSLAT	
eqep.h	EQEP_getStrobePositionLatch
QPOSLAT	
eqep.h	EQEP_getPositionLatch
QUTMR	
-	
QUPRD	
eqep.h	EQEP_loadUnitTimer
eqep.h	EQEP_enableUnitTimer
QWDTMR	
eqep.h	EQEP_setWatchdogTimerValue
eqep.h	EQEP_getWatchdogTimerValue
QWDPRD	
eqep.h	EQEP_enableWatchdog
QDECCTL	
eqep.c	EQEP_setCompareConfig

Table 27-35. EQEP Registers to Driverlib Functions (continued)

File	Driverlib Function
eqep.c	EQEP_setInputPolarity
eqep.h	EQEP_setDecoderConfig
eqep.h	EQEP_enableDirectionChangeDuringIndex
eqep.h	EQEP_disableDirectionChangeDuringIndex
QEPCTL	
eqep.h	EQEP_enableModule
eqep.h	EQEP_disableModule
eqep.h	EQEP_setPositionCounterConfig
eqep.h	EQEP_enableUnitTimer
eqep.h	EQEP_disableUnitTimer
eqep.h	EQEP_enableWatchdog
eqep.h	EQEP_disableWatchdog
eqep.h	EQEP_setPositionInitMode
eqep.h	EQEP_setSWPositionInit
eqep.h	EQEP_setLatchMode
eqep.h	EQEP_setEmulationMode
QCAPCTL	
eqep.h	EQEP_setCaptureConfig
eqep.h	EQEP_enableCapture
eqep.h	EQEP_disableCapture
QPOSCTL	
eqep.c	EQEP_setCompareConfig
eqep.h	EQEP_enableCompare
eqep.h	EQEP_disableCompare
eqep.h	EQEP_setComparePulseWidth
QEINT	
eqep.h	EQEP_enableInterrupt
eqep.h	EQEP_disableInterrupt
QFLG	
eqep.h	EQEP_getInterruptStatus
eqep.h	EQEP_getError
QCLR	
eqep.h	EQEP_clearInterruptStatus
QFRC	
eqep.h	EQEP_forceInterrupt
QEPSTS	
eqep.h	EQEP_getDirection
eqep.h	EQEP_getStatus
eqep.h	EQEP_clearStatus
QCTMR	
eqep.h	EQEP_getCaptureTimer
eqep.h	EQEP_getCaptureTimerLatch
QCPRD	
eqep.h	EQEP_getCapturePeriod
eqep.h	EQEP_getCapturePeriodLatch

Table 27-35. EQEP Registers to Driverlib Functions (continued)

File	Driverlib Function
QCTMRLAT	
eqep.h	EQEP_getCaptureTimerLatch
QCPRDLAT	
eqep.h	EQEP_getCapturePeriodLatch
REV	
-	
QEPSTROBESEL	
eqep.h	EQEP_setStrobeSource
QMACTRL	
eqep.h	EQEP_setQMAModuleMode
QEPSRCSEL	
eqep.h	EQEP_selectSource

Chapter 28

Sigma Delta Filter Module (SDFM)

The sigma delta filter module (SDFM) is a four-channel digital filter designed specifically for current measurement and resolver position decoding in motor control applications. Each input channel can receive an independent delta-sigma (ΔΣ) modulator bit stream. The bit streams are processed by four individually-programmable digital decimation filters. The filter set includes a fast comparator (secondary filter) for immediate digital threshold comparisons for over-current and under-current monitoring, and zeros crossing detection.

28.1 Introduction

Figure 28-1 shows the SDFM CPU Interface.

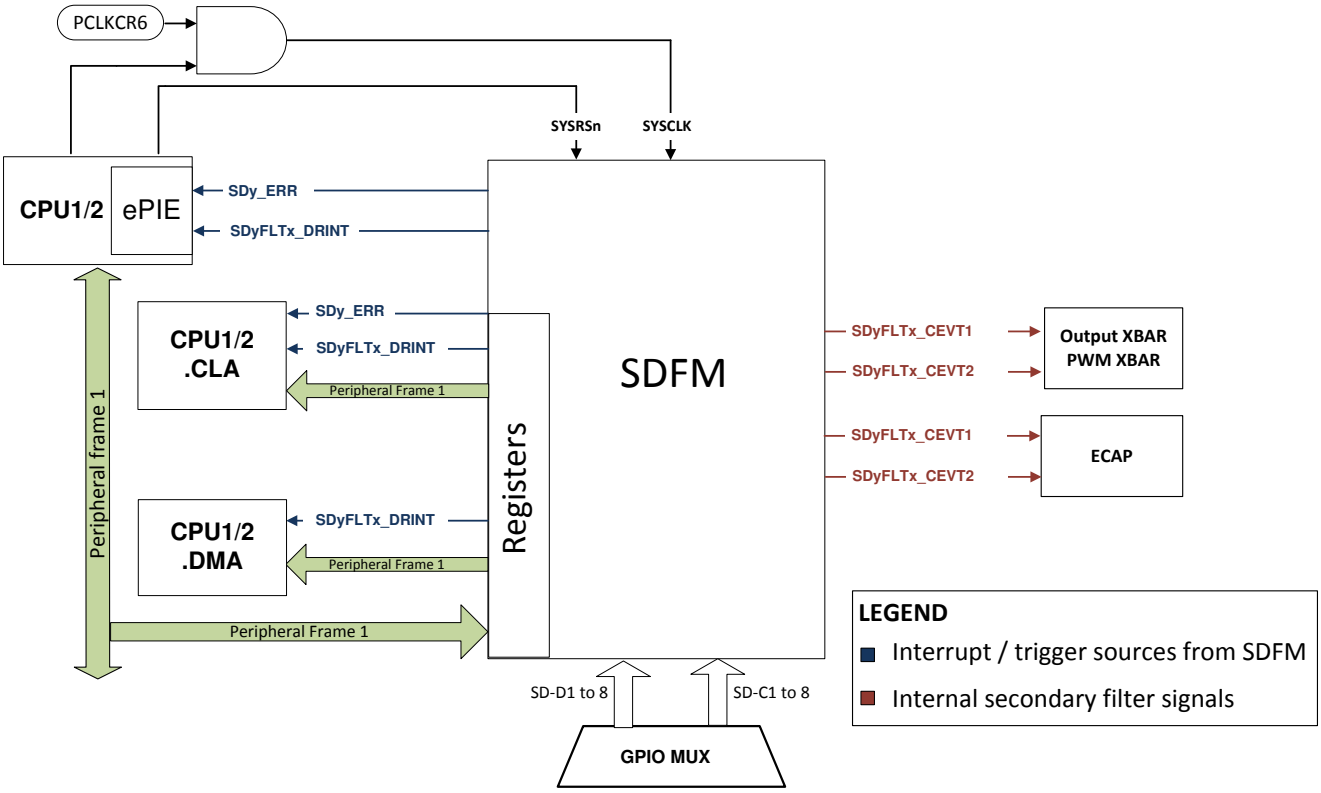


Figure 28-1. Sigma Delta Filter Module (SDFM) CPU Interface

28.1.1 SDFM Related Collateral

Foundational Materials

- C2000 Academy - SDFM
- *How delta-sigma ADCs work, Part 1 Application Report*
- *How delta-sigma ADCs work, Part 2 Application Report*
- Nuts and Bolts of the Delta-Sigma Converter (Video)
- Sigma Delta Filter Module (SDFM) Training for C2000™ MCUs (Video)

Getting Started Materials

- *Achieving Better Signal Integrity With Isolated Delta-Sigma Modulators in Motor Drives Application Report*
  - Critical information for a hardware designer
- Using Sigma Delta Filter Module (SDFM) to Measure the Analog Input Signal
  - NOTE: This is a non-TI (third party) site.

Expert Materials

- C2000 DesignDRIVE Development Kit for Industrial Motor Control
- *Diagnosing Delta-Sigma Modulator Bitstream Using C2000™ Configurable Logic Block Application Report*
- *Isolated Current Shunt and Voltage Measurement Kit Application Report*
- Isolated, Shunt-Based Current Sensing Reference Design
- *Quick Response Control of PMSM Using Fast Current Loop Application Report*
- The case for isolated delta-sigma modulators: Is my system fast enough for short-circuit detection?
- Vienna Rectifier-Based Three Phase Power Factor Correction Reference Design Using C2000 MCU

28.1.2 Features

SDFM features include:

- Eight external pins per SDFM module
  - Four sigma-delta data input pins per SDFM module (SD-Dx, where x = 1 to 4)
  - Four sigma-delta clock input pins per SDFM module (SD-Cx, where x = 1 to 4)
- Different configurable modulator clock modes supported:
  - Mode 0: Modulator clock rate equals the modulator data rate.
- Four independent, configurable secondary filter (comparator) units per SDFM module:
  - Four different filter type selection (Sinc1/Sinc2/SincFast/Sinc3) options available
  - Ability to detect over-value condition, under-value condition, and Threshold-crossing conditions
    1. Two independent Higher Threshold comparators (used to detect over-value condition)
    2. Two independent Lower Threshold comparators (used to detect under-value condition)
    3. One independent Threshold-Crossing comparator (used to measure duty cycle/frequency with eCAP)
  - OSR value for comparator filter unit (COSR) programmable from 1 to 32
- Four independent configurable primary filter (data filter) units per SDFM module:
  - Four different filter type selection (Sinc1/Sinc2/SincFast/Sinc3) options available
  - OSR value for data filter unit (DOSR) programmable from 1 to 256
  - Ability to enable or disable (or both) individual filter module
  - Ability to synchronize all four independent filters of an SDFM module by using the Master Filter Enable (MFE) bit or by using PWM signals
- Data filter output can be represented in either 16 bits or 32 bits.
- Data filter unit has a programmable mode FIFO to reduce interrupt overhead. The FIFO has the following features:
  - The primary filter (data filter) has a 16-deep x 32-bit FIFO.
  - The FIFO can interrupt the CPU after programmable number of data-ready events.
  - FIFO Wait-for-Sync feature: Ability to ignore data-ready events until the PWM synchronization signal (SDSYNC) is received. Once the SDSYNC event is received, the FIFO is populated on every data-ready event.
  - Data filter output can be represented in either 16 bits or 32 bits.
- PWMx.SOCA/SOCB can be configured to serve as SDSYNC source on a per-data-filter-channel basis.
- PWMs can be used to generate a modulator clock for sigma-delta modulators.
- Configurable Input Qualification available for both SD-Cx and SD-Dx
- Ability to use one filter channel clock (SD-C1) to provide clock to other filter clock channels.
- Configurable digital filter available on comparator filter events to blankout comparator events caused by spurious noise

28.1.3 Block Diagram

Each SDFM module has four independent filter modules. These filter modules are identical and can be configured independently. Each individual filter module has the following units:

- Input control unit
- Primary filter (data filter) unit
- Secondary filter (comparator filter) unit with 4 independent comparators

Figure 28-2 shows the SDFM module block diagram. The SDFM port operation is configured and controlled by the registers listed in Section 28.12.

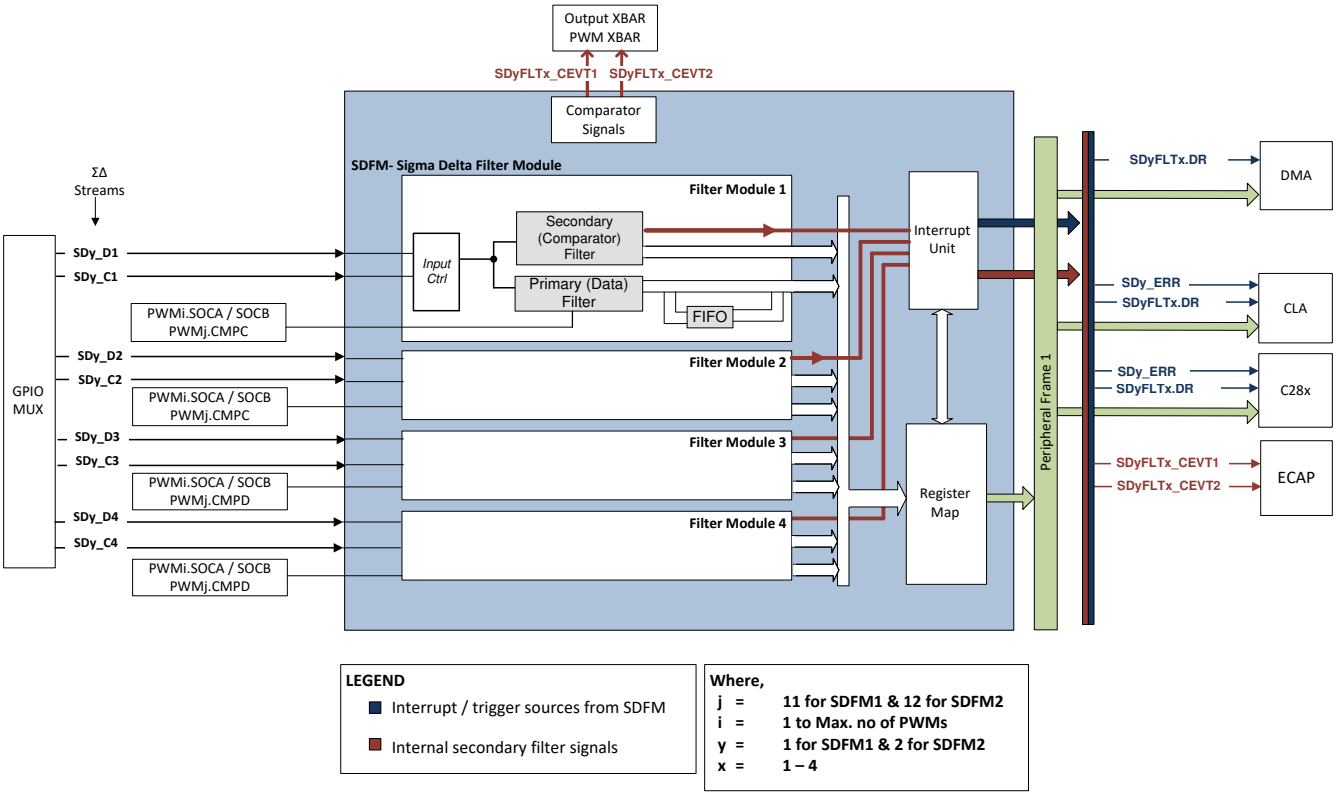


Figure 28-2. Sigma Delta Filter Module (SDFM) Block Diagram

Each filter module shown in Figure 28-3 has a primary (data) filter and a secondary (comparator) filter pair that receives the same bit stream. Except for the input bit stream, both the primary and secondary filter are completely independent of each other. Each of these filter modules can be independently configured. So, in a SDFM module, there is a total of four primary filters and four secondary filters.

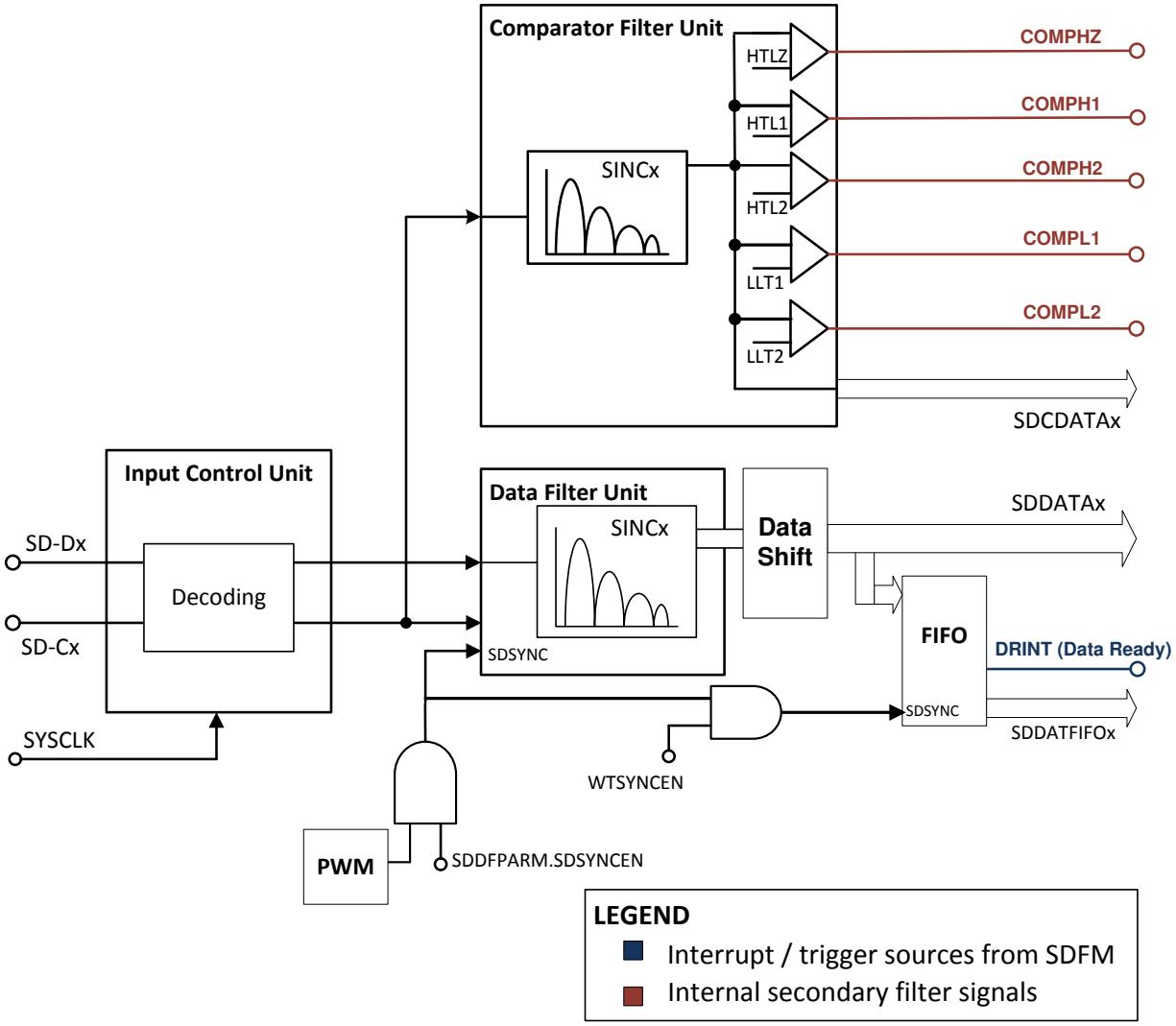


Figure 28-3. Block Diagram of One Filter Module

28.2 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

For proper SDFM operation, use the following GPIO input qualification. Other GPIO qualifications are not supported.

- GPIO Input qualification is ASYNC, make sure to check the SDFM Electrical Data and Timing (Using ASYNC) requirement is met and be aware of the following caution message. SDFM Input Qualification feature is used to provide protection against random noise glitches.

CAUTION

The SDFM clock inputs (SDx_Cy pins) directly clock the SDFM module. Any glitches or ringing noise on these inputs can corrupt the SDFM module operation. Special precautions must be taken on these signals to make sure of a clean and noise-free signal that meets SDFM timing requirements. Precautions such as series termination for ringing due to any impedance mismatch of the clock driver and spacing of traces from other noisy signals are recommended.

Note

The SDFM module expects SD-Dx to change on the falling edge of SD-Cx and strobes for SD-Dx on the rising edge. But some SD-modulators in the market change SD-Dx on the rising edge and expect SDFM to strobe for data on the falling edge. In such cases, the GPIO inversion feature (GPxINV) is used on SD-Cx pin to change polarity and make it compatible with the SDFM.

See the *General-Purpose Input/Output (GPIO)* chapter for more details on GPIO mux and settings.

28.3 Input Qualification

Impulse noise sources such as EMI, crosstalk, and so on, has the possibility of corrupting SDCLK and SDDATA bit streams, resulting in corrupted SDFM filtered data. This impulse noise effects can be mitigated when using the input qualification feature that synchronizes SDCLK/SDDATA signals with PLLCLK. By default, both SDCLK and SDDATA bit stream are not synchronized. SDCLK can be synchronized to PLLCLK by setting SDCTLPARMx.SDCLKSYNC = 1 and SDDATA can be synchronized to PLLCLK by setting SDCTLPARMx.SDDATASYNC = 1. Figure 28-4 shows optional Input Qualification option on SDCLK and SDDATA lines.

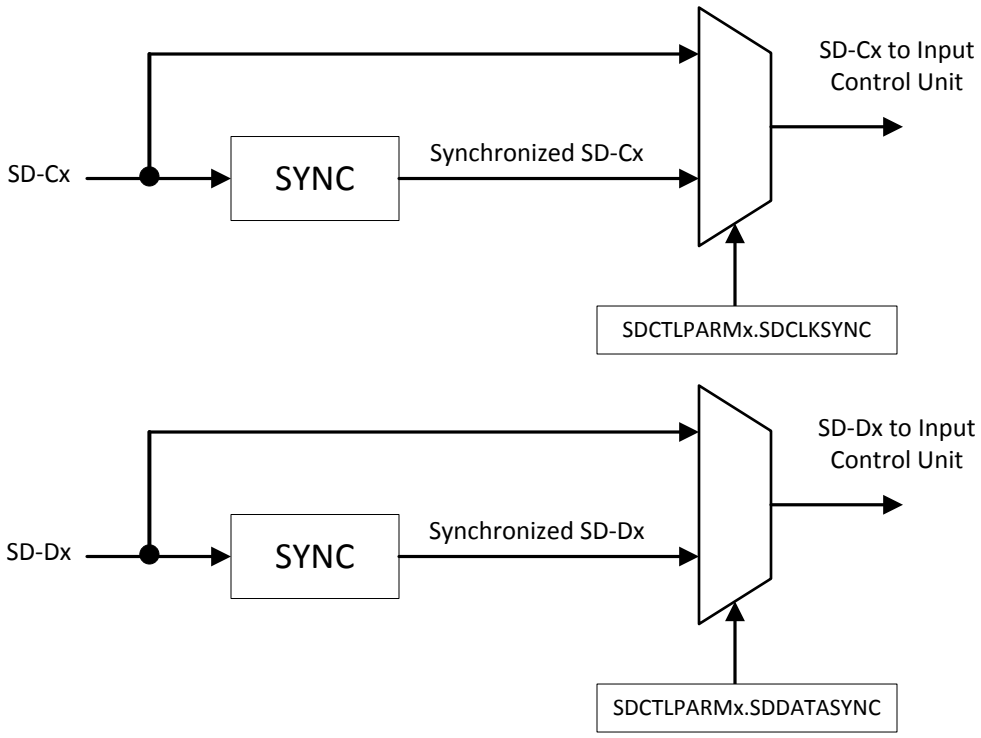


Figure 28-4. Input Qualification on SD-Cx and SD-Dx

28.4 Input Control Unit

The input control unit receives sigma delta modulated data and a sigma delta modulated clock. The modulated data received is captured and passed on to the data filter unit and comparator unit. This unit can be configured to receive the modulated data in Mode 0. Table 28-1 and Figure 28-5 show how SDCTLPARMx.MOD bits can be configured in Mode 0.

Table 28-1. Modulator Clock Modes

Modulator Mode [MOD]	Description
0	The modulator clock is running with the modulator data rate. The modulator data is strobed at every rising edge of the modulator clock.
1	Reserved
2	Reserved
3	Reserved

Note

To achieve the maximum value, the sigma-delta modulator has to be operated at absolute maximum positive or negative full scale, which is outside of the recommended full scale range of 80% of most sigma-delta modulators.

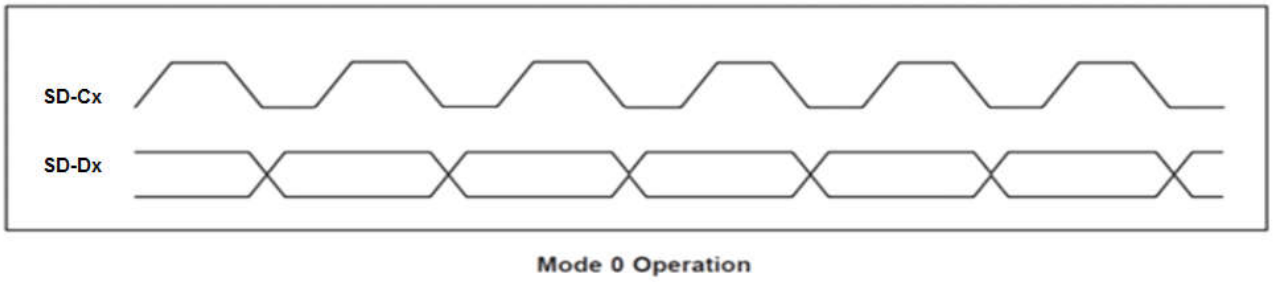


Figure 28-5. Different Modulator Modes Supported

28.5 SDFM Clock Control

In systems, the modulator clock can be generated using PWMs. Assuming all the SD-CLKs see the same delay on board traces, you can potentially use just one clock to clock multiple filters; thereby, saving on the number of pins used for SDFM. To enable this, Filter1 SDCLK (SD-C1) can possibly apply to other filter channels if required. The SDCTLPARAMx.SDCLKSEL register bit field can be configured to select filter channel SDCLK. See Figure 28-6 to view this feature.

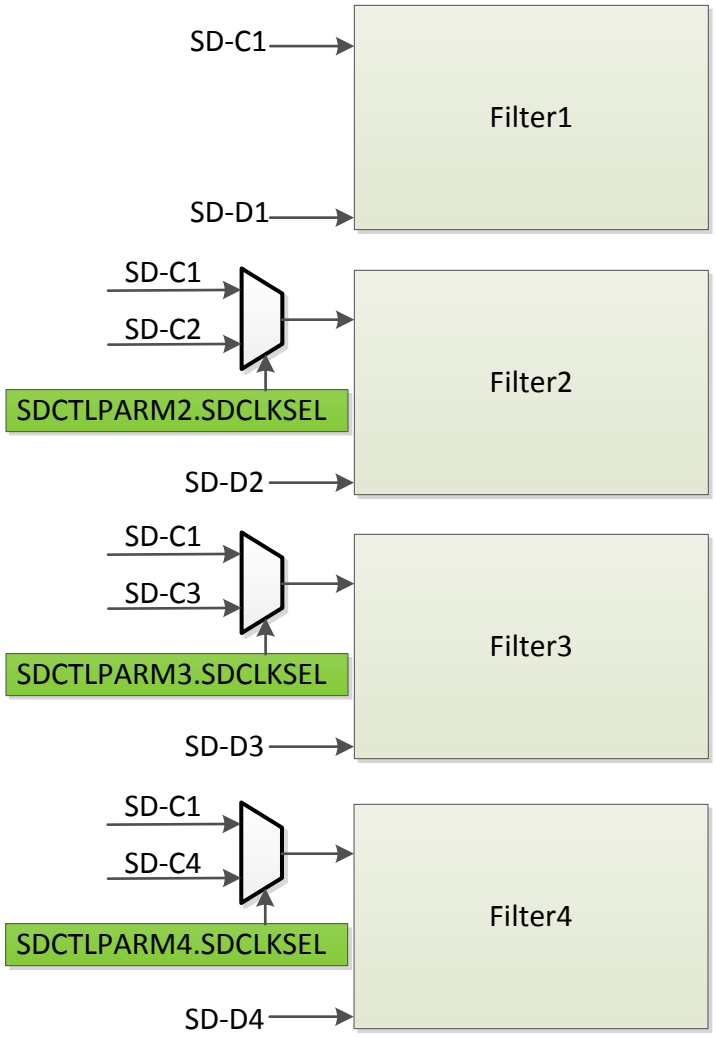


Figure 28-6. SDFM Clock Control

28.6 Sinc Filter

Both the comparator filter and data filter available in SDFM have the Sinc^N filter as the core. The Sinc^N filter is essentially a low-pass filter that converts the input bit stream into digital data by digital filtering and decimation. This filtered digital data represents analog input given to the sigma delta modulator. Simplified Sinc^N architecture consists of cascaded integrators and differentiators separated by a down-sampler as shown in Figure 28-7. The Z-transfer function of the Sinc filter of order N is shown in Figure 28-8.

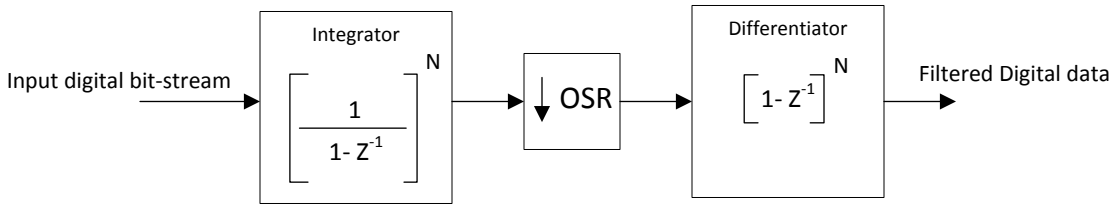


Figure 28-7. Simplified Sinc Filter Architecture

$$H(Z) = \left[\frac{1 - Z^{-OSR}}{1 - Z^{-1}} \right]^{N}$$

N = Order of Sinc filter
OSR = Over Sampling Ratio

Figure 28-8. Z-Transform of Sinc Filter of Order N

Effective resolution of the Sinc filter (ENOB) depends upon filter type, OSR and sigma-delta modulator frequency. Typically, higher resolution or ENOB can be achieved by higher OSR for a given filter type; however, the tradeoff is increased filter delay. It is important to choose the right sigma delta modulator by studying the optimal speed versus resolution tradeoff. Refer to the corresponding sigma delta modulator data sheet to determine the effective resolution for a given Sinc filter configuration. Figure 28-9 shows the frequency response of different filter structures when OSR = 32 and when the sigma delta modulator frequency is 10 MHz.

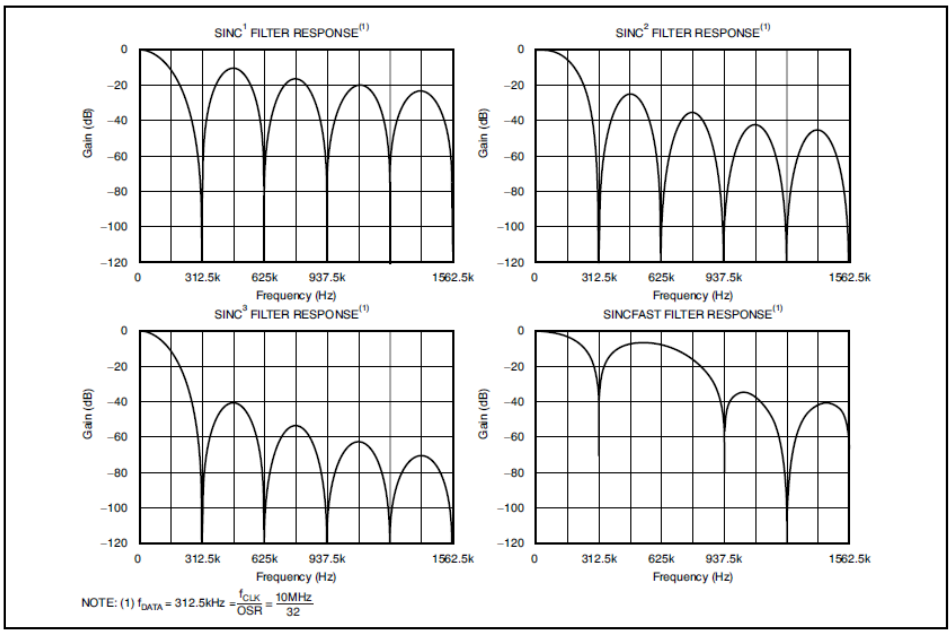


Figure 28-9. Frequency Response of Different Sinc Filters

The order of different sinc filter is shown in Table 28-2.

Table 28-2. Order of Sinc Filter

Filter Type	Order of Sinc Filter
Sinc1	1
Sinc2	2
Sinc3	3
SincFast	3

28.6.1 Data Rate and Latency of the Sinc Filter

The data rate of the sinc filter (filter throughput) represented in samples/sec is calculated by the following formula:

$$Data\ rate\ of\ Sinc\ filter = \frac{Modulator\ data\ rate}{OSR} \quad (15)$$

The latency of the sinc filter represented in secs is defined as the amount of time taken by a sinc filter type to deliver the correct filtered output upon initiation. For a given filter type, latency is calculated by the following formula:

$$Latency\ of\ Sinc\ filter = \frac{Order\ of\ Sinc\ filter}{Data\ rate\ of\ Sinc\ filter} \quad (16)$$

Example configuration:

Sinc filter type	= sinc3
Modulator data rate	= 10 MHz
OSR	= 256
Data rate of Sinc Filter	= 10 MHz / 256 = 39.1 K samples / sec
Sinc filter latency	= 76.8 µs
Sinc filter type	= sinc2
Modulator data rate	= 10 MHz
OSR	= 256
Data rate of Sinc Filter	= 10 MHz / 256 = 39.1 K samples / sec
Sinc filter latency	=51.2 µs

28.7 Data (Primary) Filter Unit

The data filter is a configurable Sinc filter which supports the following filter types: Sinc1, Sinc2, Sinc3, and SincFast. The data filter OSR (DOSR) settings can be configured from 1 to 256 and is independent of the comparator filter. Effective resolution of the data filter (ENOB) depends upon Data filter type, DOSR, and sigma-delta modulator frequency. By default, the data filter is disabled and setting of SDDFPARMx.FEN = 1 enables the data filter. The data filter output is represented in 26-bit signed integer in two's complement format. This filter unit translates a low input signal as '-1' and a high input signal as '1'. The resulting calculation gives both positive and negative values for the output of the data filter. Table 28-3 shows the different full scale values that the data filter can store using different OSRs.

See Section 28.6.1 to understand how to calculate data rate and latency of data filter.

Table 28-3. Peak Data Values for Different DOSR/Filter Combinations

DOSR	Sinc1	Sinc2	Sinc3	SincFast
x	x	x^2	x^3	$2x^2$
4	–4 to 4	–16 to 16	–64 to 64	–32 to 32
8	–8 to 8	–64 to 64	–512 to 512	–128 to 128
16	–16 to 16	–256 to 256	–4096 to 4096	–512 to 512
32	–32 to 32	–1024 to 1024	–32,768 to 32,768	–2048 to 2048
64	–64 to 64	–4096 to 4096	–262,144 to 262,144	–8192 to 8192
128	–128 to 128	–16,384 to 16,384	–2,097,152 to 2,097,152	–32,768 to 32,768
256	–256 to 256	–65,536 to 65,536	–16,777,216 to 16,777,216	–131,072 to 131,072

28.7.1 32-bit or 16-bit Data Filter Output Representation

The data filter output can be represented in either 32-bit or 16-bit format.

32-bit data filter representation:

- When SDDPARMx.DR = 1, data filter output is represented in 32-bit format. Writes to shift control bits do not have any bearing on the output of the data filter in this configuration.

16-bit data filter representation:

- By default, data filter output is represented in 16-bit format
- When SDDPARMx.DR = 0, data filter output is represented in 16-bit format. But it is the responsibility of the user to configure the corresponding shift control bits in the SDDPARMx register to control which 16-bit part of the 32-bit word is sent to the register map.

  For example, for the data filter configuration below:
  - Filter type = Sinc3
  - OSR = 128
  - SDDPARMx.DR = 0

  The data filter with a 26-bit signed output value can be in the range of –16,777,216 to 16,777,216. But, 16-bit signed output supports values only from –32,768 to 32,767. Therefore, it is required to configure shift control bits (SDDPARMx.SH) to 7 to represent the data filter output correctly in 16-bit format. Table 28-4 shows the configuration settings of shift control bits for different OSR and filter types.

Table 28-4. Shift Control Bit Configuration Settings

OSR	Sinc1	Sinc2	SincFast	Sinc3
1 to 31	0	0	0	0
32 to 40	0	0	0	1
41 to 50	0	0	0	2
51 to 63	0	0	0	3
64 to 80	0	0	0	4
81 to 101	0	0	0	5
102 to 127	0	0	0	6
128 to 161	0	0	1	7
162 to 181	0	0	1	8
182 to 203	0	1	2	8
204 to 255	0	1	2	9
256	0	2	3	10

CAUTION

Configuring shift control bits incorrectly results in getting an incorrect 16-bit data filter output.

28.7.2 Data FIFO

Each primary (data) filter channel has a 16-level deep, 32-bit FIFO.

FIFOs can be configured to collect a programmable number of data filter samples before issuing data-ready interrupt. This reduces the number of data-ready interrupts generated and resulting interrupt overhead for managed data flow.

By default, FIFO operation is disabled. FIFOs can be enabled by setting SDFIFOCTLx.FFEN = 1. When FIFO is enabled, each data-ready event from the data filter populates the FIFO, and the status of the FIFO at any given time is updated in the SDFIFOCTLx.SDFFST bit field.

Setting up FIFO to interrupt after receiving programmable number of data ready events:

- Enable SDFM FIFO (Set SDFIFOCTLx.FFEN = 1)
- Enable SDFM FIFO interrupt (Set SDFIFOCTLx.FFIEN = 1)
- Configure SDFIFOCTLx.SDFFIL bit field to any value between 0 to 16
- Configure SDFM data ready event to interrupt on FIFO interrupt (SDFFINT) (Set SDFFINTx = 1)
- Select data-ready interrupt source is SDFFINTx (DRINTx = SDFFINTx) (SDFIFOCTLx.DRINTSEL = 1)

When the SDFIFOCTLx.SDFFST >= SDFIFOCTLx.SDFFIL condition is met, the SDIFLG.SDFFINTx bit is set and an interrupt is generated on the DRINTx. SDIFLG.SDFFINTx flag can be cleared by setting the SDIFLGCLR. SDFFINTx bit field.

Wait for Sync feature:

The FIFO wait for sync feature can be used to ignore data-ready events from the data filter until the SDSYNC (from PWM) event is triggered.

By default, the Wait for Sync feature is disabled. This feature can be enabled by setting SDSYNCx.WTSYNCEN = 1

When the wait for sync feature is disabled:

FIFOs get populated on every data ready event until the FIFO gets full (or) when SDFIFOCTLx.SDFFST >= SDFIFOCTLx.SDFFIL.

When the wait for sync feature enabled:

FIFOs do not get populated on every data ready event until the FIFO receives a SDSYNC event. On a SYSYNC event, the FIFO sets SDSYNCx.WTSYNFLG = 1 and data ready events from the primary filter start populating the FIFO until either the FIFOs get full or when SDFIFOCTLx.SDFFST >= SDFIFOCTLx.SDFFIL. WTSYNFLG can be cleared either automatically or manually.

When WTSYNFLG = 0, FIFOs contents are frozen and subsequent data ready events do not populate FIFO until next SDSYNC event.

WTSYNFLG automatic clear mode:

By default, this mode is enabled. When SDSYNCx.WTSCLREN = 1, WTSYNFLG is automatically cleared on SDFFINT event.

WTSYNFLG manual clear mode:

Setting SDSYNCx.WTSYNCLR = 1 can be used to clear WTSYNFLG manually.

Clearing FIFO contents:

FIFO contents can cleared by any of the following methods:-

- Disabling FIFO clear FIFO contents. This can be done by clearing SDFIFOCTLx.FFEN = 0.
- Disabling Primary filter clear FIFO contents. This can done by either clearing SDDFPARMx.FEN = 0 (or) by clearing SDMFILEN.MFE = 0.
- FIFO contents can also be automatically cleared upon receiving the SDSYNC event. By default, this feature is disabled and this feature can be enabled by setting FIFO Clear-on-SDSYNC enable (SDSYNCx.FFSYNCCLREN = 1).

  Note: The above feature is only enabled when wait for sync feature is enabled (SDSYNCx. WTSYNCEN = 1).

FIFO debug access behavior:

Debug access of the SDDATFIFOx registers does not affect the FIFO pointers. On a CPU/RTDMA access to the SDDATFIFOx register, the FIFO read pointers advance to the next available entry in the FIFO.

28.7.3 SDSYNC Event

Primary (data) filters can be synchronized with respect to the PWM event (called SDSYNC event). The SDSYNC signal from the PWM module is used to reset the DOSR counter. This feature is by default disabled and can be enabled by setting SDDFPARMx.SDSYNCEN = 1. Each primary filter can be synchronized from any of the available PWMx SOCA/SOCB signals (see Table 28-5). Additionally, PWM11.CMPC/CMPD can be used to reset SDFM1 filter modules and PWM12.CMPC/CMPD can be used to reset SDFM2 filter modules. The SDSYNCx.SDSYNCSEL bits allow the user to configure which PWM signal provides the SDSYNC pulse to the primary filter. Figure 28-10 shows how device PWM signals are connected to the SDFM modules.

Table 28-5. SDSYNCx.SYNCSEL

SDSYNCx.SYNCSEL	Sync Source
0	EPWM1_SOCA
1	EPWM1_SOCA
2-3	Reserved
4	EPWM2_SOCA
5	EPWM2_SOCA
6-7	Reserved
8	EPWM3_SOCA
9	EPWM3_SOCA
10-11	Reserved
12	EPWM4_SOCA
13	EPWM4_SOCA
14-15	Reserved
16	EPWM5_SOCA
17	EPWM5_SOCA
18-19	Reserved
20	EPWM6_SOCA
21	EPWM6_SOCA
22-23	Reserved
24	EPWM7_SOCA
25	EPWM7_SOCA
26-27	Reserved
28	EPWM8_SOCA
29	EPWM8_SOCA
30-31	Reserved
32	EPWM9_SOCA
33	EPWM9_SOCA
34-35	Reserved
36	EPWM10_SOCA
37	EPWM10_SOCA
38-39	Reserved
40	EPWM11_SOCA
41	EPWM11_SOCA
42-43	Reserved
44	EPWM12_SOCA
45	EPWM12_SOCA
46-47	Reserved
48	EPWM13_SOCA

Table 28-5. SDSYNCx.SYNCSEL (continued)

SDSYNCx.SYNCSEL	Sync Source
49	EPWM13_SOCA
50-51	Reserved
52	EPWM14_SOCA
53	EPWM14_SOCA
54-55	Reserved
56	EPWM15_SOCA
57	EPWM15_SOCA
58-59	Reserved
60	EPWM16_SOCA
61	EPWM16_SOCB
62	Reserved
63	SD1SYNC1-EPWM11_CMPC SD1SYNC2-EPWM11_CMPC SD1SYNC3-EPWM11_CMPD SD1SYNC4-EPWM11_CMPD SD2SYNC1-EPWM12_CMPC SD2SYNC2-EPWM12_CMPC SD2SYNC3-EPWM12_CMPD SD2SYNC4-EPWM12_CMPD

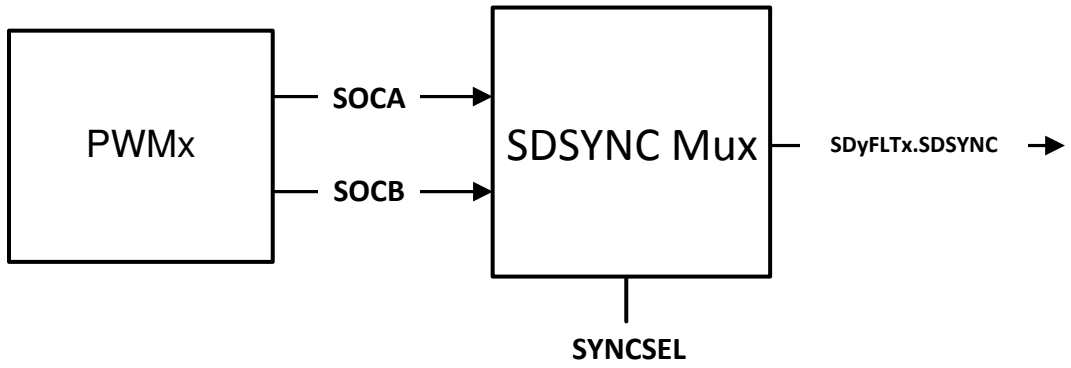


Figure 28-10. SDSYNC Event

Note

Make sure that only one SDSYNC event is generated per PWM timer period. Using PWM in up-count or down-count mode can automatically make sure that only SDSYNC events are generated. But, if up-down count mode is used, then make sure that only one SDSYNC event per PWM cycle is generated; otherwise, the filter synchronizer corrupts SDFM timing by providing two pulses per PWM cycle.

Because of the inherent architecture of the Sinc filter (Sinc1, Sinc2, Sinc3, SincFast), the first few samples, depending upon filter type, are incorrect. Table 28-6 shows the number of incorrect samples on the following conditions:

- When Sinc filter is enabled and configured for first time.
- When Sinc filter is disabled and re-enabled or reconfigured in the middle of operation.
- When data filter receives SDSYNC event from PWM.

Table 28-6. Number of Incorrect Samples Tabulated

Filter Type	Number of Incorrect Samples After the Filter is Enabled and Configured
Sinc1	No incorrect sample.
Sinc2	The first sample of the Sinc2 filter is incorrect.

Table 28-6. Number of Incorrect Samples Tabulated (continued)

Filter Type	Number of Incorrect Samples After the Filter is Enabled and Configured
SincFast	The first two samples of the SincFast filter are incorrect.
Sinc3	The first two samples of the Sinc3 filter are incorrect.

CAUTION

SDFM comparator interrupts can be enabled only after providing sufficient settling time to make sure the comparator filter does not trip on these incorrect samples. Therefore, SDFM comparator interrupts (CEVT1 and CEVT2) can be enabled only after a sufficient delay is provided after the comparator filter is configured. This sufficient delay is calculated by adding the latency of the comparator filter and 5 SD-Cx clock cycles.

28.8 Comparator (Secondary) Filter Unit

Most control systems require protection of the system by tripping the PWM in case the current or voltage goes out of bounds. The primary purpose of the secondary (comparator) filter is to allow the user to monitor input conditions with a fast settling time. This allows the user to trip PWMs to protect the system from potential damage.

Note

The secondary (comparator) filter cannot be synchronized with respect to the PWM event (SDSYNC event).

The comparator filter is a configurable Sinc filter that supports the following filter types: Sinc1, Sinc2, Sinc3, and SincFast. The comparator OSR (COSR) settings can be configured from 1 to 32 and is independent of the data filter. Effective resolution of the comparator filter (ENOB) depends upon the comparator filter type, COSR, and sigma-delta modulator frequency. By default, the comparator filter is disabled and setting SDCPARMx.CEN = 1 enables the comparator filter. The comparator filter output is represented in 16-bit unsigned format. This filter unit translates a low input signal as 0 and a high input signal as 1. The resulting calculations give only positive values for the output of the comparator filter. Table 28-7 shows the different full-scale values that the comparator filter can store using different OSRs.

Table 28-7. Peak Data Values for Different OSR/Filter Combinations

OSR	Sinc1	Sinc2	Sinc3	SincFast
x	0 to x	0 to x2	0 to x3	0 to 2x2
4	0 to 4	0 to 16	0 to 64	0 to 32
8	0 to 8	0 to 64	0 to 512	0 to 128
16	0 to 16	0 to 256	0 to 4096	0 to 512
32	0 to 32	0 to 1024	0 to 32,768	0 to 2048

See Section 28.6.1 to understand how to calculate data rate and latency of comparator filter.

The output of the comparator filter is memory-mapped and can be read in the SDCDATAx register. This register, SDCDATAx, is updated every COSR number of SD-Cx cycles. The comparator filter digital output is connected to digital comparators explained below.

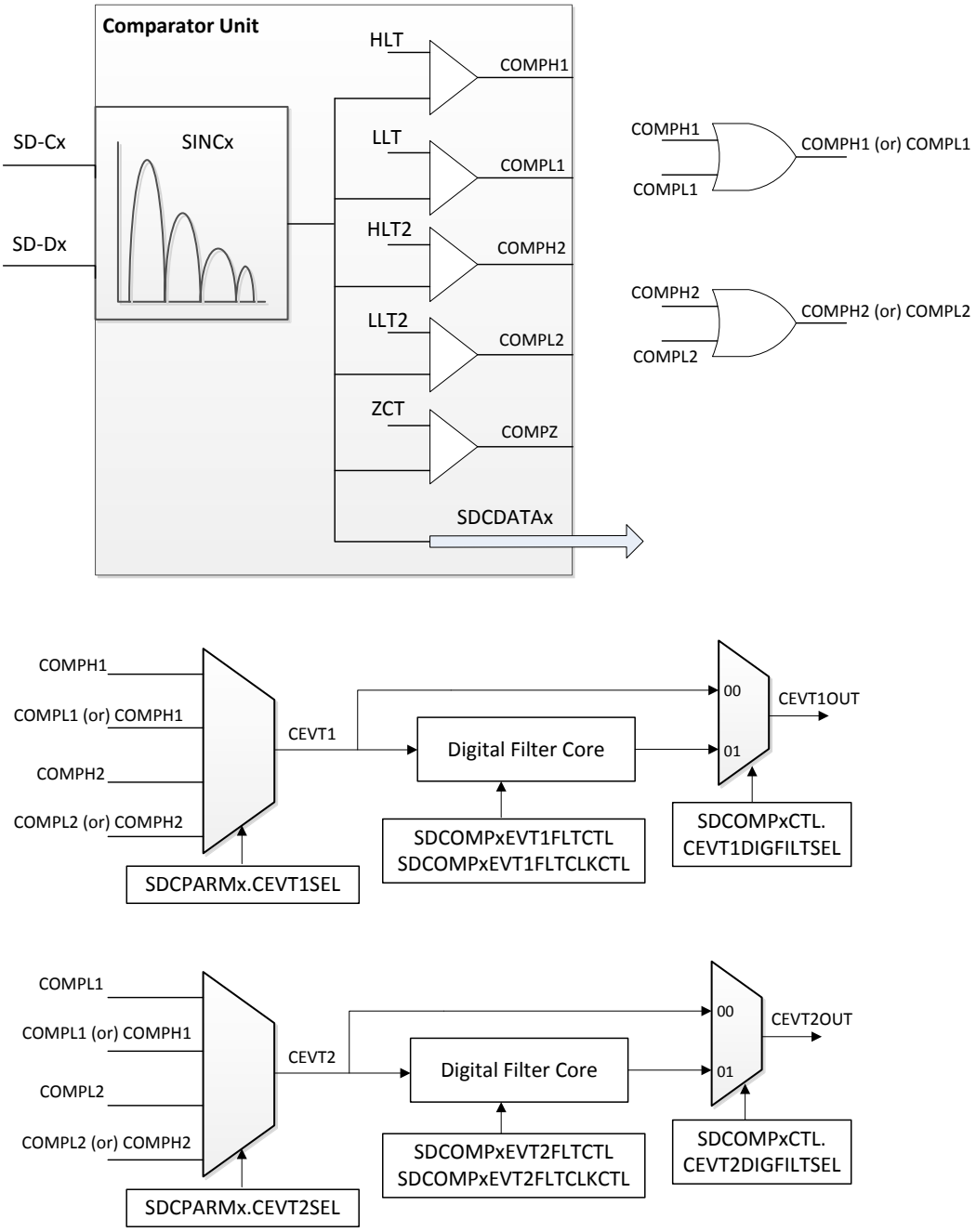


Figure 28-11. Comparator Unit Structure

28.8.1 Higher Threshold (HLT) Comparators

- High threshold comparator can be used to detect over-value condition.
- When comparator data > = higher threshold register, a high threshold event is generated.
- Higher threshold comparator events except for COMPHZx can be configured to trigger following events: CPU interrupt, CLA task, PWM trip.
- This device has three High Threshold comparators:
  - **Higher Threshold 1 (HLT1) Comparator**:
    - When comparator data > = (SDFLTxCMPH1.HLT), HLT1 comparator generates COMPH1 event.
    - The COMPH1 event is connected to both CEVT1 and CEVT2.
  - **Higher Threshold 2 (HLT2) Comparator**:
    - When comparator data > = (SDFLTxCMPH2.HLT), HLT2 comparator generates COMPH2 event.
    - The COMPH2 event is connected to both CEVT1 and CEVT2.
  - **Higher Threshold (HTLZ) Comparator**:
    - When comparator data > = (SDFLT1CMPHZ.CMPHZ), it can generate a Higher Threshold (B) event (COMPHZx) and sets the corresponding SDSTATUS.HZx flag. But, this event cannot be configured to generate SDFM interrupt (SDx_ERR). The COMPHZ signals from HTLZ comparator are connected to CLB XBAR.

28.8.2 Lower Threshold (LLT) Comparators

- The low threshold comparator can be used to detect under-value condition.
- When comparator data < = Lower Threshold register, a low threshold event is generated.
- Lower threshold comparator events can be configured to trigger following events: CPU interrupt, CLA task, PWM trip.
- Lower threshold comparator events can be used in conjunction with ECAP to measure the frequency / duty cycle of Threshold crossing
- This device has two low threshold comparators. .
  - **Lower Threshold 1 (LLT1) Comparator**
    - When comparator data < = (SDFLTxCMPL1.LLT), the LLT1 comparator generates COMPL1 event.
    - The COMPL1 event is connected to both CEVT1 and CEVT2.
  - **Lower Threshold 2 (LLT2) Comparator**
    - When comparator data < = (SDFLTxCMPL2.LLT), LLT2 comparator generates COMPL2 event.
    - The COMPL2 event is connected to both CEVT1 and CEVT2.

28.8.3 Digital Filter

The digital filter works on a window of FIFO samples (SAMPWIN + 1) taken from the input. The filter output resolves to the majority value of the sample window, where majority is defined by the threshold (THRESH) value. If the majority threshold is not satisfied, the filter output remains unchanged.

For proper operation, the value of THRESH must be greater than SAMPWIN / 2.

A prescale function (CLKPRESCALE) determines the filter sampling rate, where the filter FIFO captures one sample every CLKPRESCALE system clocks. Old data from the FIFO is discarded.

A conceptual model of the digital filter is shown in Figure 28-12.

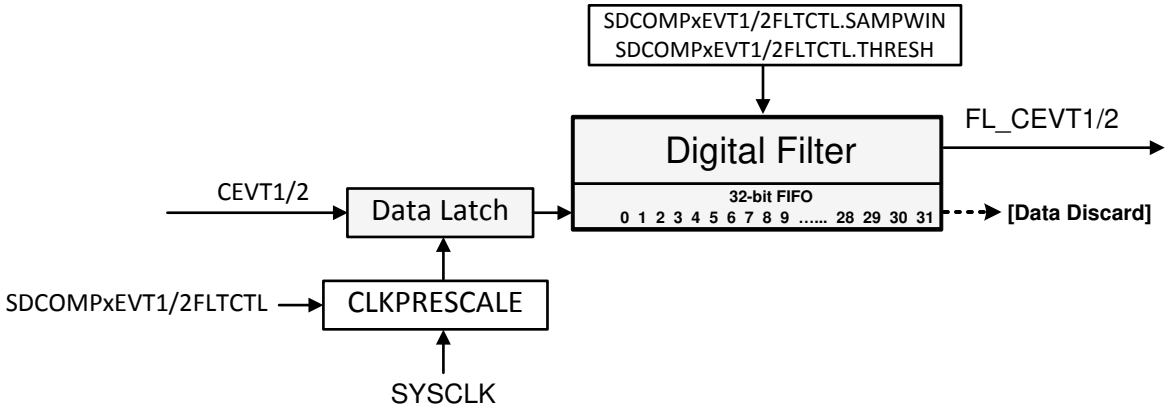


Figure 28-12. Digital Filter

Equivalent C code of the filter implementation is:

```
if (FILTER_OUTPUT == 0) {
    if (Num_1s_in_SAMPWIN >= THRESH) {
        FILTER_OUTPUT = 1;
    }
}
else {
    if (Num_0s_in_SAMPWIN >= THRESH) {
        FILTER_OUTPUT = 0;
    }
}
```

Filter Initialization Sequence

To ensure proper operation of the digital filter, the following initialization sequence is recommended:

1. Configure the digital filter parameters for operation:
   - Set SAMPWIN for the number of samples to monitor in the FIFO window.
   - Set THRESH for the threshold required for majority qualification.
   - Set CLKPRESCALE for the digital filter clock prescale value.
2. Initialize the sample values in the digital FIFO window by setting FILINIT = 1.

28.9 Theoretical SDFM Filter Output

The following equations can be used to derive a theoretical filter output of an SDFM filter output for both a comparator filter and a data filter.

$$\text{Density of ones in bitstream} = \frac{\text{Input Voltage} + \text{Vclipping}}{2 \times \text{Vclipping}} \tag{17}$$

Where:

- Vclipping = maximum differential voltage input range of modulator
- Input voltage = Differential input voltage applied to the modulator

$$\text{Comparator Filter Output (Theoretical)} = \text{Density of ones in bitstream} \times \text{Maximum Filter Output (FilterType, COSR)} \tag{18}$$

$$\text{FilterOutput} = \left\{\frac{\text{absolute(Input voltage)}}{\text{Vclipping}}\right\} \times \text{Maximum Filter Output (FilterType, DOSR)} \tag{19}$$

$$\text{Data Filter Output\_32bit(Theoretical)} = \begin{cases} \text{FilterOutput} & \text{if Input Voltage is +ve voltage} \\ \text{2's complement of FilterOutput} & \text{if input voltage is -ve voltage} \end{cases} \tag{20}$$

$$\text{Data Filter Output\_16bit(Theoretical)} = \text{Data Filter Output\_32bit(Theoretical)} >> \text{Shift value(FilterType, OSR)} \tag{21}$$

For example, when using the AMC1306x25 modulator:

AMC1306x25	Vclipping = Input voltage (AINP - AINN) =	320 mV 100 mV
SDFM filter settings	Filter type = Comparator OSR (COSR) = Data filter OSR (DOSR) =	3 32 100

Density of ones in bitstream	Using Equation 17	0.65625
Comparator filter output Filter type = Sinc3 COSR = 32	Using Equation 18	21504
Data filter output (32-bit) Filter type = Sinc3 DOSR = 100	Using Equation 19 and Equation 20	312500
Data filter output (32-bit) Filter type = Sinc3 DOSR = 100 (Right shift by 5)	Using Equation 21	9765

28.10 Interrupt Unit

Each SDFM can generate five CPU interrupts such as SDFM Error (SDy_ERR) and SDFM data ready (SDy_DRINT1 / SDy_DRINT2, SDy_DRINT3, SDy_DRINT4) interrupts for each filter module.

28.10.1 SDFM (SDyERR) Interrupt Sources

Figure 28-13 shows the structure of SDy_ERR interrupt. SDy_ERR interrupt can be triggered by any of these 16 events.

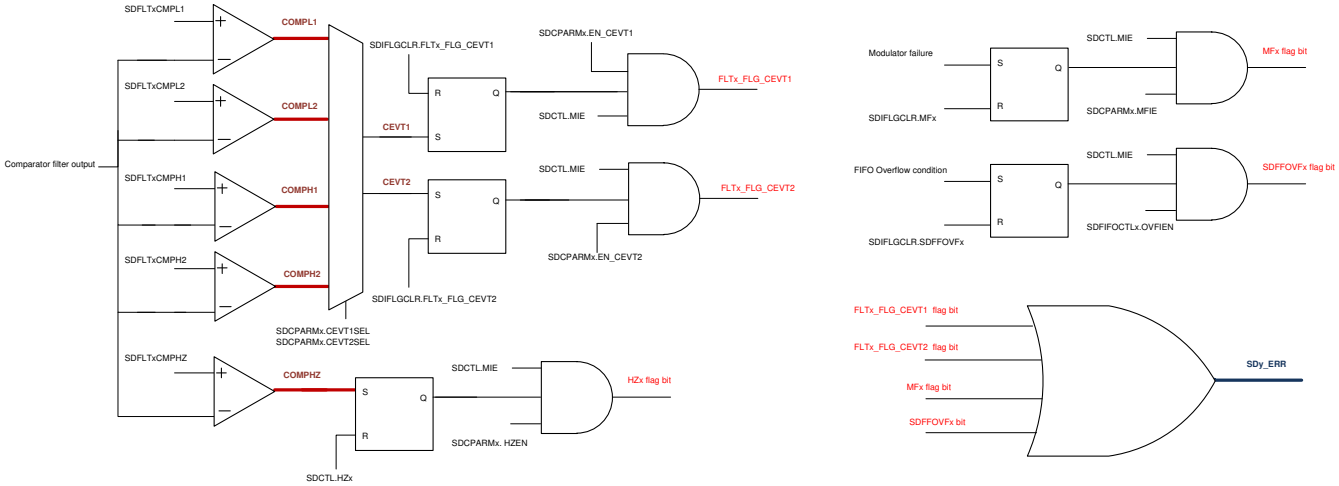


Figure 28-13. SDFM Error (SD_ERR) Interrupt Sources

1. Comparator Event1 (CEVT1)

CEVT1 events from any of the four comparator filter module can trigger CPU interrupt. This event can be configured to trigger SDy_ERR interrupt only if below configurations are made:

- Enable Main interrupt enable (SDCTL.MIE = 1)
- Enable comparator Event1 interrupt (SDCPARMx.EN_CEVT1 = 1)

On a CEVT1 event, SDIFLG.FLTx_FLG_CEVT1 flag bit is set. This flag bit can only be reset if the corresponding bit in SDIFLGCLR register is set and if the interrupt source is no longer active.

2. Comparator Event2 (CEVT2)

CEVT2 events from any of the four comparator filter module can trigger CPU interrupt. This event can be configured to trigger SDy_ERR interrupt only if below configurations are made:

- Enable Main interrupt enable (SDCTL.MIE = 1)
- Enable comparator event1 interrupt (SDCPARMx.EN_CEVT2 = 1)

On a CEVT2 event, SDIFLG.FLTx_FLG_CEVT2 flag bit is set. This flag bit can only be reset if the corresponding bit in SDIFLGCLR register is set and if the interrupt source is no longer active.

3. Modulator Failure (MFx) event

Modulator failures (MFx) are generated when SD-Cx goes missing. The modulator clock is considered missing if SD-Cx does not toggle for 64-SYSCLKs. MFx events from any of the four filter modules can trigger CPU interrupt. This event can be configured to trigger SDy_ERR interrupt only if below configurations are made:

- Enable Main Interrupt Enable (SDCTL.MIE = 1)
- Enable modulator clock failure interrupt source (SDCPARMx.MFIE = 1)

On a MFx event, SDIFLG.MFx flag bit is set. This flag bit can only be reset if the corresponding bit in SDIFLGCLR register is set and if the interrupt source is no longer active.

4. FIFO overflow (SDFFOVFx) event

The number of filter data available in FIFO at any given point can be tracked in SDFIFOCTLx.SDFFST. If the number of words received in FIFO is greater than Max FIFO depth (16), SDFFOVFx event is generated. SDFFOVFx events from any of the four filter modules can trigger CPU interrupt. This event can be configured to trigger SDy_ERR interrupt, only if below configurations are made:

- Enable SDFM FIFO (Set SDFIFOCTLx.FFEN = 1)
- Enable SDFM FIFO overflow interrupt (Set SDFIFOCTLx.OVFIEN = 1) and
- Enable Main interrupt enable (Set SDCTL.MIE = 1)

On a SDFFOVFx event, all subsequent data (primary) filter data is lost and is not stored in FIFO. SDIFLG.SDFFOVFx flag bit is set on a FIFO overflow event and this bit can be cleared if the corresponding bit in SDIFLGCLR register is set and if the interrupt source is no longer active.

28.10.2 Data Ready (DRINT) Interrupt Sources

Figure 28-14 shows the structure of interrupt SDy_DRINTx interrupt. Each SDy_DRINTx interrupt is triggered by corresponding Data Filter channel.

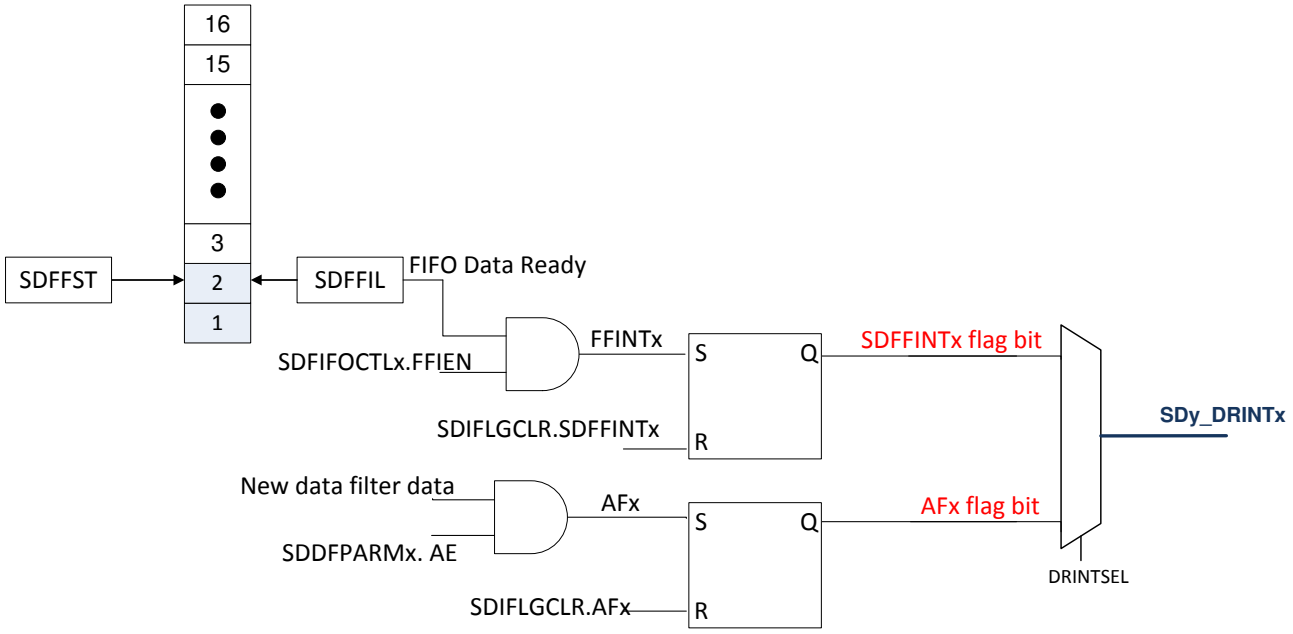


Figure 28-14. SDFM Data Ready (SDy_DRINTx) Interrupt

1. Data Acknowledge (AFx)

When the primary filter is ready with a new filter data, AFx event is generated. AFx events from each filter can generate their own SDy_DRINTx interrupt. This event can be configured to trigger SDy_DRINTx interrupt only if below configurations are made:

- Enable individual filter interrupts (SDDFPARMx. AE = 1)
- Select data-ready interrupt source AFx (DRINTx = AFx) (SDFIFOCTLx.DRINTSEL = 0)

On an AFx event, the SDIFLG.AFx flag bit is set. This flag bit can only be reset, if the corresponding bit in SDIFLGCLR register is set and if the interrupt source is no longer active.

2. Four FIFO Data ready interrupt (SDFFINTx)

FIFO Data Ready event is generated whenever SDFIFOCTLx.SDFFST >= SDFIFOCTLx.SDFFIL condition is met. FIFO data ready events from each filter can generate their own SDy_DRINTx interrupt. This event can be configured to trigger SDy_DRINTx interrupt only if below configurations are made:

Table 28-8 shows how the DRINTx output is selected.

- Enable SDFM FIFO (Set SDFIFOCTLx.FFEN = 1) and
- Enable SDFM FIFO interrupt (Set SDFIFOCTLx.FFIEN = 1)
- Select data-Ready interrupt source is SDFFINTx (DRINTx = SDFFINTx) (SDFIFOCTLx.DRINTSEL = 1)

Table 28-8. SDFM Data-Ready Interrupt (SDy_DRINTx) Output Selection

DRINTSEL	AE	FFIEN	FFEN	DRINTx
0	0	x	X	0
0	1	x	X	AFx
1	x	0	X	0
1	x	x	0	0
1	x	1	1	SDFFINTx

28.11 Software

28.11.1 SDFM Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/sdfm

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

28.11.1.1 SDFM Filter Sync CPU

FILE: sdfm_ex1_filter_sync_cpuread.c

In this example, SDFM filter data is read by CPU in SDFM ISR routine. The SDFM configuration is shown below:

- SDFM used in this example - SDFM1
- Input control mode selected - MODE0
- Comparator settings
  - Sinc3 filter selected
  - OSR = 32
  - HLT = 0x7FFF (Higher threshold setting)
  - LLT = 0x0000(Lower threshold setting)
- Data filter settings
  - All the 4 filter modules enabled
  - Sinc3 filter selected
  - OSR = 128
  - All the 4 filters are synchronized by using MFE (Master Filter enable bit)
  - Filter output represented in 16 bit format
  - In order to convert 25 bit Data filter into 16 bit format user needs to right shift by 7 bits for Sinc3 filter with OSR = 128
- Interrupt module settings for SDFM filter
  - All the 4 higher threshold comparator interrupts disabled
  - All the 4 lower threshold comparator interrupts disabled
  - All the 4 modulator failure interrupts disabled
  - All the 4 filter will generate interrupt when a new filter data is available.

External Connections

- SDFM_PIN_MUX_OPTION1 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO16-GPIO31
- SDFM_PIN_MUX_OPTION2 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO48-GPIO63
- SDFM_PIN_MUX_OPTION3 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO122-GPIO137

Watch Variables

- *filter1Result* - Output of filter 1
- *filter2Result* - Output of filter 2
- *filter3Result* - Output of filter 3
- *filter4Result* - Output of filter 4

28.11.1.2 SDFM Filter Sync CLA

FILE: sdfm_ex2_filter_sync_claread.c

In this example, SDFM filter data is read by CLA in Cla1Task1. The SDFM configuration is shown below:

- SDFM1 used in this example.For using SDFM2, few modifications would be needed in the example.
- MODE0 Input control mode selected
- Comparator settings
  - Sinc3 filter selected
  - OSR = 32

  - hlt = 0x7FFF (Higher threshold setting)
  - llt = 0x0000(Lower threshold setting)
- Data filter settings
  - All the 4 filter modules enabled
  - Sinc3 filter selected
  - OSR = 256
  - All the 4 filters are synchronized by using MFE (Master Filter enable bit)
  - Filter output represented in 16 bit format
  - In order to convert 25 bit Data filter into 16 bit format user needs to right shift by 10 bits for Sinc3 filter with OSR = 256
- Interrupt module settings for SDFM filter
  - All the 4 higher threshold comparator interrupts disabled
  - All the 4 lower threshold comparator interrupts disabled
  - All the 4 modulator failure interrupts disabled
  - All the 4 filter will generate interrupt when a new filter data is available

External Connections

- SDFM_PIN_MUX_OPTION1 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO16-GPIO31
- SDFM_PIN_MUX_OPTION2 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO48-GPIO63
- SDFM_PIN_MUX_OPTION3 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO122-GPIO137

Watch Variables

- *filter1Result* - Output of filter 1
- *filter2Result* - Output of filter 2
- *filter3Result* - Output of filter 3
- *filter4Result* - Output of filter 4

28.11.1.3 SDFM Filter Sync DMA

FILE: sdfm_ex3_filter_sync_dmaread.c

In this example, SDFM filter data is read by DMA. The SDFM configuration is shown below:

- SDFM1 used in this example. For using SDFM2, few modifications would be needed in the example.
- MODE0 Input control mode selected
- Comparator settings
  - Sinc3 filter selected
  - OSR = 32
  - hlt = 0x7FFF (Higher threshold setting)
  - llt = 0x0000(Lower threshold setting)
- Data filter settings
  - All the 4 filter modules enabled
  - Sinc3 filter selected
  - OSR = 256
  - All the 4 filters are synchronized by using MFE (Master Filter enable bit)
  - Filter output represented in 16 bit format
  - In order to convert 25 bit Data filter into 16 bit format user needs to right shift by 10 bits for Sinc3 filter with OSR = 256
- Interrupt module settings for SDFM filter
  - All the 4 higher threshold comparator interrupts disabled
  - All the 4 lower threshold comparator interrupts disabled
  - All the 4 modulator failure interrupts disabled
  - All the 4 filter will generate interrupt when a new filter data is available

External Connections

- SDFM_PIN_MUX_OPTION1 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO16-GPIO31
- SDFM_PIN_MUX_OPTION2 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO48-GPIO63
- SDFM_PIN_MUX_OPTION3 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO122-GPIO137

Watch Variables

- *filter1Result* - Output of filter 1
- *filter2Result* - Output of filter 2
- *filter3Result* - Output of filter 3
- *filter4Result* - Output of filter 4

28.11.1.4 SDFM PWM Sync

FILE: sdfm_ex4_pwm_sync_cpuread.c

In this example, SDFM filter data is read by CPU in SDFM ISR routine. The SDFM configuration is shown below:

- SDFM1 is used in this example. For using SDFM2, few modifications would be needed in the example.
- MODE0 Input control mode selected
- Comparator settings
  - Sinc3 filter selected
  - OSR = 32
  - hlt = 0x7FFF (Higher threshold setting)
  - llt = 0x0000(Lower threshold setting)

Data filter settings

- All the 4 filter modules enabled
- Sinc3 filter selected
- OSR = 256
- All the 4 filters are synchronized by using PWM (Master Filter enable bit)
- Filter output represented in 16 bit format
- In order to convert 25 bit Data filter into 16 bit format user needs to right shift by 10 bits for Sinc3 filter with OSR = 256

Interrupt module settings for SDFM filter

- All the 4 higher threshold comparator interrupts disabled
- All the 4 lower threshold comparator interrupts disabled
- All the 4 modulator failure interrupts disabled
- All the 4 filter will generate interrupt when a new filter data is available

External Connections

- SDFM_PIN_MUX_OPTION1 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO16-GPIO31
- SDFM_PIN_MUX_OPTION2 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO48-GPIO63
- SDFM_PIN_MUX_OPTION3 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO122-GPIO137

Watch Variables

- *filter1Result* - Output of filter 1
- *filter2Result* - Output of filter 2
- *filter3Result* - Output of filter 3
- *filter4Result* - Output of filter 4

28.11.1.5 SDFM Type 1 Filter FIFO

FILE: sdfm_ex5_type1_filter_fifo_cpuread.c

This example configures SDFM1 filter in type 1 to demonstrate data read through CPU in FIFO & non-FIFO mode. Data filter is configured in mode 0 to select SINC3 filter with OSR of 256. Filter output is configured for 16-bit format and data shift of 10 is used.

This example demonstrates the FIFO usage if enabled. FIFO length is set at 16 and data ready interrupt is configured to be triggered when FIFO is full. In this example, SDFM filter data is read by CPU in SDFM Data Ready ISR routine.

External Connections

- SDFM_PIN_MUX_OPTION1 Connect Sigma-Delta streams(SD1-D1, SD1-C1) to (GPIO16, GPIO17)
- SDFM_PIN_MUX_OPTION2 Connect Sigma-Delta streams(SD1-D1, SD1-C1) to (GPIO48, GPIO49)
- SDFM_PIN_MUX_OPTION3 Connect Sigma-Delta streams(SD1-D1, SD1-C1) to (GPIO122, GPIO123)

Watch Variables

- *filter1Result* - Output of filter 1

28.11.1.6 SDFM Filter Sync CLA

FILE: sdfm_ex6_FIFO_freeze_claread.c

In this example, SDFM FIFO will not be filled until a SDSYNC event. On a SDSYNC event, SDFM data filter output will start filling FIFO and stop filling after programmable number 'N' of FIFO is filled.

SDy-C1 (Filter1 channel clock) is internally configured to connected SDy-C2 / SDy-C3 / SDy-C4 SDFM configuration is shown below:

- SDFM1 used in this example.For using SDFM2, few modifications would be needed in the example.
- MODE0 Input control mode selected
- Comparator settings
  - Sinc3 filter selected
  - OSR = 32
  - hlt = 0x7FFF (Higher threshold setting)
  - llt = 0x0000(Lower threshold setting)
- Data filter settings
  - All the 4 filter modules enabled
  - Sinc3 filter selected
  - OSR = 256
  - All the 4 filters are synchronized by using MFE (Master Filter enable bit)
  - Filter output represented in 16 bit format
  - In order to convert 25 bit Data filter into 16 bit format user needs to right shift by 10 bits for Sinc3 filter with OSR = 256
- Interrupt module settings for SDFM filter
  - All the 4 higher threshold comparator interrupts disabled
  - All the 4 lower threshold comparator interrupts disabled
  - All the 4 modulator failure interrupts disabled
  - All the 4 filter will generate interrupt when a new filter data is available

External Connections

- SDFM_PIN_MUX_OPTION1 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO16-GPIO31
- SDFM_PIN_MUX_OPTION2 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO48-GPIO63
- SDFM_PIN_MUX_OPTION3 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO122-GPIO137

Watch Variables

- *filter1Result* - Output of filter 1
- *filter2Result* - Output of filter 2
- *filter3Result* - Output of filter 3

- *filter4Result* - Output of filter 4

28.12 SDFM Registers

This section describes the Sigma Delta Filter Module registers.

28.12.1 SDFM Base Address Table (C28)

Table 28-9. SDFM Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
Sdfm1Regs	SDFM_REGS	SDFM1_BASE	0x0000_5E00	YES	YES	YES	YES	YES
Sdfm2Regs	SDFM_REGS	SDFM2_BASE	0x0000_5E80	YES	YES	YES	YES	YES

28.12.2 SDFM_REGS Registers

Table 28-10 lists the memory-mapped registers for the SDFM_REGS registers. All register offset addresses not listed in Table 28-10 should be considered as reserved locations and the register contents should not be modified.

Table 28-10. SDFM_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	SDIFLG	SD Interrupt Flag Register		Go
2h	SDIFLGCLR	SD Interrupt Flag Clear Register		Go
4h	SDCTL	SD Control Register	EALLOW	Go
6h	SDMFILEN	SD Master Filter Enable	EALLOW	Go
7h	SDSTATUS	SD Status Register		Go
10h	SDCTLPARM1	Control Parameter Register for Ch1	EALLOW	Go
11h	SDDFPARM1	Data Filter Parameter Register for Ch1	EALLOW	Go
12h	SDDPARM1	Data Parameter Register for Ch1	EALLOW	Go
13h	SDFLT1CMPH1	High-level Threshold Register for Ch1	EALLOW	Go
14h	SDFLT1CMPL1	Low-level Threshold Register for Ch1	EALLOW	Go
15h	SDCPARM1	Comparator Filter Parameter Register for Ch1	EALLOW	Go
16h	SDDATA1	Data Filter Data Register (16 or 32bit) for Ch1		Go
18h	SDDATFIFO1	Filter Data FIFO Output(32b) for Ch1		Go
1Ah	SDCDATA1	Comparator Filter Data Register (16b) for Ch1		Go
1Bh	SDFLT1CMPH2	Second high level threhold for CH1	EALLOW	Go
1Ch	SDFLT1CMPHZ	High-level (Z) Threshold Register for Ch1	EALLOW	Go
1Dh	SDFIFOCTL1	FIFO Control Register for Ch1	EALLOW	Go
1Eh	SDSYNC1	SD Filter Sync control for Ch1	EALLOW	Go
1Fh	SDFLT1CMPL2	Second low level threhold for CH1	EALLOW	Go
20h	SDCTLPARM2	Control Parameter Register for Ch2	EALLOW	Go
21h	SDDFPARM2	Data Filter Parameter Register for Ch2	EALLOW	Go
22h	SDDPARM2	Data Parameter Register for Ch2	EALLOW	Go
23h	SDFLT2CMPH1	High-level Threshold Register for Ch2	EALLOW	Go
24h	SDFLT2CMPL1	Low-level Threshold Register for Ch2	EALLOW	Go
25h	SDCPARM2	Comparator Filter Parameter Register for Ch2	EALLOW	Go
26h	SDDATA2	Data Filter Data Register (16 or 32bit) for Ch2		Go
28h	SDDATFIFO2	Filter Data FIFO Output(32b) for Ch2		Go
2Ah	SDCDATA2	Comparator Filter Data Register (16b) for Ch2		Go
2Bh	SDFLT2CMPH2	Second high level threhold for CH2	EALLOW	Go
2Ch	SDFLT2CMPHZ	High-level (Z) Threshold Register for Ch2	EALLOW	Go
2Dh	SDFIFOCTL2	FIFO Control Register for Ch2	EALLOW	Go
2Eh	SDSYNC2	SD Filter Sync control for Ch2	EALLOW	Go
2Fh	SDFLT2CMPL2	Second low level threhold for CH2	EALLOW	Go
30h	SDCTLPARM3	Control Parameter Register for Ch3	EALLOW	Go
31h	SDDFPARM3	Data Filter Parameter Register for Ch3	EALLOW	Go
32h	SDDPARM3	Data Parameter Register for Ch3	EALLOW	Go
33h	SDFLT3CMPH1	High-level Threshold Register for Ch3	EALLOW	Go
34h	SDFLT3CMPL1	Low-level Threshold Register for Ch3	EALLOW	Go
35h	SDCPARM3	Comparator Filter Parameter Register for Ch3	EALLOW	Go
36h	SDDATA3	Data Filter Data Register (16 or 32bit) for Ch3		Go
38h	SDDATFIFO3	Filter Data FIFO Output(32b) for Ch3		Go

Table 28-10. SDFM_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
3Ah	SDCDATA3	Comparator Filter Data Register (16b) for Ch3		Go
3Bh	SDFLT3CMPH2	Second high level threhold for CH3	EALLOW	Go
3Ch	SDFLT3CMPHZ	High-level (Z) Threshold Register for Ch3	EALLOW	Go
3Dh	SDFIFOCTL3	FIFO Control Register for Ch3	EALLOW	Go
3Eh	SDSYNC3	SD Filter Sync control for Ch3	EALLOW	Go
3Fh	SDFLT3CMPL2	Second low level threhold for CH3	EALLOW	Go
40h	SDCTLPARM4	Control Parameter Register for Ch4	EALLOW	Go
41h	SDDFPARM4	Data Filter Parameter Register for Ch4	EALLOW	Go
42h	SDDPARM4	Data Parameter Register for Ch4	EALLOW	Go
43h	SDFLT4CMPH1	High-level Threshold Register for Ch4	EALLOW	Go
44h	SDFLT4CMPL1	Low-level Threshold Register for Ch4	EALLOW	Go
45h	SDCPARM4	Comparator Filter Parameter Register for Ch4	EALLOW	Go
46h	SDDATA4	Data Filter Data Register (16 or 32bit) for Ch4		Go
48h	SDDATFIFO4	Filter Data FIFO Output(32b) for Ch4		Go
4Ah	SDCDATA4	Comparator Filter Data Register (16b) for Ch4		Go
4Bh	SDFLT4CMPH2	Second high level threhold for CH4	EALLOW	Go
4Ch	SDFLT4CMPHZ	High-level (Z) Threshold Register for Ch4	EALLOW	Go
4Dh	SDFIFOCTL4	FIFO Control Register for Ch4	EALLOW	Go
4Eh	SDSYNC4	SD Filter Sync control for Ch4	EALLOW	Go
4Fh	SDFLT4CMPL2	Second low level threhold for CH4	EALLOW	Go
60h	SDCOMP1CTL	SD Comparator event filter1 Control Register	EALLOW	Go
61h	SDCOMP1EVT2FLTCTL	COMPL/CEVT2 Digital filter1 Control Register	EALLOW	Go
62h	SDCOMP1EVT2FLTCLKCTL	COMPL/CEVT2 Digital filter1 Clock Control Register	EALLOW	Go
63h	SDCOMP1EVT1FLTCTL	COMPH/CEVT1 Digital filter1 Control Register	EALLOW	Go
64h	SDCOMP1EVT1FLTCLKCTL	COMPH/CEVT1 Digital filter1 Clock Control Register	EALLOW	Go
67h	SDCOMP1LOCK	SD compartor event filter1 Lock Register	EALLOW	Go
68h	SDCOMP2CTL	SD Comparator event filter2 Control Register	EALLOW	Go
69h	SDCOMP2EVT2FLTCTL	COMPL/CEVT2 Digital filter2 Control Register	EALLOW	Go
6Ah	SDCOMP2EVT2FLTCLKCTL	COMPL/CEVT2 Digital filter2 Clock Control Register	EALLOW	Go
6Bh	SDCOMP2EVT1FLTCTL	COMPH/CEVT1 Digital filter2 Control Register	EALLOW	Go
6Ch	SDCOMP2EVT1FLTCLKCTL	COMPH/CEVT1 Digital filter2 Clock Control Register	EALLOW	Go
6Fh	SDCOMP2LOCK	SD compartor event filter2 Lock Register	EALLOW	Go
70h	SDCOMP3CTL	SD Comparator event filter3 Control Register	EALLOW	Go
71h	SDCOMP3EVT2FLTCTL	COMPL/CEVT2 Digital filter3 Control Register	EALLOW	Go
72h	SDCOMP3EVT2FLTCLKCTL	COMPL/CEVT2 Digital filter3 Clock Control Register	EALLOW	Go
73h	SDCOMP3EVT1FLTCTL	COMPH/CEVT1 Digital filter3 Control Register	EALLOW	Go
74h	SDCOMP3EVT1FLTCLKCTL	COMPH/CEVT1 Digital filter3 Clock Control Register	EALLOW	Go
77h	SDCOMP3LOCK	SD compartor event filter3 Lock Register	EALLOW	Go
78h	SDCOMP4CTL	SD Comparator event filter4 Control Register	EALLOW	Go
79h	SDCOMP4EVT2FLTCTL	COMPL/CEVT2 Digital filter4 Control Register	EALLOW	Go

Table 28-10. SDFM_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
7Ah	SDCOMP4EVT2FLTCLKCTL	COMPL/CEVT2 Digital filter4 Clock Control Register	EALLOW	Go
7Bh	SDCOMP4EVT1FLTCTL	COMPH/CEVT1 Digital filter4 Control Register	EALLOW	Go
7Ch	SDCOMP4EVT1FLTCLKCTL	COMPH/CEVT1 Digital filter4 Clock Control Register	EALLOW	Go
7Fh	SDCOMP4LOCK	SD compartor event filter4 Lock Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 28-11 shows the codes that are used for access types in this section.

Table 28-11. SDFM_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

28.12.2.1 SDIFLG Register (Offset = 0h) [Reset = 0h]

SDIFLG is shown in Figure 28-15 and described in Table 28-12.

Return to the Summary Table.

SD Interrupt Flag Register

Figure 28-15. SDIFLG Register

31	30	29	28	27	26	25	24
MIF	RESERVED						
R-0h	R-0-0h						
23	22	21	20	19	18	17	16
SDFFINT4	SDFFINT3	SDFFINT2	SDFFINT1	SDFFOVF4	SDFFOVF3	SDFFOVF2	SDFFOVF1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
AF4	AF3	AF2	AF1	MF4	MF3	MF2	MF1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
FLT4_FLG_CEVT2	FLT4_FLG_CEVT1	FLT3_FLG_CEVT2	FLT3_FLG_CEVT1	FLT2_FLG_CEVT2	FLT2_FLG_CEVT1	FLT1_FLG_CEVT2	FLT1_FLG_CEVT1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 28-12. SDIFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MIF	R	0h	Set whenever any "error" interrupt (MF1-4,IFL1-4,IFH1-4,SDFFOVF1-4) is active Reset type: SYSRSn
30-24	RESERVED	R-0	0h	Reserved
23	SDFFINT4	R	0h	SDFIFO data ready interrupt for Ch4 Reset type: SYSRSn
22	SDFFINT3	R	0h	SDFIFO data ready interrupt for Ch3 Reset type: SYSRSn
21	SDFFINT2	R	0h	SDFIFO data ready interrupt for Ch2 Reset type: SYSRSn
20	SDFFINT1	R	0h	SDFIFO data ready interrupt for Ch1 0: SDFIFO data ready interrupt has NOT occurred 1: SDFIFO data ready interrupt has occurred Reset type: SYSRSn
19	SDFFOVF4	R	0h	FIFO Overflow Flag for Ch4 Reset type: SYSRSn
18	SDFFOVF3	R	0h	FIFO Overflow Flag for Ch3 Reset type: SYSRSn
17	SDFFOVF2	R	0h	FIFO Overflow Flag for Ch2 Reset type: SYSRSn
16	SDFFOVF1	R	0h	FIFO Overflow Flag for Ch1 0 - FIFO has not overflowed 1 - FIFO overflowed. # words received in FIFO > FIFO depth (16), NEW word is lost Reset type: SYSRSn
15	AF4	R	0h	Acknowledge flag for Filter 4 0: No new data available for Filter (in non-FIFO mode) 1: New data available for Filter (in non-FIFO mode) Reset type: SYSRSn

Table 28-12. SDIFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14	AF3	R	0h	Acknowledge flag for Filter 3 0: No new data available for Filter (in non-FIFO mode) 1: New data available for Filter (in non-FIFO mode) Reset type: SYSRSn
13	AF2	R	0h	Acknowledge flag for Filter 2 0: No new data available for Filter (in non-FIFO mode) 1: New data available for Filter (in non-FIFO mode) Reset type: SYSRSn
12	AF1	R	0h	Acknowledge flag for Filter 1 0: No new data available for Filter (in non-FIFO mode) 1: New data available for Filter (in non-FIFO mode) Reset type: SYSRSn
11	MF4	R	0h	Modulator Failure for Filter 4 0: Modulator is operating normally for Filter 1: Modulator failure for Filter Reset type: SYSRSn
10	MF3	R	0h	Modulator Failure for Filter 3 0: Modulator is operating normally for Filter 1: Modulator failure for Filter Reset type: SYSRSn
9	MF2	R	0h	Modulator Failure for Filter 2 0: Modulator is operating normally for Filter 1: Modulator failure for Filter Reset type: SYSRSn
8	MF1	R	0h	Modulator Failure for Filter 1 0: Modulator is operating normally for Filter 1: Modulator failure for Filter Reset type: SYSRSn
7	FLT4_FLG_CEVT2	R	0h	CEVT2 Interrupt flag for filter4 0: CEVT2 event has not occured 1: CEVT2 event has occurred Reset type: SYSRSn
6	FLT4_FLG_CEVT1	R	0h	CEVT1 Interrupt flag for filter4 0: CEVT1 event has not occured 1: CEVT1 event has occurred Reset type: SYSRSn
5	FLT3_FLG_CEVT2	R	0h	CEVT2 Interrupt flag for filter3 0: CEVT2 event has not occured 1: CEVT2 event has occurred Reset type: SYSRSn
4	FLT3_FLG_CEVT1	R	0h	CEVT1 Interrupt flag for filter3 0: CEVT1 event has not occured 1: CEVT1 event has occurred Reset type: SYSRSn
3	FLT2_FLG_CEVT2	R	0h	CEVT2 Interrupt flag for filter2 0: CEVT2 event has not occured 1: CEVT2 event has occurred Reset type: SYSRSn
2	FLT2_FLG_CEVT1	R	0h	CEVT1 Interrupt flag for filter2 0: CEVT1 event has not occured 1: CEVT1 event has occurred Reset type: SYSRSn
1	FLT1_FLG_CEVT2	R	0h	CEVT2 Interrupt flag for filter1 0: CEVT2 event has not occured 1: CEVT2 event has occurred Reset type: SYSRSn

Table 28-12. SDIFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	FLT1_FLG_CEVT1	R	0h	CEVT1 Interrupt flag for filter1 0: CEVT1 event has not occured 1: CEVT1 event has occurred Reset type: SYSRSn

28.12.2.2 SDIFLGCLR Register (Offset = 2h) [Reset = 0h]

SDIFLGCLR is shown in Figure 28-16 and described in Table 28-13.

Return to the Summary Table.

SD Module Interrupt Flag Clear Bits:
Writing a "1" will clear the respective flag bit in the SDIFLG register.
Writes of "0" are ignored.
Note: If user writes a "1" to clear a bit on the same cycle that the hardware is trying to set the bit to "1", then hardware has priority and the bit will not be cleared.

Figure 28-16. SDIFLGCLR Register

31	30	29	28	27	26	25	24
MIF	RESERVED						
R-0/W1S-0h	R-0-0h						
23	22	21	20	19	18	17	16
SDFFINT4	SDFFINT3	SDFFINT2	SDFFINT1	SDFFOVF4	SDFFOVF3	SDFFOVF2	SDFFOVF1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
AF4	AF3	AF2	AF1	MF4	MF3	MF2	MF1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
FLT4_FLG_CE VT2	FLT4_FLG_CE VT1	FLT3_FLG_CE VT2	FLT3_FLG_CE VT1	FLT2_FLG_CE VT2	FLT2_FLG_CE VT1	FLT1_FLG_CE VT2	FLT1_FLG_CE VT1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 28-13. SDIFLGCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MIF	R-0/W1S	0h	Flag-clear bit for SDFM Master Interrupt flag. Writing a 1 to clear MIF flag in SDIFLG register Writes of "0" are ignored. Note: If the MIF flag is cleared and other Interrupts are still pending, MIF will again be set to 1 on the following SysClk cycle, and the INT output will be reasserted (pulsed low) Reset type: SYSRSn
30-24	RESERVED	R-0	0h	Reserved
23	SDFFINT4	R-0/W1S	0h	SDFIFO data ready Interrupt flag-clear bit for Ch4 Reset type: SYSRSn
22	SDFFINT3	R-0/W1S	0h	SDFIFO data ready Interrupt flag-clear bit for Ch3 Reset type: SYSRSn
21	SDFFINT2	R-0/W1S	0h	SDFIFO data ready Interrupt flag-clear bit for Ch2 Reset type: SYSRSn
20	SDFFINT1	R-0/W1S	0h	SDFIFO data ready Interrupt flag-clear bit for Ch1 Reset type: SYSRSn
19	SDFFOVF4	R-0/W1S	0h	SDFIFO overflow clear Ch4 Reset type: SYSRSn
18	SDFFOVF3	R-0/W1S	0h	SDFIFO overflow clear Ch3 Reset type: SYSRSn
17	SDFFOVF2	R-0/W1S	0h	SDFIFO overflow clear Ch2 Reset type: SYSRSn
16	SDFFOVF1	R-0/W1S	0h	SDFIFO overflow clear Ch1 Reset type: SYSRSn
15	AF4	R-0/W1S	0h	Flag-clear bit for Acknowledge flag for Filter 4 Reset type: SYSRSn

Table 28-13. SDIFLGCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14	AF3	R-0/W1S	0h	Flag Clear bit for AF3 Reset type: SYSRSn
13	AF2	R-0/W1S	0h	Flag Clear bit for AF2 Reset type: SYSRSn
12	AF1	R-0/W1S	0h	Flag Clear bit for AF1 Reset type: SYSRSn
11	MF4	R-0/W1S	0h	Flag Clear bit for MF4 Reset type: SYSRSn
10	MF3	R-0/W1S	0h	Flag Clear bit for MF3 Reset type: SYSRSn
9	MF2	R-0/W1S	0h	Flag Clear bit for MF2 Reset type: SYSRSn
8	MF1	R-0/W1S	0h	Flag Clear bit for MF1 Reset type: SYSRSn
7	FLT4_FLG_CEVT2	R-0/W1S	0h	Flag Clear bit for FLT4_FLG_CEVT2 Reset type: SYSRSn
6	FLT4_FLG_CEVT1	R-0/W1S	0h	Flag Clear bit for FLT4_FLG_CEVT1 Reset type: SYSRSn
5	FLT3_FLG_CEVT2	R-0/W1S	0h	Flag Clear bit for FLT3_FLG_CEVT2 Reset type: SYSRSn
4	FLT3_FLG_CEVT1	R-0/W1S	0h	Flag Clear bit for FLT3_FLG_CEVT1 Reset type: SYSRSn
3	FLT2_FLG_CEVT2	R-0/W1S	0h	Flag Clear bit for FLT2_FLG_CEVT2 Reset type: SYSRSn
2	FLT2_FLG_CEVT1	R-0/W1S	0h	Flag Clear bit for FLT2_FLG_CEVT1 Reset type: SYSRSn
1	FLT1_FLG_CEVT2	R-0/W1S	0h	Flag Clear bit for FLT1_FLG_CEVT2 Reset type: SYSRSn
0	FLT1_FLG_CEVT1	R-0/W1S	0h	Flag Clear bit for FLT1_FLG_CEVT1 Reset type: SYSRSn

28.12.2.3 SDCTL Register (Offset = 4h) [Reset = 0h]

SDCTL is shown in Figure 28-17 and described in Table 28-14.

Return to the Summary Table.

SD Control Register

Figure 28-17. SDCTL Register

15	14	13	12	11	10	9	8
RESERVED	RESERVED	MIE	RESERVED				
R-0-0h	R-0-0h	R/W-0h	R-0-0h				
7	6	5	4	3	2	1	0
RESERVED				HZ4	HZ3	HZ2	HZ1
R-0-0h				R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 28-14. SDCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14	RESERVED	R-0	0h	Reserved
13	MIE	R/W	0h	Master SDy_ERR interrupt enable 0: SDy_ERR Interrupt and interrupt flags are disabled 1: SDy_ERR Interrupt and interrupt flags are enabled Reset type: SYSRSn
12-4	RESERVED	R-0	0h	Reserved
3	HZ4	R-0/W1S	0h	Flag Clear bit for HZ4 Reset type: SYSRSn
2	HZ3	R-0/W1S	0h	Flag Clear bit for HZ3 Reset type: SYSRSn
1	HZ2	R-0/W1S	0h	Flag Clear bit for HZ2 Reset type: SYSRSn
0	HZ1	R-0/W1S	0h	Flag Clear bit for HZ1 Reset type: SYSRSn

28.12.2.4 SDMFILEN Register (Offset = 6h) [Reset = 0h]

SDMFILEN is shown in Figure 28-18 and described in Table 28-15.

Return to the Summary Table.

SD Master Filter Enable

Figure 28-18. SDMFILEN Register

15	14	13	12	11	10	9	8
RESERVED			RESERVED	MFE	RESERVED	RESERVED	RESERVED
R-0-0h			R-0-0h	R/W-0h	R-0-0h	R-0-0h	R-0-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED			RESERVED			
R-0-0h	R-0-0h			R-0-0h			

Table 28-15. SDMFILEN Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R-0	0h	Reserved
12	RESERVED	R-0	0h	Reserved
11	MFE	R/W	0h	Master Filter Enable 0: All the four data filter units of SDFM module are disabled. All FIFOs are cleared 1: Data filter units can be enabled if bit FEN is '1'. Reset type: SYSRSn
10	RESERVED	R-0	0h	Reserved
9	RESERVED	R-0	0h	Reserved
8-7	RESERVED	R-0	0h	Reserved
6-4	RESERVED	R-0	0h	Reserved
3-0	RESERVED	R-0	0h	Reserved

28.12.2.5 SDSTATUS Register (Offset = 7h) [Reset = 0h]

SDSTATUS is shown in Figure 28-19 and described in Table 28-16.

Return to the Summary Table.

SD Status Register

Figure 28-19. SDSTATUS Register

15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED				HZ4	HZ3	HZ2	HZ1
R-0-0h				R-0h	R-0h	R-0h	R-0h

Table 28-16. SDSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	RESERVED	R	0h	Reserved
13	RESERVED	R	0h	Reserved
12	RESERVED	R	0h	Reserved
11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7-4	RESERVED	R-0	0h	Reserved
3	HZ4	R	0h	High-level Threshold crossing (Z) flag Ch4 Primarily intended for detecting "zero"-crossing events. Unlike the primary comparator IFHx flag, it does not have the ability to generate an interrupt. 0: Comparator filter output < SDCMPHZ4.HLTZ 1: Comparator filter output >= SDCMPHZ4.HLTZ Reset type: SYSRSn
2	HZ3	R	0h	High-level Threshold crossing (Z) flag Ch3 Primarily intended for detecting "zero"-crossing events. Unlike the primary comparator IFHx flag, it does not have the ability to generate an interrupt. 0: Comparator filter output < SDCMPHZ3.HLTZ 1: Comparator filter output >= SDCMPHZ3.HLTZ Reset type: SYSRSn
1	HZ2	R	0h	High-level Threshold crossing (Z) flag Ch2 Primarily intended for detecting "zero"-crossing events. Unlike the primary comparator IFHx flag, it does not have the ability to generate an interrupt. 0: Comparator filter output < SDCMPHZ2.HLTZ 1: Comparator filter output >= SDCMPHZ2.HLTZ Reset type: SYSRSn
0	HZ1	R	0h	High-level Threshold crossing (Z) flag Ch1 Primarily intended for detecting "zero"-crossing events. Unlike the primary comparator IFHx flag, it does not have the ability to generate an interrupt. 0: Comparator filter output < SDCMPHZ1.HLTZ 1: Comparator filter output >= SDCMPHZ1.HLTZ Reset type: SYSRSn

28.12.2.6 SDCTLPARM1 Register (Offset = 10h) [Reset = 0h]

SDCTLPARM1 is shown in Figure 28-20 and described in Table 28-17.

Return to the Summary Table.

Control Parameter Register for Ch1

Figure 28-20. SDCTLPARM1 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED	SDDATASYNC	RESERVED	SDCLKSYNC	SDCLKSEL	RESERVED	MOD	
R-0-0h	R/W-0h	R-0-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	

Table 28-17. SDCTLPARM1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R-0	0h	Reserved
7	RESERVED	R-0	0h	Reserved
6	SDDATASYNC	R/W	0h	0: SD Data is not passed through a synchronizer. 1: SD Data is passed through a synchronizer. Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4	SDCLKSYNC	R/W	0h	0: SD Clock is not passed through a synchronizer. 1: SD Clock is passed through a synchronizer. Reset type: SYSRSn
3	SDCLKSEL	R/W	0h	SD1 Clock source select. 0: Clock source to SDFM filter is its channel clock. 1: Clock source to SDFM filter is SD1 filter clock. Reset type: SYSRSn
2	RESERVED	R/W	0h	Reserved
1-0	MOD	R/W	0h	Modulator clock modes 0: Mode 0: Modulator clock running at 1x data rate 1: Reserved 2: Reserved 3: Reserved Reset type: SYSRSn

28.12.2.7 SDDFPARM1 Register (Offset = 11h) [Reset = 0h]

SDDFPARM1 is shown in Figure 28-21 and described in Table 28-18.

Return to the Summary Table.

Data Filter Parameter Register for Ch1

Figure 28-21. SDDFPARM1 Register

15	14	13	12	11	10	9	8
RESERVED			SDSYNCEN	SST		AE	FEN
R-0h			R/W-0h	R/W-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DOSR							
R/W-0h							

Table 28-18. SDDFPARM1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12	SDSYNCEN	R/W	0h	PWM synchronization (SDSYNC) of data filter 0: PWM synchronization of data filter is disabled 1: PWM synchronization of data filter is enabled Note: SDSYNCx.SYNCSEL bits define which PWM signal is used to synchronize PWMs Reset type: SYSRSn
11-10	SST	R/W	0h	Data filter structure 00: Data filter runs with a Sincfast structure 01: Data filter runs with a Sinc1 structure 10: Data filter runs with a Sinc2 structure 11: Data filter runs with a Sinc3 structure Reset type: SYSRSn
9	AE	R/W	0h	Data filter Acknowledge Enable 0: Acknowledge flag is disabled for the particular filter 1: Acknowledge flag is enabled for the particular filter Reset type: SYSRSn
8	FEN	R/W	0h	Filter Enable 0: The data filter is disabled and no data is produced 1: The data filter is enabled and data are produced in the data filter Note: When filter is disabled, DOSR counter held in reset, filter data erased. Also resets FIFO pointers and clears the FIFO Reset type: SYSRSn
7-0	DOSR	R/W	0h	Data filter Oversampling ratio The actual oversampling ratio of data filter is DOSR + 1 These bits set the oversampling ratio of the data filter. 0x0FF represents an oversampling ratio of 256. Reset type: SYSRSn

28.12.2.8 SDDPARM1 Register (Offset = 12h) [Reset = 0h]

SDDPARM1 is shown in Figure 28-22 and described in Table 28-19.

Return to the Summary Table.

Data Parameter Register for Ch1

Figure 28-22. SDDPARM1 Register

15	14	13	12	11	10	9	8
SH					DR	RESERVED	
R/W-0h					R/W-0h	R-0-0h	
7	6	5	4	3	2	1	0
RESERVED							
R-0-0h							

Table 28-19. SDDPARM1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	SH	R/W	0h	Shift Control These bits indicate by how many bits the 16-bit window is shifted up when 16-bit data representation is chosen. Reset type: SYSRSn
10	DR	R/W	0h	Data filter Data representation 0: Data stored in 16b 2's complement 1: Data stored in 32b 2's complement Reset type: SYSRSn
9-0	RESERVED	R-0	0h	Reserved

28.12.2.9 SDFLT1CMPH1 Register (Offset = 13h) [Reset = 7FFFh]

SDFLT1CMPH1 is shown in Figure 28-23 and described in Table 28-20.

Return to the Summary Table.

High-level Threshold Register for Ch1

Figure 28-23. SDFLT1CMPH1 Register

15	14	13	12	11	10	9	8
RESERVED	HLT						
R-0-0h	R/W-7FFFh						
7	6	5	4	3	2	1	0
HLT							
R/W-7FFFh							

Table 28-20. SDFLT1CMPH1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLT	R/W	7FFFh	Unsigned high-level threshold for the comparator filter output. Reset type: SYSRSn

28.12.2.10 SDFLT1CMPL1 Register (Offset = 14h) [Reset = 0h]

SDFLT1CMPL1 is shown in Figure 28-24 and described in Table 28-21.

Return to the Summary Table.

Low-level Threshold Register for Ch1

Figure 28-24. SDFLT1CMPL1 Register

15	14	13	12	11	10	9	8
RESERVED	LLT						
R-0-0h	R/W-0h						
7	6	5	4	3	2	1	0
LLT							
R/W-0h							

Table 28-21. SDFLT1CMPL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	LLT	R/W	0h	Unsigned low-level threshold for the comparator filter output. Reset type: SYSRSn

28.12.2.11 SDCPARM1 Register (Offset = 15h) [Reset = 2000h]

SDCPARM1 is shown in Figure 28-25 and described in Table 28-22.

Return to the Summary Table.

Comparator Filter Parameter Register for Ch1

Figure 28-25. SDCPARM1 Register

15	14	13	12	11	10	9	8
CEVT2SEL		CEN	CEVT1SEL		HZEN	MFIE	CS1_CS0
R/W-0h		R/W-1h	R/W-0h		R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CS1_CS0	EN_CEVT2	EN_CEVT1	COSR				
R/W-0h	R/W-0h	R/W-0h	R/W-0h				

Table 28-22. SDCPARM1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	CEVT2SEL	R/W	0h	Comparator Event2 Select 00: COMPL1 01: COMPL1 OR COMPH1 10: COMPL2 11: COMPL2 OR COMPH2 Reset type: SYSRSn
13	CEN	R/W	1h	Comparator Filter enable 0: Disable comparator filter 1: Enable comparator filter Reset type: SYSRSn
12-11	CEVT1SEL	R/W	0h	Comparator Event1 Select 00: COMPH1 01: COMPL1 OR COMPH1 10: COMPH2 11: COMPL2 OR COMPH2 Reset type: SYSRSn
10	HZEN	R/W	0h	High level (Z) Threshold crossing output enable 0: Disable Higher level Threshold (Z) crossing 1: Enable Higher level Threhold (Z) crossing Reset type: SYSRSn
9	MFIE	R/W	0h	Modulator Failure Interrupt Enable 0: Disable modulator failure interrupt and its flag 1: Enable modulator failure interrupt and its flag Reset type: SYSRSn
8-7	CS1_CS0	R/W	0h	Comparator filter structure 00: Comparator filter runs with a sincfast structure 01: Comparator filter runs with a Sinc1 structure 10: Comparator filter runs with a Sinc2 structure 11: Comparator filter runs with a Sinc3 structure Reset type: SYSRSn
6	EN_CEVT2	R/W	0h	CEVT2 interrupt enable 0: Disable CEVT2 interrupt 1: Enable CEVT2 interrupt Reset type: SYSRSn
5	EN_CEVT1	R/W	0h	CEVT1 interrupt enable 0: Disable CEVT1 interrupt 1: Enable CEVT1 interrupt Reset type: SYSRSn

Table 28-22. SDCPARM1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	COSR	R/W	0h	Comparator Oversampling ratio. The actual rate is COSR + 1. These bits set the oversampling ratio of the filter. 0x1F represents an oversampling ratio of 32 Reset type: SYSRSn

28.12.2.12 SDDATA1 Register (Offset = 16h) [Reset = 0h]

SDDATA1 is shown in Figure 28-26 and described in Table 28-23.

Return to the Summary Table.

Data Filter Data Register (16 or 32bit) for Ch1

Figure 28-26. SDDATA1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA32HI																DATA16															
R-0h																R-0h															

Table 28-23. SDDATA1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA32HI	R	0h	Hi-order 16b in 32b mode, 16-bit Data in 16b mode Reset type: SYSRSn
15-0	DATA16	R	0h	Lo-order 16b in 32b mode Reset type: SYSRSn

28.12.2.13 SDDATFIFO1 Register (Offset = 18h) [Reset = 0h]

SDDATFIFO1 is shown in Figure 28-27 and described in Table 28-24.

Return to the Summary Table.

Filter Data FIFO Output(32b) for Ch1

Figure 28-27. SDDATFIFO1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA32HI																DATA16															
R-0h																R-0h															

Table 28-24. SDDATFIFO1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA32HI	R	0h	Hi-order 16b in 32b mode, 16-bit Data in 16b mode Reset type: SYSRSn
15-0	DATA16	R	0h	Lo-order 16b in 32b mode Reset type: SYSRSn

28.12.2.14 SDCDATA1 Register (Offset = 1Ah) [Reset = 0h]

SDCDATA1 is shown in Figure 28-28 and described in Table 28-25.

Return to the Summary Table.

Comparator Filter Data Register (16b) for Ch1

Figure 28-28. SDCDATA1 Register

15	14	13	12	11	10	9	8
DATA16							
R-0h							
7	6	5	4	3	2	1	0
DATA16							
R-0h							

Table 28-25. SDCDATA1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DATA16	R	0h	Comparator Data output - 16b only Reset type: SYSRSn

28.12.2.15 SDFLT1CMPH2 Register (Offset = 1Bh) [Reset = 7FFFh]

SDFLT1CMPH2 is shown in Figure 28-29 and described in Table 28-26.

Return to the Summary Table.

Second high level threhold for CH1

Figure 28-29. SDFLT1CMPH2 Register

15	14	13	12	11	10	9	8
RESERVED	HLT2						
R-0-0h	R/W-7FFFh						
7	6	5	4	3	2	1	0
HLT2							
R/W-7FFFh							

Table 28-26. SDFLT1CMPH2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLT2	R/W	7FFFh	Second Unsigned high-level threshold for the comparator filter output. Reset type: SYSRSn

28.12.2.16 SDFLT1CMPHZ Register (Offset = 1Ch) [Reset = 0h]

SDFLT1CMPHZ is shown in Figure 28-30 and described in Table 28-27.

Return to the Summary Table.

High-level (Z) Threshold Register for Ch1

Figure 28-30. SDFLT1CMPHZ Register

15	14	13	12	11	10	9	8
RESERVED	HLTZ						
R-0-0h	R/W-0h						
7	6	5	4	3	2	1	0
HLTZ							
R/W-0h							

Table 28-27. SDFLT1CMPHZ Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLTZ	R/W	0h	Unsigned High-level threshold (Z) for the comparator filter output Primarily intended for detecting "zero"-crossing events. Unlike the primary comparator SDCMPHx, it does not have the ability to generate an interrupt. Reset type: SYSRSn

28.12.2.17 SDFIFOCTL1 Register (Offset = 1Dh) [Reset = 0h]

SDFIFOCTL1 is shown in Figure 28-31 and described in Table 28-28.

Return to the Summary Table.

FIFO Control Register for Ch1

Figure 28-31. SDFIFOCTL1 Register

15	14	13	12	11	10	9	8
OVFIEN	DRINTSEL	FFEN	FFIEN	RESERVED	SDFFST		
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0-0h	R-0h		
7	6	5	4	3	2	1	0
SDFFST		RESERVED	SDFFIL				
R-0h		R-0-0h	R/W-0h				

Table 28-28. SDFIFOCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	OVFIEN	R/W	0h	SDFIFO Overflow interrupt enable 0: SDFIFO Overflow condition will not generate an interrupt 1: SDFIFO overflow condition generates an interrupt on SDy_ERR Reset type: SYSRSn
14	DRINTSEL	R/W	0h	Data-Ready Interrupt (DRINT) source select 0 = AF1 (Select non-FIFO data-ready interrupt) 1 = SDFFINT1 (Select FIFO data-ready interrupt) Reset type: SYSRSn
13	FFEN	R/W	0h	SDFIFO Enable 0: Disable FIFO operation 1: Enable FIFO operation Note: When FIFO is disabled, FIFO contents are cleared Reset type: SYSRSn
12	FFIEN	R/W	0h	SDFIFO data ready Interrupt Enable Reset type: SYSRSn
11	RESERVED	R-0	0h	Reserved
10-6	SDFFST	R	0h	SDFIFO Status 00000 FIFO empty 00001 FIFO has 1 word . . . . 10000 FIFO has 16 words Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4-0	SDFFIL	R/W	0h	SDFIFO interrupt level bits The FIFO will generate an interrupt when the FIFO status (SDFFST) >= FIFO level (SDFFIL) Reset type: SYSRSn

28.12.2.18 SDSYNC1 Register (Offset = 1Eh) [Reset = 43Fh]

SDSYNC1 is shown in Figure 28-32 and described in Table 28-29.

Return to the Summary Table.

SD Filter Sync control for Ch1

Figure 28-32. SDSYNC1 Register

15	14	13	12	11	10	9	8
RESERVED					WTSCLREN	FFSYNCCLREN	WTSYNCLR
R-0-0h					R/W-1h	R/W-0h	R-0/W-0h
7	6	5	4	3	2	1	0
WTSYNFLG	WTSYNCEN	SYNCSEL					
R-0h	R/W-0h	R/W-3Fh					

Table 28-29. SDSYNC1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R-0	0h	Reserved
10	WTSCLREN	R/W	1h	WTSYNFLG Clear-on-FIFOINT Enable 0: WTSYNFLG can only be cleared manually (using WTSYNCLR bit) 1: WTSYNFLG is cleared automatically on SDFFINT Reset type: SYSRSn
9	FFSYNCCLREN	R/W	0h	FIFO Clear-on-SDSYNC Enable 0: SDFIFO is not automaticaly cleared upon receiving SDSYNC 1: SDFIFO is automaticaly cleared upon receiving SDSYNC Reset type: SYSRSn
8	WTSYNCLR	R-0/W	0h	Wait-for-Sync Flag Clear (always reads 0) 0: Write of 0 has no affect 1: Write of 1 clears WTSYNFLG Reset type: SYSRSn
7	WTSYNFLG	R	0h	Wait-for-Sync Flag 0: SDSYNC event has not occurred 1: SDSYNC event occurred. Reset type: SYSRSn
6	WTSYNCEN	R/W	0h	Wait-for-Sync Enable 0: Incoming Data written to SDFIFO on every Data-Ready (DR) Event 1: Incoming Data written to SDFIFO on DR event only after SDSYNC event occurs Reset type: SYSRSn
5-0	SYNCSEL	R/W	3Fh	Defines source for the SDSYNC Input on this channel Refer SDSYNCx.SYNCSEL table Reset type: SYSRSn

28.12.2.19 SDFLT1CMPL2 Register (Offset = 1Fh) [Reset = 0h]

SDFLT1CMPL2 is shown in Figure 28-33 and described in Table 28-30.

Return to the Summary Table.

Second low level threhold for CH1

Figure 28-33. SDFLT1CMPL2 Register

15	14	13	12	11	10	9	8
RESERVED	LLT2						
R-0-0h	R/W-0h						
7	6	5	4	3	2	1	0
LLT2							
R/W-0h							

Table 28-30. SDFLT1CMPL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	LLT2	R/W	0h	Second Unsigned low-level threshold for the comparator filter output. Reset type: SYSRSn

28.12.2.20 SDCTLPARM2 Register (Offset = 20h) [Reset = 0h]

SDCTLPARM2 is shown in Figure 28-34 and described in Table 28-31.

Return to the Summary Table.

Control Parameter Register for Ch2

Figure 28-34. SDCTLPARM2 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	SDDATASYNC	RESERVED	SDCLKSYNC	SDCLKSEL	RESERVED	MOD	
R-0-0h	R/W-0h	R-0-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	

Table 28-31. SDCTLPARM2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	RESERVED	R-0	0h	Reserved
6	SDDATASYNC	R/W	0h	0: SD Data is not passed through a synchronizer. 1: SD Data is passed through a synchronizer. Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4	SDCLKSYNC	R/W	0h	0: SD Clock is not passed through a synchronizer. 1: SD Clock is passed through a synchronizer. Reset type: SYSRSn
3	SDCLKSEL	R/W	0h	SD2 Clock source select. 0: Clock source to SDFM filter is its channel clock. 1: Clock source to SDFM filter is SD1 filter clock. Reset type: SYSRSn
2	RESERVED	R/W	0h	Reserved
1-0	MOD	R/W	0h	Modulator clock modes 0: Mode 0: Modulator clock running at 1x data rate 1: Reserved 2: Reserved 3: Reserved Reset type: SYSRSn

28.12.2.21 SDDFPARM2 Register (Offset = 21h) [Reset = 0h]

SDDFPARM2 is shown in Figure 28-35 and described in Table 28-32.

Return to the Summary Table.

Data Filter Parameter Register for Ch2

Figure 28-35. SDDFPARM2 Register

15	14	13	12	11	10	9	8
RESERVED			SDSYNCEN	SST		AE	FEN
R-0h			R/W-0h	R/W-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DOSR							
R/W-0h							

Table 28-32. SDDFPARM2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12	SDSYNCEN	R/W	0h	PWM synchronization (SDSYNC) of data filter 0: PWM synchronization of data filter is disabled 1: PWM synchronization of data filter is enabled Note: SDSYNCx.SYNCSEL bits define which PWM signal is used to synchronize PWMs Reset type: SYSRSn
11-10	SST	R/W	0h	Data filter structure 00: Data filter runs with a Sincfast structure 01: Data filter runs with a Sinc1 structure 10: Data filter runs with a Sinc2 structure 11: Data filter runs with a Sinc3 structure Reset type: SYSRSn
9	AE	R/W	0h	Data filter Acknowledge Enable 0: Acknowledge flag is disabled for the particular filter 1: Acknowledge flag is enabled for the particular filter Reset type: SYSRSn
8	FEN	R/W	0h	Filter Enable 0: The data filter is disabled and no data is produced 1: The data filter is enabled and data are produced in the data filter Note: When filter is disabled, DOSR counter held in reset, filter data erased. Also resets FIFO pointers and clears the FIFO Reset type: SYSRSn
7-0	DOSR	R/W	0h	Data filter Oversampling ratio The actual oversampling ratio of data filter is DOSR + 1 These bits set the oversampling ratio of the data filter. 0x0FF represents an oversampling ratio of 256. Reset type: SYSRSn

28.12.2.22 SDDPARM2 Register (Offset = 22h) [Reset = 0h]

SDDPARM2 is shown in Figure 28-36 and described in Table 28-33.

Return to the Summary Table.

Data Parameter Register for Ch2

Figure 28-36. SDDPARM2 Register

15	14	13	12	11	10	9	8
SH					DR	RESERVED	
R/W-0h					R/W-0h	R-0-0h	
7	6	5	4	3	2	1	0
RESERVED							
R-0-0h							

Table 28-33. SDDPARM2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	SH	R/W	0h	Shift Control These bits indicate by how many bits the 16-bit window is shifted up when 16-bit data representation is chosen. Reset type: SYSRSn
10	DR	R/W	0h	Data filter Data representation 0: Data stored in 16b 2's complement 1: Data stored in 32b 2's complement Reset type: SYSRSn
9-0	RESERVED	R-0	0h	Reserved

28.12.2.23 SDFLT2CMPH1 Register (Offset = 23h) [Reset = 7FFFh]

SDFLT2CMPH1 is shown in Figure 28-37 and described in Table 28-34.

Return to the Summary Table.

High-level Threshold Register for Ch2

Figure 28-37. SDFLT2CMPH1 Register

15	14	13	12	11	10	9	8
RESERVED	HLT						
R-0-0h	R/W-7FFFh						
7	6	5	4	3	2	1	0
HLT							
R/W-7FFFh							

Table 28-34. SDFLT2CMPH1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLT	R/W	7FFFh	Unsigned high-level threshold for the comparator filter output. Reset type: SYSRSn

28.12.2.24 SDFLT2CMPL1 Register (Offset = 24h) [Reset = 0h]

SDFLT2CMPL1 is shown in Figure 28-38 and described in Table 28-35.

Return to the Summary Table.

Low-level Threshold Register for Ch2

Figure 28-38. SDFLT2CMPL1 Register

15	14	13	12	11	10	9	8
RESERVED	LLT						
R-0-0h	R/W-0h						
7	6	5	4	3	2	1	0
LLT							
R/W-0h							

Table 28-35. SDFLT2CMPL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	LLT	R/W	0h	Unsigned low-level threshold for the comparator filter output. Reset type: SYSRSn

28.12.2.25 SDCPARM2 Register (Offset = 25h) [Reset = 2000h]

SDCPARM2 is shown in Figure 28-39 and described in Table 28-36.

Return to the Summary Table.

Comparator Filter Parameter Register for Ch2

Figure 28-39. SDCPARM2 Register

15	14	13	12	11	10	9	8
CEVT2SEL		CEN	CEVT1SEL		HZEN	MFIE	CS1_CS0
R/W-0h		R/W-1h	R/W-0h		R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CS1_CS0	EN_CEVT2	EN_CEVT1	COSR				
R/W-0h	R/W-0h	R/W-0h	R/W-0h				

Table 28-36. SDCPARM2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	CEVT2SEL	R/W	0h	Comparator Event2 Select 00: COMPL1 01: COMPL1 OR COMPH1 10: COMPL2 11: COMPL2 OR COMPH2 Reset type: SYSRSn
13	CEN	R/W	1h	Comparator Filter enable 0: Disable comparator filter 1: Enable comparator filter Reset type: SYSRSn
12-11	CEVT1SEL	R/W	0h	Comparator Event1 Select 00: COMPH1 01: COMPL1 OR COMPH1 10: COMPH2 11: COMPL2 OR COMPH2 Reset type: SYSRSn
10	HZEN	R/W	0h	High level (Z) Threshold crossing output enable 0: Disable Higher level Threshold (Z) crossing 1: Enable Higher level Threhold (Z) crossing Reset type: SYSRSn
9	MFIE	R/W	0h	Modulator Failure Interrupt Enable 0: Disable modulator failure interrupt and its flag 1: Enable modulator failure interrupt and its flag Reset type: SYSRSn
8-7	CS1_CS0	R/W	0h	Comparator filter structure 00: Comparator filter runs with a sincfast structure 01: Comparator filter runs with a Sinc1 structure 10: Comparator filter runs with a Sinc2 structure 11: Comparator filter runs with a Sinc3 structure Reset type: SYSRSn
6	EN_CEVT2	R/W	0h	CEVT2 interrupt enable 0: Disable CEVT2 interrupt 1: Enable CEVT2 interrupt Reset type: SYSRSn
5	EN_CEVT1	R/W	0h	CEVT1 interrupt enable 0: Disable CEVT1 interrupt 1: Enable CEVT1 interrupt Reset type: SYSRSn

Table 28-36. SDCPARM2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	COSR	R/W	0h	Comparator Oversampling ratio. The actual rate is COSR + 1. These bits set the oversampling ratio of the filter. 0x1F represents an oversampling ratio of 32 Reset type: SYSRSn

28.12.2.26 SDDATA2 Register (Offset = 26h) [Reset = 0h]

SDDATA2 is shown in Figure 28-40 and described in Table 28-37.

Return to the Summary Table.

Data Filter Data Register (16 or 32bit) for Ch2

Figure 28-40. SDDATA2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA32HI																DATA16															
R-0h																R-0h															

Table 28-37. SDDATA2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA32HI	R	0h	Hi-order 16b in 32b mode, 16-bit Data in 16b mode Reset type: SYSRSn
15-0	DATA16	R	0h	Lo-order 16b in 32b mode Reset type: SYSRSn

28.12.2.27 SDDATFIFO2 Register (Offset = 28h) [Reset = 0h]

SDDATFIFO2 is shown in Figure 28-41 and described in Table 28-38.

Return to the Summary Table.

Filter Data FIFO Output(32b) for Ch2

Figure 28-41. SDDATFIFO2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA32HI																DATA16															
R-0h																R-0h															

Table 28-38. SDDATFIFO2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA32HI	R	0h	Hi-order 16b in 32b mode, 16-bit Data in 16b mode Reset type: SYSRSn
15-0	DATA16	R	0h	Lo-order 16b in 32b mode Reset type: SYSRSn

28.12.2.28 SDCDATA2 Register (Offset = 2Ah) [Reset = 0h]

SDCDATA2 is shown in Figure 28-42 and described in Table 28-39.

Return to the Summary Table.

Comparator Filter Data Register (16b) for Ch2

Figure 28-42. SDCDATA2 Register

15	14	13	12	11	10	9	8
DATA16							
R-0h							
7	6	5	4	3	2	1	0
DATA16							
R-0h							

Table 28-39. SDCDATA2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DATA16	R	0h	Comparator Data output - 16b only Reset type: SYSRSn

28.12.2.29 SDFLT2CMPH2 Register (Offset = 2Bh) [Reset = 7FFFh]

SDFLT2CMPH2 is shown in Figure 28-43 and described in Table 28-40.

Return to the Summary Table.

Second high level threhold for CH2

Figure 28-43. SDFLT2CMPH2 Register

15	14	13	12	11	10	9	8
RESERVED	HLT2						
R-0-0h	R/W-7FFFh						
7	6	5	4	3	2	1	0
HLT2							
R/W-7FFFh							

Table 28-40. SDFLT2CMPH2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLT2	R/W	7FFFh	Second Unsigned high-level threshold for the comparator filter output. Reset type: SYSRSn

28.12.2.30 SDFLT2CMPHZ Register (Offset = 2Ch) [Reset = 0h]

SDFLT2CMPHZ is shown in Figure 28-44 and described in Table 28-41.

Return to the Summary Table.

High-level (Z) Threshold Register for Ch2

Figure 28-44. SDFLT2CMPHZ Register

15	14	13	12	11	10	9	8
RESERVED	HLTZ						
R-0-0h	R/W-0h						
7	6	5	4	3	2	1	0
HLTZ							
R/W-0h							

Table 28-41. SDFLT2CMPHZ Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLTZ	R/W	0h	Unsigned High-level threshold (Z) for the comparator filter output Primarily intended for detecting "zero"-crossing events. Unlike the primary comparator SDCMPHx, it does not have the ability to generate an interrupt. Reset type: SYSRSn

28.12.2.31 SDFIFOCTL2 Register (Offset = 2Dh) [Reset = 0h]

SDFIFOCTL2 is shown in Figure 28-45 and described in Table 28-42.

Return to the Summary Table.

FIFO Control Register for Ch2

Figure 28-45. SDFIFOCTL2 Register

15	14	13	12	11	10	9	8
OVFIEN	DRINTSEL	FFEN	FFIEN	RESERVED	SDFFST		
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0-0h	R-0h		
7	6	5	4	3	2	1	0
SDFFST		RESERVED	SDFFIL				
R-0h		R-0-0h	R/W-0h				

Table 28-42. SDFIFOCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	OVFIEN	R/W	0h	SDFIFO Overflow interrupt enable 0: SDFIFO Overflow condition will not generate an interrupt 1: SDFIFO overflow condition generates an interrupt on SDy_ERR Reset type: SYSRSn
14	DRINTSEL	R/W	0h	Data-Ready Interrupt (DRINT) source select 0 = AF2 (Select non-FIFO data-ready interrupt) 1 = SDFFINT2 (Select FIFO data-ready interrupt) Reset type: SYSRSn
13	FFEN	R/W	0h	SDFIFO Enable 0: Disable FIFO operation 1: Enable FIFO operation Note: When FIFO is disabled, FIFO contents are cleared Reset type: SYSRSn
12	FFIEN	R/W	0h	SDFIFO data ready Interrupt Enable Reset type: SYSRSn
11	RESERVED	R-0	0h	Reserved
10-6	SDFFST	R	0h	SDFIFO Status 00000 FIFO empty 00001 FIFO has 1 word . . . . 10000 FIFO has 16 words Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4-0	SDFFIL	R/W	0h	SDFIFO interrupt level bits The FIFO will generate an interrupt when the FIFO status (SDFFST) >= FIFO level (SDFFIL) Reset type: SYSRSn

28.12.2.32 SDSYNC2 Register (Offset = 2Eh) [Reset = 43Fh]

SDSYNC2 is shown in Figure 28-46 and described in Table 28-43.

Return to the Summary Table.

SD Filter Sync control for Ch2

Figure 28-46. SDSYNC2 Register

15	14	13	12	11	10	9	8
RESERVED					WTSCLREN	FFSYNCCLREN	WTSYNCLR
R-0-0h					R/W-1h	R/W-0h	R-0/W-0h
7	6	5	4	3	2	1	0
WTSYNFLG	WTSYNCEN	SYNCSEL					
R-0h	R/W-0h	R/W-3Fh					

Table 28-43. SDSYNC2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R-0	0h	Reserved
10	WTSCLREN	R/W	1h	WTSYNFLG Clear-on-FIFOINT Enable 0: WTSYNFLG can only be cleared manually (using WTSYNCLR bit) 1: WTSYNFLG is cleared automatically on SDFFINT Reset type: SYSRSn
9	FFSYNCCLREN	R/W	0h	FIFO Clear-on-SDSYNC Enable 0: SDFIFO is not automaticaly cleared upon receiving SDSYNC 1: SDFIFO is automaticaly cleared upon receiving SDSYNC Reset type: SYSRSn
8	WTSYNCLR	R-0/W	0h	Wait-for-Sync Flag Clear (always reads 0) 0: Write of 0 has no affect 1: Write of 1 clears WTSYNFLG Reset type: SYSRSn
7	WTSYNFLG	R	0h	Wait-for-Sync Flag 0: SDSYNC event has not occurred 1: SDSYNC event occurred. Reset type: SYSRSn
6	WTSYNCEN	R/W	0h	Wait-for-Sync Enable 0: Incoming Data written to SDFIFO on every Data-Ready (DR) Event 1: Incoming Data written to SDFIFO on DR event only after SDSYNC event occurs Reset type: SYSRSn
5-0	SYNCSEL	R/W	3Fh	Defines source for the SDSYNC Input on this channel Refer SDSYNCx.SYNCSEL table Reset type: SYSRSn

28.12.2.33 SDFLT2CMPL2 Register (Offset = 2Fh) [Reset = 0h]

SDFLT2CMPL2 is shown in Figure 28-47 and described in Table 28-44.

Return to the Summary Table.

Second low level threhold for CH2

Figure 28-47. SDFLT2CMPL2 Register

15	14	13	12	11	10	9	8
RESERVED	LLT2						
R-0-0h	R/W-0h						
7	6	5	4	3	2	1	0
LLT2							
R/W-0h							

Table 28-44. SDFLT2CMPL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	LLT2	R/W	0h	Second Unsigned low-level threshold for the comparator filter output. Reset type: SYSRSn

28.12.2.34 SDCTLPARM3 Register (Offset = 30h) [Reset = 0h]

SDCTLPARM3 is shown in Figure 28-48 and described in Table 28-45.

Return to the Summary Table.

Control Parameter Register for Ch3

Figure 28-48. SDCTLPARM3 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	SDDATASYNC	RESERVED	SDCLKSYNC	SDCLKSEL	RESERVED	MOD	
R-0-0h	R/W-0h	R-0-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	

Table 28-45. SDCTLPARM3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	RESERVED	R-0	0h	Reserved
6	SDDATASYNC	R/W	0h	0: SD Data is not passed through a synchronizer. 1: SD Data is passed through a synchronizer. Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4	SDCLKSYNC	R/W	0h	0: SD Clock is not passed through a synchronizer. 1: SD Clock is passed through a synchronizer. Reset type: SYSRSn
3	SDCLKSEL	R/W	0h	SD3 Clock source select. 0: Clock source to SDFM filter is its channel clock. 1: Clock source to SDFM filter is SD1 filter clock. Reset type: SYSRSn
2	RESERVED	R/W	0h	Reserved
1-0	MOD	R/W	0h	Modulator clock modes 0: Mode 0: Modulator clock running at 1x data rate 1: Reserved 2: Reserved 3: Reserved Reset type: SYSRSn

28.12.2.35 SDDFPARM3 Register (Offset = 31h) [Reset = 0h]

SDDFPARM3 is shown in Figure 28-49 and described in Table 28-46.

Return to the Summary Table.

Data Filter Parameter Register for Ch3

Figure 28-49. SDDFPARM3 Register

15	14	13	12	11	10	9	8
RESERVED			SDSYNCEN	SST		AE	FEN
R-0h			R/W-0h	R/W-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DOSR							
R/W-0h							

Table 28-46. SDDFPARM3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12	SDSYNCEN	R/W	0h	PWM synchronization (SDSYNC) of data filter 0: PWM synchronization of data filter is disabled 1: PWM synchronization of data filter is enabled Note: SDSYNCx.SYNCSEL bits define which PWM signal is used to synchronize PWMs Reset type: SYSRSn
11-10	SST	R/W	0h	Data filter structure 00: Data filter runs with a Sincfast structure 01: Data filter runs with a Sinc1 structure 10: Data filter runs with a Sinc2 structure 11: Data filter runs with a Sinc3 structure Reset type: SYSRSn
9	AE	R/W	0h	Data filter Acknowledge Enable 0: Acknowledge flag is disabled for the particular filter 1: Acknowledge flag is enabled for the particular filter Reset type: SYSRSn
8	FEN	R/W	0h	Filter Enable 0: The data filter is disabled and no data is produced 1: The data filter is enabled and data are produced in the data filter Note: When filter is disabled, DOSR counter held in reset, filter data erased. Also resets FIFO pointers and clears the FIFO Reset type: SYSRSn
7-0	DOSR	R/W	0h	Data filter Oversampling ratio The actual oversampling ratio of data filter is DOSR + 1 These bits set the oversampling ratio of the data filter. 0x0FF represents an oversampling ratio of 256. Reset type: SYSRSn

28.12.2.36 SDDPARM3 Register (Offset = 32h) [Reset = 0h]

SDDPARM3 is shown in Figure 28-50 and described in Table 28-47.

Return to the Summary Table.

Data Parameter Register for Ch3

Figure 28-50. SDDPARM3 Register

15	14	13	12	11	10	9	8
SH					DR	RESERVED	
R/W-0h					R/W-0h	R-0-0h	
7	6	5	4	3	2	1	0
RESERVED							
R-0-0h							

Table 28-47. SDDPARM3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	SH	R/W	0h	Shift Control These bits indicate by how many bits the 16-bit window is shifted up when 16-bit data representation is chosen. Reset type: SYSRSn
10	DR	R/W	0h	Data filter Data representation 0: Data stored in 16b 2's complement 1: Data stored in 32b 2's complement Reset type: SYSRSn
9-0	RESERVED	R-0	0h	Reserved

28.12.2.37 SDFLT3CMPH1 Register (Offset = 33h) [Reset = 7FFFh]

SDFLT3CMPH1 is shown in Figure 28-51 and described in Table 28-48.

Return to the Summary Table.

High-level Threshold Register for Ch3

Figure 28-51. SDFLT3CMPH1 Register

15	14	13	12	11	10	9	8
RESERVED	HLT						
R-0-0h	R/W-7FFFh						
7	6	5	4	3	2	1	0
HLT							
R/W-7FFFh							

Table 28-48. SDFLT3CMPH1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLT	R/W	7FFFh	Unsigned high-level threshold for the comparator filter output. Reset type: SYSRSn

28.12.2.38 SDFLT3CMPL1 Register (Offset = 34h) [Reset = 0h]

SDFLT3CMPL1 is shown in Figure 28-52 and described in Table 28-49.

Return to the Summary Table.

Low-level Threshold Register for Ch3

Figure 28-52. SDFLT3CMPL1 Register

15	14	13	12	11	10	9	8
RESERVED	LLT						
R-0-0h	R/W-0h						
7	6	5	4	3	2	1	0
LLT							
R/W-0h							

Table 28-49. SDFLT3CMPL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	LLT	R/W	0h	Unsigned low-level threshold for the comparator filter output. Reset type: SYSRSn

28.12.2.39 SDCPARM3 Register (Offset = 35h) [Reset = 2000h]

SDCPARM3 is shown in Figure 28-53 and described in Table 28-50.

Return to the Summary Table.

Comparator Filter Parameter Register for Ch3

Figure 28-53. SDCPARM3 Register

15	14	13	12	11	10	9	8
CEVT2SEL		CEN	CEVT1SEL		HZEN	MFIE	CS1_CS0
R/W-0h		R/W-1h	R/W-0h		R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CS1_CS0	EN_CEVT2	EN_CEVT1	COSR				
R/W-0h	R/W-0h	R/W-0h	R/W-0h				

Table 28-50. SDCPARM3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	CEVT2SEL	R/W	0h	Comparator Event2 Select 00: COMPL1 01: COMPL1 OR COMPH1 10: COMPL2 11: COMPL2 OR COMPH2 Reset type: SYSRSn
13	CEN	R/W	1h	Comparator Filter enable 0: Disable comparator filter 1: Enable comparator filter Reset type: SYSRSn
12-11	CEVT1SEL	R/W	0h	Comparator Event1 Select 00: COMPH1 01: COMPL1 OR COMPH1 10: COMPH2 11: COMPL2 OR COMPH2 Reset type: SYSRSn
10	HZEN	R/W	0h	High level (Z) Threshold crossing output enable 0: Disable Higher level Threshold (Z) crossing 1: Enable Higher level Threhold (Z) crossing Reset type: SYSRSn
9	MFIE	R/W	0h	Modulator Failure Interrupt Enable 0: Disable modulator failure interrupt and its flag 1: Enable modulator failure interrupt and its flag Reset type: SYSRSn
8-7	CS1_CS0	R/W	0h	Comparator filter structure 00: Comparator filter runs with a sincfast structure 01: Comparator filter runs with a Sinc1 structure 10: Comparator filter runs with a Sinc2 structure 11: Comparator filter runs with a Sinc3 structure Reset type: SYSRSn
6	EN_CEVT2	R/W	0h	CEVT2 interrupt enable 0: Disable CEVT2 interrupt 1: Enable CEVT2 interrupt Reset type: SYSRSn
5	EN_CEVT1	R/W	0h	CEVT1 interrupt enable 0: Disable CEVT1 interrupt 1: Enable CEVT1 interrupt Reset type: SYSRSn

Table 28-50. SDCPARM3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	COSR	R/W	0h	Comparator Oversampling ratio. The actual rate is COSR + 1. These bits set the oversampling ratio of the filter. 0x1F represents an oversampling ratio of 32 Reset type: SYSRSn

28.12.2.40 SDDATA3 Register (Offset = 36h) [Reset = 0h]

SDDATA3 is shown in Figure 28-54 and described in Table 28-51.

Return to the Summary Table.

Data Filter Data Register (16 or 32bit) for Ch3

Figure 28-54. SDDATA3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA32HI																DATA16															
R-0h																R-0h															

Table 28-51. SDDATA3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA32HI	R	0h	Hi-order 16b in 32b mode, 16-bit Data in 16b mode Reset type: SYSRSn
15-0	DATA16	R	0h	Lo-order 16b in 32b mode Reset type: SYSRSn

28.12.2.41 SDDATFIFO3 Register (Offset = 38h) [Reset = 0h]

SDDATFIFO3 is shown in Figure 28-55 and described in Table 28-52.

Return to the Summary Table.

Filter Data FIFO Output(32b) for Ch3

Figure 28-55. SDDATFIFO3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA32HI																DATA16															
R-0h																R-0h															

Table 28-52. SDDATFIFO3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA32HI	R	0h	Hi-order 16b in 32b mode, 16-bit Data in 16b mode Reset type: SYSRSn
15-0	DATA16	R	0h	Lo-order 16b in 32b mode Reset type: SYSRSn

28.12.2.42 SDCDATA3 Register (Offset = 3Ah) [Reset = 0h]

SDCDATA3 is shown in Figure 28-56 and described in Table 28-53.

Return to the Summary Table.

Comparator Filter Data Register (16b) for Ch3

Figure 28-56. SDCDATA3 Register

15	14	13	12	11	10	9	8
DATA16							
R-0h							
7	6	5	4	3	2	1	0
DATA16							
R-0h							

Table 28-53. SDCDATA3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DATA16	R	0h	Comparator Data output - 16b only Reset type: SYSRSn

28.12.2.43 SDFLT3CMPH2 Register (Offset = 3Bh) [Reset = 7FFFh]

SDFLT3CMPH2 is shown in Figure 28-57 and described in Table 28-54.

Return to the Summary Table.

Second high level threhold for CH3

Figure 28-57. SDFLT3CMPH2 Register

15	14	13	12	11	10	9	8
RESERVED	HLT2						
R-0-0h	R/W-7FFFh						
7	6	5	4	3	2	1	0
HLT2							
R/W-7FFFh							

Table 28-54. SDFLT3CMPH2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLT2	R/W	7FFFh	Second Unsigned high-level threshold for the comparator filter output. Reset type: SYSRSn

28.12.2.44 SDFLT3CMPHZ Register (Offset = 3Ch) [Reset = 0h]

SDFLT3CMPHZ is shown in Figure 28-58 and described in Table 28-55.

Return to the Summary Table.

High-level (Z) Threshold Register for Ch3

Figure 28-58. SDFLT3CMPHZ Register

15	14	13	12	11	10	9	8
RESERVED	HLTZ						
R-0-0h	R/W-0h						
7	6	5	4	3	2	1	0
HLTZ							
R/W-0h							

Table 28-55. SDFLT3CMPHZ Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLTZ	R/W	0h	Unsigned High-level threshold (Z) for the comparator filter output Primarily intended for detecting "zero"-crossing events. Unlike the primary comparator SDCMPHx, it does not have the ability to generate an interrupt. Reset type: SYSRSn

28.12.2.45 SDFIFOCTL3 Register (Offset = 3Dh) [Reset = 0h]

SDFIFOCTL3 is shown in Figure 28-59 and described in Table 28-56.

Return to the Summary Table.

FIFO Control Register for Ch3

Figure 28-59. SDFIFOCTL3 Register

15	14	13	12	11	10	9	8
OVFIEN	DRINTSEL	FFEN	FFIEN	RESERVED	SDFFST		
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0-0h	R-0h		
7	6	5	4	3	2	1	0
SDFFST		RESERVED	SDFFIL				
R-0h		R-0-0h	R/W-0h				

Table 28-56. SDFIFOCTL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	OVFIEN	R/W	0h	SDFIFO Overflow interrupt enable 0: SDFIFO Overflow condition will not generate an interrupt 1: SDFIFO overflow condition generates an interrupt on SDy_ERR Reset type: SYSRSn
14	DRINTSEL	R/W	0h	Data-Ready Interrupt (DRINT) source select 0 = AF3 (Select non-FIFO data-ready interrupt) 1 = SDFFINT3 (Select FIFO data-ready interrupt) Reset type: SYSRSn
13	FFEN	R/W	0h	SDFIFO Enable 0: Disable FIFO operation 1: Enable FIFO operation Note: When FIFO is disabled, FIFO contents are cleared Reset type: SYSRSn
12	FFIEN	R/W	0h	SDFIFO data ready Interrupt Enable Reset type: SYSRSn
11	RESERVED	R-0	0h	Reserved
10-6	SDFFST	R	0h	SDFIFO Status 00000 FIFO empty 00001 FIFO has 1 word . . . . 10000 FIFO has 16 words Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4-0	SDFFIL	R/W	0h	SDFIFO interrupt level bits The FIFO will generate an interrupt when the FIFO status (SDFFST) >= FIFO level (SDFFIL) Reset type: SYSRSn

28.12.2.46 SDSYNC3 Register (Offset = 3Eh) [Reset = 43Fh]

SDSYNC3 is shown in Figure 28-60 and described in Table 28-57.

Return to the Summary Table.

SD Filter Sync control for Ch3

Figure 28-60. SDSYNC3 Register

15	14	13	12	11	10	9	8
RESERVED					WTSCLREN	FFSYNCCLREN	WTSYNCLR
R-0-0h					R/W-1h	R/W-0h	R-0/W-0h
7	6	5	4	3	2	1	0
WTSYNFLG	WTSYNCEN	SYNCSEL					
R-0h	R/W-0h	R/W-3Fh					

Table 28-57. SDSYNC3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R-0	0h	Reserved
10	WTSCLREN	R/W	1h	WTSYNFLG Clear-on-FIFOINT Enable 0: WTSYNFLG can only be cleared manually (using WTSYNCLR bit) 1: WTSYNFLG is cleared automatically on SDFFINT Reset type: SYSRSn
9	FFSYNCCLREN	R/W	0h	FIFO Clear-on-SDSYNC Enable 0: SDFIFO is not automaticaly cleared upon receiving SDSYNC 1: SDFIFO is automaticaly cleared upon receiving SDSYNC Reset type: SYSRSn
8	WTSYNCLR	R-0/W	0h	Wait-for-Sync Flag Clear (always reads 0) 0: Write of 0 has no affect 1: Write of 1 clears WTSYNFLG Reset type: SYSRSn
7	WTSYNFLG	R	0h	Wait-for-Sync Flag 0: SDSYNC event has not occurred 1: SDSYNC event occurred. Reset type: SYSRSn
6	WTSYNCEN	R/W	0h	Wait-for-Sync Enable 0: Incoming Data written to SDFIFO on every Data-Ready (DR) Event 1: Incoming Data written to SDFIFO on DR event only after SDSYNC event occurs Reset type: SYSRSn
5-0	SYNCSEL	R/W	3Fh	Defines source for the SDSYNC Input on this channel Refer SDSYNCx.SYNCSEL table Reset type: SYSRSn

28.12.2.47 SDFLT3CMPL2 Register (Offset = 3Fh) [Reset = 0h]

SDFLT3CMPL2 is shown in Figure 28-61 and described in Table 28-58.

Return to the Summary Table.

Second low level threhold for CH3

Figure 28-61. SDFLT3CMPL2 Register

15	14	13	12	11	10	9	8
RESERVED	LLT2						
R-0-0h	R/W-0h						
7	6	5	4	3	2	1	0
LLT2							
R/W-0h							

Table 28-58. SDFLT3CMPL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	LLT2	R/W	0h	Second Unsigned low-level threshold for the comparator filter output. Reset type: SYSRSn

28.12.2.48 SDCTLPARM4 Register (Offset = 40h) [Reset = 0h]

SDCTLPARM4 is shown in Figure 28-62 and described in Table 28-59.

Return to the Summary Table.

Control Parameter Register for Ch4

Figure 28-62. SDCTLPARM4 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	SDDATASYNC	RESERVED	SDCLKSYNC	SDCLKSEL	RESERVED	MOD	
R-0-0h	R/W-0h	R-0-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	

Table 28-59. SDCTLPARM4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	RESERVED	R-0	0h	Reserved
6	SDDATASYNC	R/W	0h	0: SD Data is not passed through a synchronizer. 1: SD Data is passed through a synchronizer. Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4	SDCLKSYNC	R/W	0h	0: SD Clock is not passed through a synchronizer. 1: SD Clock is passed through a synchronizer. Reset type: SYSRSn
3	SDCLKSEL	R/W	0h	SD4 Clock source select. 0: Clock source to SDFM filter is its channel clock. 1: Clock source to SDFM filter is SD1 filter clock. Reset type: SYSRSn
2	RESERVED	R/W	0h	Reserved
1-0	MOD	R/W	0h	Modulator clock modes 0: Mode 0: Modulator clock running at 1x data rate 1: Reserved 2: Reserved 3: Reserved Reset type: SYSRSn

28.12.2.49 SDDFPARM4 Register (Offset = 41h) [Reset = 0h]

SDDFPARM4 is shown in Figure 28-63 and described in Table 28-60.

Return to the Summary Table.

Data Filter Parameter Register for Ch4

Figure 28-63. SDDFPARM4 Register

15	14	13	12	11	10	9	8
RESERVED			SDSYNCEN	SST		AE	FEN
R-0h			R/W-0h	R/W-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DOSR							
R/W-0h							

Table 28-60. SDDFPARM4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12	SDSYNCEN	R/W	0h	PWM synchronization (SDSYNC) of data filter 0: PWM synchronization of data filter is disabled 1: PWM synchronization of data filter is enabled Note: SDSYNCx.SYNCSEL bits define which PWM signal is used to synchronize PWMs Reset type: SYSRSn
11-10	SST	R/W	0h	Data filter structure 00: Data filter runs with a Sincfast structure 01: Data filter runs with a Sinc1 structure 10: Data filter runs with a Sinc2 structure 11: Data filter runs with a Sinc3 structure Reset type: SYSRSn
9	AE	R/W	0h	Data filter Acknowledge Enable 0: Acknowledge flag is disabled for the particular filter 1: Acknowledge flag is enabled for the particular filter Reset type: SYSRSn
8	FEN	R/W	0h	Filter Enable 0: The data filter is disabled and no data is produced 1: The data filter is enabled and data are produced in the data filter Note: When filter is disabled, DOSR counter held in reset, filter data erased. Also resets FIFO pointers and clears the FIFO Reset type: SYSRSn
7-0	DOSR	R/W	0h	Data filter Oversampling ratio The actual oversampling ratio of data filter is DOSR + 1 These bits set the oversampling ratio of the data filter. 0x0FF represents an oversampling ratio of 256. Reset type: SYSRSn

28.12.2.50 SDDPARM4 Register (Offset = 42h) [Reset = 0h]

SDDPARM4 is shown in Figure 28-64 and described in Table 28-61.

Return to the Summary Table.

Data Parameter Register for Ch4

Figure 28-64. SDDPARM4 Register

15	14	13	12	11	10	9	8
SH					DR	RESERVED	
R/W-0h					R/W-0h	R-0-0h	
7	6	5	4	3	2	1	0
RESERVED							
R-0-0h							

Table 28-61. SDDPARM4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	SH	R/W	0h	Shift Control These bits indicate by how many bits the 16-bit window is shifted up when 16-bit data representation is chosen. Reset type: SYSRSn
10	DR	R/W	0h	Data filter Data representation 0: Data stored in 16b 2's complement 1: Data stored in 32b 2's complement Reset type: SYSRSn
9-0	RESERVED	R-0	0h	Reserved

28.12.2.51 SDFLT4CMPH1 Register (Offset = 43h) [Reset = 7FFFh]

SDFLT4CMPH1 is shown in Figure 28-65 and described in Table 28-62.

Return to the Summary Table.

High-level Threshold Register for Ch4

Figure 28-65. SDFLT4CMPH1 Register

15	14	13	12	11	10	9	8
RESERVED	HLT						
R-0-0h	R/W-7FFFh						
7	6	5	4	3	2	1	0
HLT							
R/W-7FFFh							

Table 28-62. SDFLT4CMPH1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLT	R/W	7FFFh	Unsigned high-level threshold for the comparator filter output. Reset type: SYSRSn

28.12.2.52 SDFLT4CMPL1 Register (Offset = 44h) [Reset = 0h]

SDFLT4CMPL1 is shown in Figure 28-66 and described in Table 28-63.

Return to the Summary Table.

Low-level Threshold Register for Ch4

Figure 28-66. SDFLT4CMPL1 Register

15	14	13	12	11	10	9	8
RESERVED	LLT						
R-0-0h	R/W-0h						
7	6	5	4	3	2	1	0
LLT							
R/W-0h							

Table 28-63. SDFLT4CMPL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	LLT	R/W	0h	Unsigned low-level threshold for the comparator filter output. Reset type: SYSRSn

28.12.2.53 SDCPARM4 Register (Offset = 45h) [Reset = 2000h]

SDCPARM4 is shown in Figure 28-67 and described in Table 28-64.

Return to the Summary Table.

Comparator Filter Parameter Register for Ch4

Figure 28-67. SDCPARM4 Register

15	14	13	12	11	10	9	8
CEVT2SEL		CEN	CEVT1SEL		HZEN	MFIE	CS1_CS0
R/W-0h		R/W-1h	R/W-0h		R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CS1_CS0	EN_CEVT2	EN_CEVT1	COSR				
R/W-0h	R/W-0h	R/W-0h	R/W-0h				

Table 28-64. SDCPARM4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	CEVT2SEL	R/W	0h	Comparator Event2 Select 00: COMPL1 01: COMPL1 OR COMPH1 10: COMPL2 11: COMPL2 OR COMPH2 Reset type: SYSRSn
13	CEN	R/W	1h	Comparator Filter enable 0: Disable comparator filter 1: Enable comparator filter Reset type: SYSRSn
12-11	CEVT1SEL	R/W	0h	Comparator Event1 Select 00: COMPH1 01: COMPL1 OR COMPH1 10: COMPH2 11: COMPL2 OR COMPH2 Reset type: SYSRSn
10	HZEN	R/W	0h	High level (Z) Threshold crossing output enable 0: Disable Higher level Threshold (Z) crossing 1: Enable Higher level Threhold (Z) crossing Reset type: SYSRSn
9	MFIE	R/W	0h	Modulator Failure Interrupt Enable 0: Disable modulator failure interrupt and its flag 1: Enable modulator failure interrupt and its flag Reset type: SYSRSn
8-7	CS1_CS0	R/W	0h	Comparator filter structure 00: Comparator filter runs with a sincfast structure 01: Comparator filter runs with a Sinc1 structure 10: Comparator filter runs with a Sinc2 structure 11: Comparator filter runs with a Sinc3 structure Reset type: SYSRSn
6	EN_CEVT2	R/W	0h	CEVT2 interrupt enable 0: Disable CEVT2 interrupt 1: Enable CEVT2 interrupt Reset type: SYSRSn
5	EN_CEVT1	R/W	0h	CEVT1 interrupt enable 0: Disable CEVT1 interrupt 1: Enable CEVT1 interrupt Reset type: SYSRSn

Table 28-64. SDCPARM4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	COSR	R/W	0h	Comparator Oversampling ratio. The actual rate is COSR + 1. These bits set the oversampling ratio of the filter. 0x1F represents an oversampling ratio of 32 Reset type: SYSRSn

28.12.2.54 SDDATA4 Register (Offset = 46h) [Reset = 0h]

SDDATA4 is shown in Figure 28-68 and described in Table 28-65.

Return to the Summary Table.

Data Filter Data Register (16 or 32bit) for Ch4

Figure 28-68. SDDATA4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA32HI																DATA16															
R-0h																R-0h															

Table 28-65. SDDATA4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA32HI	R	0h	Hi-order 16b in 32b mode, 16-bit Data in 16b mode Reset type: SYSRSn
15-0	DATA16	R	0h	Lo-order 16b in 32b mode Reset type: SYSRSn

28.12.2.55 SDDATFIFO4 Register (Offset = 48h) [Reset = 0h]

SDDATFIFO4 is shown in Figure 28-69 and described in Table 28-66.

Return to the Summary Table.

Filter Data FIFO Output(32b) for Ch4

Figure 28-69. SDDATFIFO4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA32HI																DATA16															
R-0h																R-0h															

Table 28-66. SDDATFIFO4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA32HI	R	0h	Hi-order 16b in 32b mode, 16-bit Data in 16b mode Reset type: SYSRSn
15-0	DATA16	R	0h	Lo-order 16b in 32b mode Reset type: SYSRSn

28.12.2.56 SDCDATA4 Register (Offset = 4Ah) [Reset = 0h]

SDCDATA4 is shown in Figure 28-70 and described in Table 28-67.

Return to the Summary Table.

Comparator Filter Data Register (16b) for Ch4

Figure 28-70. SDCDATA4 Register

15	14	13	12	11	10	9	8
DATA16							
R-0h							
7	6	5	4	3	2	1	0
DATA16							
R-0h							

Table 28-67. SDCDATA4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DATA16	R	0h	Comparator Data output - 16b only Reset type: SYSRSn

28.12.2.57 SDFLT4CMPH2 Register (Offset = 4Bh) [Reset = 7FFFh]

SDFLT4CMPH2 is shown in Figure 28-71 and described in Table 28-68.

Return to the Summary Table.

Second high level threhold for CH4

Figure 28-71. SDFLT4CMPH2 Register

15	14	13	12	11	10	9	8
RESERVED	HLT2						
R-0-0h	R/W-7FFFh						
7	6	5	4	3	2	1	0
HLT2							
R/W-7FFFh							

Table 28-68. SDFLT4CMPH2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLT2	R/W	7FFFh	Second Unsigned high-level threshold for the comparator filter output. Reset type: SYSRSn

28.12.2.58 SDFLT4CMPHZ Register (Offset = 4Ch) [Reset = 0h]

SDFLT4CMPHZ is shown in Figure 28-72 and described in Table 28-69.

Return to the Summary Table.

High-level (Z) Threshold Register for Ch4

Figure 28-72. SDFLT4CMPHZ Register

15	14	13	12	11	10	9	8
RESERVED	HLTZ						
R-0-0h	R/W-0h						
7	6	5	4	3	2	1	0
HLTZ							
R/W-0h							

Table 28-69. SDFLT4CMPHZ Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLTZ	R/W	0h	Unsigned High-level threshold (Z) for the comparator filter output Primarily intended for detecting "zero"-crossing events. Unlike the primary comparator SDCMPHx, it does not have the ability to generate an interrupt. Reset type: SYSRSn

28.12.2.59 SDFIFOCTL4 Register (Offset = 4Dh) [Reset = 0h]

SDFIFOCTL4 is shown in Figure 28-73 and described in Table 28-70.

Return to the Summary Table.

FIFO Control Register for Ch4

Figure 28-73. SDFIFOCTL4 Register

15	14	13	12	11	10	9	8
OVFIEN	DRINTSEL	FFEN	FFIEN	RESERVED	SDFFST		
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0-0h	R-0h		
7	6	5	4	3	2	1	0
SDFFST		RESERVED	SDFFIL				
R-0h		R-0-0h	R/W-0h				

Table 28-70. SDFIFOCTL4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	OVFIEN	R/W	0h	SDFIFO Overflow interrupt enable 0: SDFIFO Overflow condition will not generate an interrupt 1: SDFIFO overflow condition generates an interrupt on SDy_ERR Reset type: SYSRSn
14	DRINTSEL	R/W	0h	Data-Ready Interrupt (DRINT) source select 0 = AF4 (Select non-FIFO data-ready interrupt) 1 = SDFFINT4 (Select FIFO data-ready interrupt) Reset type: SYSRSn
13	FFEN	R/W	0h	SDFIFO Enable 0: Disable FIFO operation 1: Enable FIFO operation Note: When FIFO is disabled, FIFO contents are cleared Reset type: SYSRSn
12	FFIEN	R/W	0h	SDFIFO data ready Interrupt Enable Reset type: SYSRSn
11	RESERVED	R-0	0h	Reserved
10-6	SDFFST	R	0h	SDFIFO Status 00000 FIFO empty 00001 FIFO has 1 word . . . . 10000 FIFO has 16 words Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4-0	SDFFIL	R/W	0h	SDFIFO interrupt level bits The FIFO will generate an interrupt when the FIFO status (SDFFST) >= FIFO level (SDFFIL) Reset type: SYSRSn

28.12.2.60 SDSYNC4 Register (Offset = 4Eh) [Reset = 43Fh]

SDSYNC4 is shown in Figure 28-74 and described in Table 28-71.

Return to the Summary Table.

SD Filter Sync control for Ch4

Figure 28-74. SDSYNC4 Register

15	14	13	12	11	10	9	8
RESERVED					WTSCLREN	FFSYNCCLREN	WTSYNCLR
R-0-0h					R/W-1h	R/W-0h	R-0/W-0h
7	6	5	4	3	2	1	0
WTSYNFLG	WTSYNCEN	SYNCSEL					
R-0h	R/W-0h	R/W-3Fh					

Table 28-71. SDSYNC4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R-0	0h	Reserved
10	WTSCLREN	R/W	1h	WTSYNFLG Clear-on-FIFOINT Enable 0: WTSYNFLG can only be cleared manually (using WTSYNCLR bit) 1: WTSYNFLG is cleared automatically on SDFFINT Reset type: SYSRSn
9	FFSYNCCLREN	R/W	0h	FIFO Clear-on-SDSYNC Enable 0: SDFIFO is not automaticaly cleared upon receiving SDSYNC 1: SDFIFO is automaticaly cleared upon receiving SDSYNC Reset type: SYSRSn
8	WTSYNCLR	R-0/W	0h	Wait-for-Sync Flag Clear (always reads 0) 0: Write of 0 has no affect 1: Write of 1 clears WTSYNFLG Reset type: SYSRSn
7	WTSYNFLG	R	0h	Wait-for-Sync Flag 0: SDSYNC event has not occurred 1: SDSYNC event occurred. Reset type: SYSRSn
6	WTSYNCEN	R/W	0h	Wait-for-Sync Enable 0: Incoming Data written to SDFIFO on every Data-Ready (DR) Event 1: Incoming Data written to SDFIFO on DR event only after SDSYNC event occurs Reset type: SYSRSn
5-0	SYNCSEL	R/W	3Fh	Defines source for the SDSYNC Input on this channel Refer SDSYNCx.SYNCSEL table Reset type: SYSRSn

28.12.2.61 SDFLT4CMPL2 Register (Offset = 4Fh) [Reset = 0h]

SDFLT4CMPL2 is shown in Figure 28-75 and described in Table 28-72.

Return to the Summary Table.

Second low level threhold for CH4

Figure 28-75. SDFLT4CMPL2 Register

15	14	13	12	11	10	9	8
RESERVED	LLT2						
R-0-0h	R/W-0h						
7	6	5	4	3	2	1	0
LLT2							
R/W-0h							

Table 28-72. SDFLT4CMPL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	LLT2	R/W	0h	Second Unsigned low-level threshold for the comparator filter output. Reset type: SYSRSn

28.12.2.62 SDCOMP1CTL Register (Offset = 60h) [Reset = 0h]

SDCOMP1CTL is shown in Figure 28-76 and described in Table 28-73.

Return to the Summary Table.

SD Comparator event filter1 Control Register

Figure 28-76. SDCOMP1CTL Register

15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED		CEVT2DIGFILTSEL		RESERVED	RESERVED
R-0h	R-0h	R-0h		R/W-0h		R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED		CEVT1DIGFILTSEL		RESERVED	RESERVED
R-0h	R-0h	R-0h		R/W-0h		R-0h	R-0h

Table 28-73. SDCOMP1CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	RESERVED	R	0h	Reserved
13-12	RESERVED	R	0h	Reserved
11-10	CEVT2DIGFILTSEL	R/W	0h	High comparator COMPH source select. 0 CEVT2 output drives COMPLOUT 1 Reserved 2 Output of digital filter drives COMPLOUT 3 Reserved Reset type: SYSRSn
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	RESERVED	R	0h	Reserved
6	RESERVED	R	0h	Reserved
5-4	RESERVED	R	0h	Reserved
3-2	CEVT1DIGFILTSEL	R/W	0h	High comparator COMPH source select. 0 CEVT1 output drives COMPHOUT 1 Reserved 2 Output of digital filter drives COMPHOUT 3 Reserved Reset type: SYSRSn
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

28.12.2.63 SDCOMP1EVT2FLTCTL Register (Offset = 61h) [Reset = 0h]

SDCOMP1EVT2FLTCTL is shown in Figure 28-77 and described in Table 28-74.

Return to the Summary Table.

COMPL/CEVT2 Digital filter1 Control Register

Figure 28-77. SDCOMP1EVT2FLTCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 28-74. SDCOMP1EVT2FLTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	Low filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	Low filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	Low filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

28.12.2.64 SDCOMP1EVT2FLTCLKCTL Register (Offset = 62h) [Reset = 0h]

SDCOMP1EVT2FLTCLKCTL is shown in Figure 28-78 and described in Table 28-75.

Return to the Summary Table.

COMPL/CEVT2 Digital filter1 Clock Control Register

Figure 28-78. SDCOMP1EVT2FLTCLKCTL Register

15	14	13	12	11	10	9	8
RESERVED						CLKPRESCALE	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 28-75. SDCOMP1EVT2FLTCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	CLKPRESCALE	R/W	0h	Low filter sample clock prescale. Number of system clocks between samples. Reset type: SYSRSn

28.12.2.65 SDCOMP1EVT1FLTCTL Register (Offset = 63h) [Reset = 0h]

SDCOMP1EVT1FLTCTL is shown in Figure 28-79 and described in Table 28-76.

Return to the Summary Table.

COMPH/CEVT1 Digital filter1 Control Register

Figure 28-79. SDCOMP1EVT1FLTCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 28-76. SDCOMP1EVT1FLTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	High filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	High filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	High filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

28.12.2.66 SDCOMP1EVT1FLTCLKCTL Register (Offset = 64h) [Reset = 0h]

SDCOMP1EVT1FLTCLKCTL is shown in Figure 28-80 and described in Table 28-77.

Return to the Summary Table.

COMPH/CEVT1 Digital filter1 Clock Control Register

Figure 28-80. SDCOMP1EVT1FLTCLKCTL Register

15	14	13	12	11	10	9	8
RESERVED						CLKPRESCALE	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 28-77. SDCOMP1EVT1FLTCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	CLKPRESCALE	R/W	0h	High filter sample clock prescale. Number of system clocks between samples. Reset type: SYSRSn

28.12.2.67 SDCOMP1LOCK Register (Offset = 67h) [Reset = 0h]

SDCOMP1LOCK is shown in Figure 28-81 and described in Table 28-78.

Return to the Summary Table.

SD compartor event filter1 Lock Register

Figure 28-81. SDCOMP1LOCK Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	COMP	RESERVED	RESERVED	SDCOMP1CTL
R-0h			R-0h	R/WSonce-0h	R-0h	R-0h	R/WSonce-0h

Table 28-78. SDCOMP1LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	COMP	R/WSonce	0h	Lock write-access to the SDCOMP1EVT1/2FLTTCTL and COMP1FILCLKCTL registers. 0 SDCOMP1EVT1/2FLTCTL and SDCOMP1EVT1/2FLTCLKCTL registers are not locked. Write 0 to this bit has no effect. 1 SDCOMP1EVT1/2FLTCTL and SDCOMP1EVT1/2FLTCLKCTL registers are locked. Only a system reset can clear this bit. Reset type: SYSRSn
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	SDCOMP1CTL	R/WSonce	0h	Lock write-access to the SDCOMP1CTL register. 0 SDCOMP1CTL register is not locked. Write 0 to this bit has no effect. 1 SDCOMP1CTL register is locked. Only a system reset can clear this bit. Reset type: SYSRSn

28.12.2.68 SDCOMP2CTL Register (Offset = 68h) [Reset = 0h]

SDCOMP2CTL is shown in Figure 28-82 and described in Table 28-79.

Return to the Summary Table.

SD Comparator event filter2 Control Register

Figure 28-82. SDCOMP2CTL Register

15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED		CEVT2DIGFILTSEL		RESERVED	RESERVED
R-0h	R-0h	R-0h		R/W-0h		R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED		CEVT1DIGFILTSEL		RESERVED	RESERVED
R-0h	R-0h	R-0h		R/W-0h		R-0h	R-0h

Table 28-79. SDCOMP2CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	RESERVED	R	0h	Reserved
13-12	RESERVED	R	0h	Reserved
11-10	CEVT2DIGFILTSEL	R/W	0h	High comparator COMPH source select. 0 CEVT2 output drives COMPLOUT 1 Reserved 2 Output of digital filter drives COMPLOUT 3 Reserved Reset type: SYSRSn
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	RESERVED	R	0h	Reserved
6	RESERVED	R	0h	Reserved
5-4	RESERVED	R	0h	Reserved
3-2	CEVT1DIGFILTSEL	R/W	0h	High comparator COMPH source select. 0 CEVT1 output drives COMPHOUT 1 Reserved 2 Output of digital filter drives COMPHOUT 3 Reserved Reset type: SYSRSn
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

28.12.2.69 SDCOMP2EVT2FLTCTL Register (Offset = 69h) [Reset = 0h]

SDCOMP2EVT2FLTCTL is shown in Figure 28-83 and described in Table 28-80.

Return to the Summary Table.

COMPL/CEVT2 Digital filter2 Control Register

Figure 28-83. SDCOMP2EVT2FLTCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 28-80. SDCOMP2EVT2FLTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	Low filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	Low filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	Low filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

28.12.2.70 SDCOMP2EVT2FLTCLKCTL Register (Offset = 6Ah) [Reset = 0h]

SDCOMP2EVT2FLTCLKCTL is shown in Figure 28-84 and described in Table 28-81.

Return to the Summary Table.

COMPL/CEVT2 Digital filter2 Clock Control Register

Figure 28-84. SDCOMP2EVT2FLTCLKCTL Register

15	14	13	12	11	10	9	8
RESERVED						CLKPRESCALE	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 28-81. SDCOMP2EVT2FLTCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	CLKPRESCALE	R/W	0h	Low filter sample clock prescale. Number of system clocks between samples. Reset type: SYSRSn

28.12.2.71 SDCOMP2EVT1FLTCTL Register (Offset = 6Bh) [Reset = 0h]

SDCOMP2EVT1FLTCTL is shown in Figure 28-85 and described in Table 28-82.

Return to the Summary Table.

COMPH/CEVT1 Digital filter2 Control Register

Figure 28-85. SDCOMP2EVT1FLTCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 28-82. SDCOMP2EVT1FLTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	High filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	High filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	High filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

28.12.2.72 SDCOMP2EVT1FLTCLKCTL Register (Offset = 6Ch) [Reset = 0h]

SDCOMP2EVT1FLTCLKCTL is shown in Figure 28-86 and described in Table 28-83.

Return to the Summary Table.

COMPH/CEVT1 Digital filter2 Clock Control Register

Figure 28-86. SDCOMP2EVT1FLTCLKCTL Register

15	14	13	12	11	10	9	8
RESERVED						CLKPRESCALE	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 28-83. SDCOMP2EVT1FLTCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	CLKPRESCALE	R/W	0h	High filter sample clock prescale. Number of system clocks between samples. Reset type: SYSRSn

28.12.2.73 SDCOMP2LOCK Register (Offset = 6Fh) [Reset = 0h]

SDCOMP2LOCK is shown in Figure 28-87 and described in Table 28-84.

Return to the Summary Table.

SD compartor event filter2 Lock Register

Figure 28-87. SDCOMP2LOCK Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	COMP	RESERVED	RESERVED	SDCOMP2CTL
R-0h			R-0h	R/WSonce-0h	R-0h	R-0h	R/WSonce-0h

Table 28-84. SDCOMP2LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	COMP	R/WSonce	0h	Lock write-access to the SDCOMP2EVT1/2FLTTCTL and COMP2FILCLKCTL registers. 0 SDCOMP2EVT1/2FLTCTL and SDCOMP2EVT1/2FLTCLKCTL registers are not locked. Write 0 to this bit has no effect. 1 SDCOMP2EVT1/2FLTCTL and SDCOMP2EVT1/2FLTCLKCTL registers are locked. Only a system reset can clear this bit. Reset type: SYSRSn
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	SDCOMP2CTL	R/WSonce	0h	Lock write-access to the SDCOMP2CTL register. 0 SDCOMP2CTL register is not locked. Write 0 to this bit has no effect. 1 SDCOMP2CTL register is locked. Only a system reset can clear this bit. Reset type: SYSRSn

28.12.2.74 SDCOMP3CTL Register (Offset = 70h) [Reset = 0h]

SDCOMP3CTL is shown in Figure 28-88 and described in Table 28-85.

Return to the Summary Table.

SD Comparator event filter3 Control Register

Figure 28-88. SDCOMP3CTL Register

15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED		CEVT2DIGFILTSEL		RESERVED	RESERVED
R-0h	R-0h	R-0h		R/W-0h		R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED		CEVT1DIGFILTSEL		RESERVED	RESERVED
R-0h	R-0h	R-0h		R/W-0h		R-0h	R-0h

Table 28-85. SDCOMP3CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	RESERVED	R	0h	Reserved
13-12	RESERVED	R	0h	Reserved
11-10	CEVT2DIGFILTSEL	R/W	0h	High comparator COMPH source select. 0 CEVT2 output drives COMPLOUT 1 Reserved 2 Output of digital filter drives COMPLOUT 3 Reserved Reset type: SYSRSn
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	RESERVED	R	0h	Reserved
6	RESERVED	R	0h	Reserved
5-4	RESERVED	R	0h	Reserved
3-2	CEVT1DIGFILTSEL	R/W	0h	High comparator COMPH source select. 0 CEVT1 output drives COMPHOUT 1 Reserved 2 Output of digital filter drives COMPHOUT 3 Reserved Reset type: SYSRSn
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

28.12.2.75 SDCOMP3EVT2FLTCTL Register (Offset = 71h) [Reset = 0h]

SDCOMP3EVT2FLTCTL is shown in Figure 28-89 and described in Table 28-86.

Return to the Summary Table.

COMPL/CEVT2 Digital filter3 Control Register

Figure 28-89. SDCOMP3EVT2FLTCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 28-86. SDCOMP3EVT2FLTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	Low filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	Low filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	Low filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

28.12.2.76 SDCOMP3EVT2FLTCLKCTL Register (Offset = 72h) [Reset = 0h]

SDCOMP3EVT2FLTCLKCTL is shown in Figure 28-90 and described in Table 28-87.

Return to the Summary Table.

COMPL/CEVT2 Digital filter3 Clock Control Register

Figure 28-90. SDCOMP3EVT2FLTCLKCTL Register

15	14	13	12	11	10	9	8
RESERVED						CLKPRESCALE	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 28-87. SDCOMP3EVT2FLTCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	CLKPRESCALE	R/W	0h	Low filter sample clock prescale. Number of system clocks between samples. Reset type: SYSRSn

28.12.2.77 SDCOMP3EVT1FLTCTL Register (Offset = 73h) [Reset = 0h]

SDCOMP3EVT1FLTCTL is shown in Figure 28-91 and described in Table 28-88.

Return to the Summary Table.

COMPH/CEVT1 Digital filter3 Control Register

Figure 28-91. SDCOMP3EVT1FLTCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 28-88. SDCOMP3EVT1FLTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	High filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	High filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	High filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

28.12.2.78 SDCOMP3EVT1FLTCLKCTL Register (Offset = 74h) [Reset = 0h]

SDCOMP3EVT1FLTCLKCTL is shown in Figure 28-92 and described in Table 28-89.

Return to the Summary Table.

COMPH/CEVT1 Digital filter3 Clock Control Register

Figure 28-92. SDCOMP3EVT1FLTCLKCTL Register

15	14	13	12	11	10	9	8
RESERVED						CLKPRESCALE	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 28-89. SDCOMP3EVT1FLTCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	CLKPRESCALE	R/W	0h	High filter sample clock prescale. Number of system clocks between samples. Reset type: SYSRSn

28.12.2.79 SDCOMP3LOCK Register (Offset = 77h) [Reset = 0h]

SDCOMP3LOCK is shown in Figure 28-93 and described in Table 28-90.

Return to the Summary Table.

SD compartor event filter3 Lock Register

Figure 28-93. SDCOMP3LOCK Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	COMP	RESERVED	RESERVED	SDCOMP3CTL
R-0h			R-0h	R/WSonce-0h	R-0h	R-0h	R/WSonce-0h

Table 28-90. SDCOMP3LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	COMP	R/WSonce	0h	Lock write-access to the SDCOMP3EVT1/2FLTTCTL and COMP3FILCLKCTL registers. 0 SDCOMP3EVT1/2FLTCTL and SDCOMP3EVT1/2FLTCLKCTL registers are not locked. Write 0 to this bit has no effect. 1 SDCOMP3EVT1/2FLTCTL and SDCOMP3EVT1/2FLTCLKCTL registers are locked. Only a system reset can clear this bit. Reset type: SYSRSn
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	SDCOMP3CTL	R/WSonce	0h	Lock write-access to the SDCOMP3CTL register. 0 SDCOMP3CTL register is not locked. Write 0 to this bit has no effect. 1 SDCOMP3CTL register is locked. Only a system reset can clear this bit. Reset type: SYSRSn

28.12.2.80 SDCOMP4CTL Register (Offset = 78h) [Reset = 0h]

SDCOMP4CTL is shown in Figure 28-94 and described in Table 28-91.

Return to the Summary Table.

SD Comparator event filter4 Control Register

Figure 28-94. SDCOMP4CTL Register

15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED		CEVT2DIGFILTSEL		RESERVED	RESERVED
R-0h	R-0h	R-0h		R/W-0h		R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED		CEVT1DIGFILTSEL		RESERVED	RESERVED
R-0h	R-0h	R-0h		R/W-0h		R-0h	R-0h

Table 28-91. SDCOMP4CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	RESERVED	R	0h	Reserved
13-12	RESERVED	R	0h	Reserved
11-10	CEVT2DIGFILTSEL	R/W	0h	High comparator COMPH source select. 0 CEVT2 output drives COMPLOUT 1 Reserved 2 Output of digital filter drives COMPLOUT 3 Reserved Reset type: SYSRSn
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	RESERVED	R	0h	Reserved
6	RESERVED	R	0h	Reserved
5-4	RESERVED	R	0h	Reserved
3-2	CEVT1DIGFILTSEL	R/W	0h	High comparator COMPH source select. 0 CEVT1 output drives COMPHOUT 1 Reserved 2 Output of digital filter drives COMPHOUT 3 Reserved Reset type: SYSRSn
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

28.12.2.81 SDCOMP4EVT2FLTCTL Register (Offset = 79h) [Reset = 0h]

SDCOMP4EVT2FLTCTL is shown in Figure 28-95 and described in Table 28-92.

Return to the Summary Table.

COMPL/CEVT2 Digital filter4 Control Register

Figure 28-95. SDCOMP4EVT2FLTCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 28-92. SDCOMP4EVT2FLTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	Low filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	Low filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	Low filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

28.12.2.82 SDCOMP4EVT2FLTCLKCTL Register (Offset = 7Ah) [Reset = 0h]

SDCOMP4EVT2FLTCLKCTL is shown in Figure 28-96 and described in Table 28-93.

Return to the Summary Table.

COMPL/CEVT2 Digital filter4 Clock Control Register

Figure 28-96. SDCOMP4EVT2FLTCLKCTL Register

15	14	13	12	11	10	9	8
RESERVED						CLKPRESCALE	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 28-93. SDCOMP4EVT2FLTCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	CLKPRESCALE	R/W	0h	Low filter sample clock prescale. Number of system clocks between samples. Reset type: SYSRSn

28.12.2.83 SDCOMP4EVT1FLTCTL Register (Offset = 7Bh) [Reset = 0h]

SDCOMP4EVT1FLTCTL is shown in Figure 28-97 and described in Table 28-94.

Return to the Summary Table.

COMPH/CEVT1 Digital filter4 Control Register

Figure 28-97. SDCOMP4EVT1FLTCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 28-94. SDCOMP4EVT1FLTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	High filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	High filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	High filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

28.12.2.84 SDCOMP4EVT1FLTCLKCTL Register (Offset = 7Ch) [Reset = 0h]

SDCOMP4EVT1FLTCLKCTL is shown in Figure 28-98 and described in Table 28-95.

Return to the Summary Table.

COMPH/CEVT1 Digital filter4 Clock Control Register

Figure 28-98. SDCOMP4EVT1FLTCLKCTL Register

15	14	13	12	11	10	9	8
RESERVED						CLKPRESCALE	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 28-95. SDCOMP4EVT1FLTCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	CLKPRESCALE	R/W	0h	High filter sample clock prescale. Number of system clocks between samples. Reset type: SYSRSn

28.12.2.85 SDCOMP4LOCK Register (Offset = 7Fh) [Reset = 0h]

SDCOMP4LOCK is shown in Figure 28-99 and described in Table 28-96.

Return to the Summary Table.

SD compartor event filter4 Lock Register

Figure 28-99. SDCOMP4LOCK Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	COMP	RESERVED	RESERVED	SDCOMP4CTL
R-0h			R-0h	R/WSonce-0h	R-0h	R-0h	R/WSonce-0h

Table 28-96. SDCOMP4LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	COMP	R/WSonce	0h	Lock write-access to the SDCOMP4EVT1/2FLTTCTL and COMP4FILCLKCTL registers. 0 SDCOMP4EVT1/2FLTCTL and SDCOMP4EVT1/2FLTCLKCTL registers are not locked. Write 0 to this bit has no effect. 1 SDCOMP4EVT1/2FLTCTL and SDCOMP4EVT1/2FLTCLKCTL registers are locked. Only a system reset can clear this bit. Reset type: SYSRSn
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	SDCOMP4CTL	R/WSonce	0h	Lock write-access to the SDCOMP4CTL register. 0 SDCOMP4CTL register is not locked. Write 0 to this bit has no effect. 1 SDCOMP4CTL register is locked. Only a system reset can clear this bit. Reset type: SYSRSn

28.12.3 SDFM Registers to Driverlib Functions

Table 28-97. SDFM Registers to Driverlib Functions

File	Driverlib Function
SDIFLG	
sdfm.h	SDFM_getThresholdStatus
sdfm.h	SDFM_getModulatorStatus
sdfm.h	SDFM_getNewFilterDataStatus
sdfm.h	SDFM_getFIFOOverflowStatus
sdfm.h	SDFM_getFIFOISRStatus
sdfm.h	SDFM_getIsrStatus
sdfm.h	SDFM_clearInterruptFlag
SDIFLGCLR	
sdfm.h	SDFM_clearInterruptFlag
SDCTL	
sdfm.h	SDFM_clearZeroCrossTripStatus
sdfm.h	SDFM_setupModulatorClock
sdfm.h	SDFM_enableMainInterrupt

Table 28-97. SDFM Registers to Driverlib Functions (continued)

File	Driverlib Function
sdfm.h	SDFM_disableMainInterrupt
sdfm.h	SDFM_selectClockSource
sdfm.h	SDFM_enableSynchronizer
sdfm.h	SDFM_disableSynchronizer
SDMFILEN	
sdfm.h	SDFM_enableMainFilter
sdfm.h	SDFM_disableMainFilter
SDSTATUS	
sdfm.h	SDFM_getZeroCrossTripStatus
SDCTLPARM1	
sdfm.h	SDFM_setupModulatorClock
sdfm.h	SDFM_selectClockSource
sdfm.h	SDFM_enableSynchronizer
sdfm.h	SDFM_disableSynchronizer
SDDFPARM1	
sdfm.h	SDFM_enableExternalReset
sdfm.h	SDFM_disableExternalReset
sdfm.h	SDFM_enableFilter
sdfm.h	SDFM_disableFilter
sdfm.h	SDFM_setFilterType
sdfm.h	SDFM_setFilterOverSamplingRatio
sdfm.h	SDFM_enableInterrupt
sdfm.h	SDFM_disableInterrupt
SDDPARM1	
sdfm.h	SDFM_setOutputDataFormat
sdfm.h	SDFM_setDataShiftValue
SDFLT1CMPH1	
sdfm.h	SDFM_setCompFilterHighThreshold
SDFLT1CMPL1	
sdfm.h	SDFM_setCompFilterLowThreshold
SDCPARM1	
sdfm.h	SDFM_enableComparator
sdfm.h	SDFM_disableComparator
sdfm.h	SDFM_selectCompEventSource
sdfm.h	SDFM_enableZeroCrossEdgeDetect
sdfm.h	SDFM_disableZeroCrossEdgeDetect
sdfm.h	SDFM_enableInterrupt
sdfm.h	SDFM_disableInterrupt
sdfm.h	SDFM_setComparatorFilterType
sdfm.h	SDFM_setCompFilterOverSamplingRatio
SDDATA1	
sdfm.h	SDFM_getFilterData
SDDATFIFO1	
sdfm.h	SDFM_getFIFOData
SDCDATA1	

Table 28-97. SDFM Registers to Driverlib Functions (continued)

File	Driverlib Function
sdfm.h	SDFM_getComparatorSincData
SDFLT1CMPH2	
-	
SDFLT1CMPHZ	
sdfm.h	SDFM_setCompFilterZeroCrossThreshold
SDFIFOCTL1	
sdfm.h	SDFM_enableFIFOBuffer
sdfm.h	SDFM_disableFIFOBuffer
sdfm.h	SDFM_enableInterrupt
sdfm.h	SDFM_disableInterrupt
sdfm.h	SDFM_getFIFODataCount
sdfm.h	SDFM_setFIFOInterruptLevel
sdfm.h	SDFM_setDataReadyInterruptSource
SDSYNC1	
sdfm.h	SDFM_getWaitForSyncStatus
sdfm.h	SDFM_clearWaitForSyncFlag
sdfm.h	SDFM_enableWaitForSync
sdfm.h	SDFM_disableWaitForSync
sdfm.h	SDFM_setPWMSyncSource
sdfm.h	SDFM_setFIFOClearOnSyncMode
sdfm.h	SDFM_setWaitForSyncClearMode
SDFLT1CMPL2	
-	
SDCTLPARM2	
-	See SDCTLPARM1
SDDFPARM2	
-	See SDDFPARM1
SDDPARM2	
-	See SDDPARM1
SDFLT2CMPH1	
-	
SDFLT2CMPL1	
-	
SDCPARM2	
-	See SDCPARM1
SDDATA2	
-	See SDDATA1
SDDATFIFO2	
-	
SDCDATA2	
-	
SDFLT2CMPH2	
-	
SDFLT2CMPHZ	
-	

Table 28-97. SDFM Registers to Driverlib Functions (continued)

File	Driverlib Function
SDFIFOCTL2	
-	
SDSYNC2	
-	
SDFLT2CMPL2	
-	
SDCTLPARM3	
-	See SDCTLPARM1
SDDFPARM3	
-	See SDDFPARM1
SDDPARM3	
-	See SDDPARM1
SDFLT3CMPH1	
-	
SDFLT3CMPL1	
-	
SDCPARM3	
-	See SDCPARM1
SDDATA3	
-	See SDDATA1
SDDATFIFO3	
-	
SDCDATA3	
-	
SDFLT3CMPH2	
-	
SDFLT3CMPHZ	
-	
SDFIFOCTL3	
-	
SDSYNC3	
-	
SDFLT3CMPL2	
-	
SDCTLPARM4	
-	See SDCTLPARM1
SDDFPARM4	
-	See SDDFPARM1
SDDPARM4	
-	See SDDPARM1
SDFLT4CMPH1	
-	
SDFLT4CMPL1	
-	
SDCPARM4	

Table 28-97. SDFM Registers to Driverlib Functions (continued)

File	Driverlib Function
-	See SDCPARM1
SDDATA4	
-	See SDDATA1
SDDATFIFO4	
-	
SDCDATA4	
-	
SDFLT4CMPH2	
-	
SDFLT4CMPHZ	
-	
SDFIFOCTL4	
-	
SDSYNC4	
-	
SDFLT4CMPL2	
-	
SDCOMP1CTL	
sdfm.h	SDFM_selectCompEventHighSource
sdfm.h	SDFM_selectCompEventLowSource
SDCOMP1EVT2FLTCTL	
sdfm.c	SDFM_configCompEventLowFilter
sdfm.h	SDFM_initCompEventLowFilter
SDCOMP1EVT2FLTCLKCTL	
sdfm.c	SDFM_configCompEventLowFilter
SDCOMP1EVT1FLTCTL	
sdfm.c	SDFM_configCompEventHighFilter
sdfm.h	SDFM_initCompEventHighFilter
SDCOMP1EVT1FLTCLKCTL	
sdfm.c	SDFM_configCompEventHighFilter
SDCOMP1LOCK	
sdfm.h	SDFM_lockCompEventFilterConfig
SDCOMP2CTL	
-	
SDCOMP2EVT2FLTCTL	
-	
SDCOMP2EVT2FLTCLKCTL	
-	
SDCOMP2EVT1FLTCTL	
-	
SDCOMP2EVT1FLTCLKCTL	
-	
SDCOMP2LOCK	
-	
SDCOMP3CTL	

Table 28-97. SDFM Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
SDCOMP3EVT2FLTCTL	
-	
SDCOMP3EVT2FLTCLKCTL	
-	
SDCOMP3EVT1FLTCTL	
-	
SDCOMP3EVT1FLTCLKCTL	
-	
SDCOMP3LOCK	
-	
SDCOMP4CTL	
-	
SDCOMP4EVT2FLTCTL	
-	
SDCOMP4EVT2FLTCLKCTL	
-	
SDCOMP4EVT1FLTCTL	
-	
SDCOMP4EVT1FLTCLKCTL	
-	
SDCOMP4LOCK	
-	

Chapter 29

► COMMUNICATION PERIPHERALS

The following chapters describe the communication peripherals.

29.1 Technical Reference Manual Overview

The block diagram for the F2838x device is shown in Figure 29-1. This Technical Reference Manual is organized into five major sections:

- C28x SYSTEM RESOURCES

  These chapters describe the C28x CPU subsystem, C28x Boot ROM, device configuration, and other system peripherals.

- ANALOG PERIPHERALS

  These chapters describe the Analog-to-Digital Converter (ADC), Buffered Digital-to-Analog Converter (DAC), Comparator Subsystem (CMPSS), and general analog subsystem configuration.

- CONTROL PERIPHERALS

  These chapters describe the Enhanced Capture (eCAP), High Resolution Capture (HRCAP), Enhanced Pulse Width Modulator (ePWM), Enhanced Quadrature Encoder Pulse (eQEP), and Sigma Delta Filter Module (SDFM) peripherals.

- COMMUNICATION PERIPHERALS

  These chapters describe the communication peripherals available to the C28x subsystem such as the I2C, SCI, FSI, McBSP, PMBUS, and SPI. The CAN, EtherCAT, and USB peripherals are also described in this section and can be assigned to the CM subsystem.

- CONNECTIVITY MANAGER (CM)

  These chapters describe the Connectivity Manager (CM) subsystem as well as the AES, GCRC, CM-I2C, CM-UART, SSI, and Ethernet peripherals.

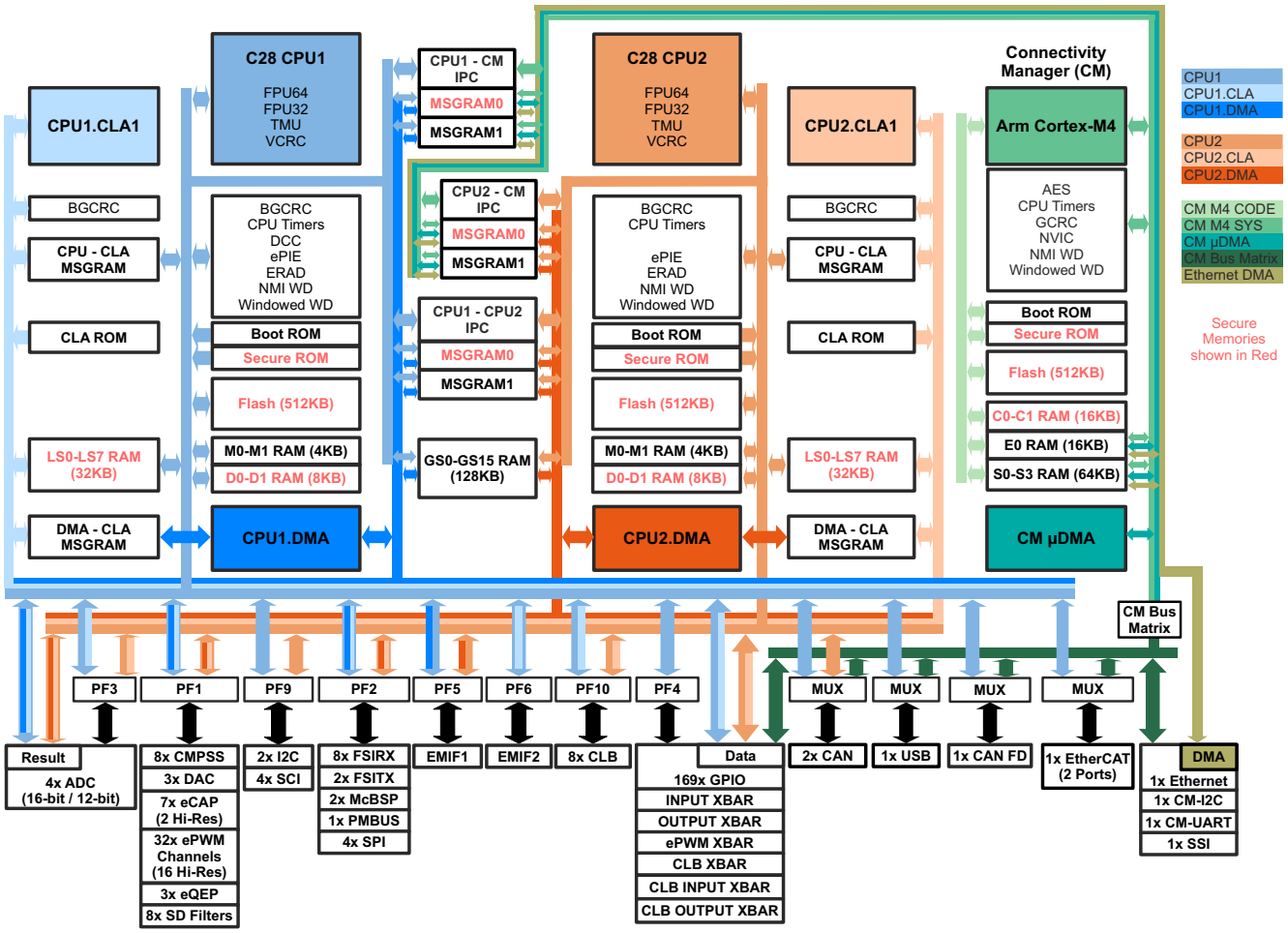


Figure 29-1. F2838x Block Diagram

Chapter 30

Controller Area Network (CAN)

This chapter describes the Controller Area Network (CAN) module. CAN is a serial communications protocol that efficiently supports distributed real-time control with a high level of reliability. The CAN module supports bit rates up to 1 Mbit/s and is compliant with the ISO11898-1 (CAN 2.0B) protocol specification.

Further information can be found in:

- *Calculator for CAN Bit Timing Parameters Application Report*
- *Programming Examples and Debug Strategies for the DCAN Module Application Report*
- *Configurable Error Generator for Controller Area Network Application Report*

30.1 Introduction

This device uses the CAN IP known as DCAN.

30.1.1 DCAN Related Collateral

Foundational Materials

- Automotive CAN Overview and Training (Video)
- C2000 Academy - CAN
  - Refer to the DCAN section
- CAN Physical layer (Video)
- CAN and CAN FD Overview (Video)
- CAN and CAN FD Protocol (Video)

Getting Started Materials

- *Programming Examples and Debug Strategies for the DCAN Module Application Report*

Expert Materials

- *Configurable Error Generator for Controller Area Network Application Report*

30.1.2 Features

The CAN module implements the following features:

- Complies with ISO11898-1 (Bosch® CAN protocol specification 2.0 A and B)
- Bit rates up to 1 Mbps
- Multiple clock sources
- 32 message objects (“message objects” are also referred to as “mailboxes” in this document; the two terms are used interchangeably), each with the following properties:
  - Configurable as receive or transmit
  - Configurable with standard (11-bit) or extended (29-bit) identifier
  - Supports programmable identifier receive mask
  - Supports data and remote frames
  - Holds 0 to 8 bytes of data
  - Parity-checked configuration and data RAM
- Individual identifier mask for each message object
- Programmable FIFO mode for message objects
- Programmable loop-back modes for self-test operation
- Suspend mode for debug support
- Software module reset
- Automatic bus-on, after bus-off state by a programmable 32-bit timer
- Message-RAM parity-check mechanism
- Two interrupt lines
- DMA support

Note

For a CAN bit clock of 200 MHz, the smallest bit rate possible is 7.8125 kbps.

Depending on the timing settings used, the accuracy of the on-chip zero-pin oscillator (specified in the data sheet) may not meet the requirements of the CAN protocol. In this situation, an external clock source must be used.

30.1.3 Block Diagram

Figure 30-1 shows a block diagram of the CAN module. Following is a description of some of the main blocks.

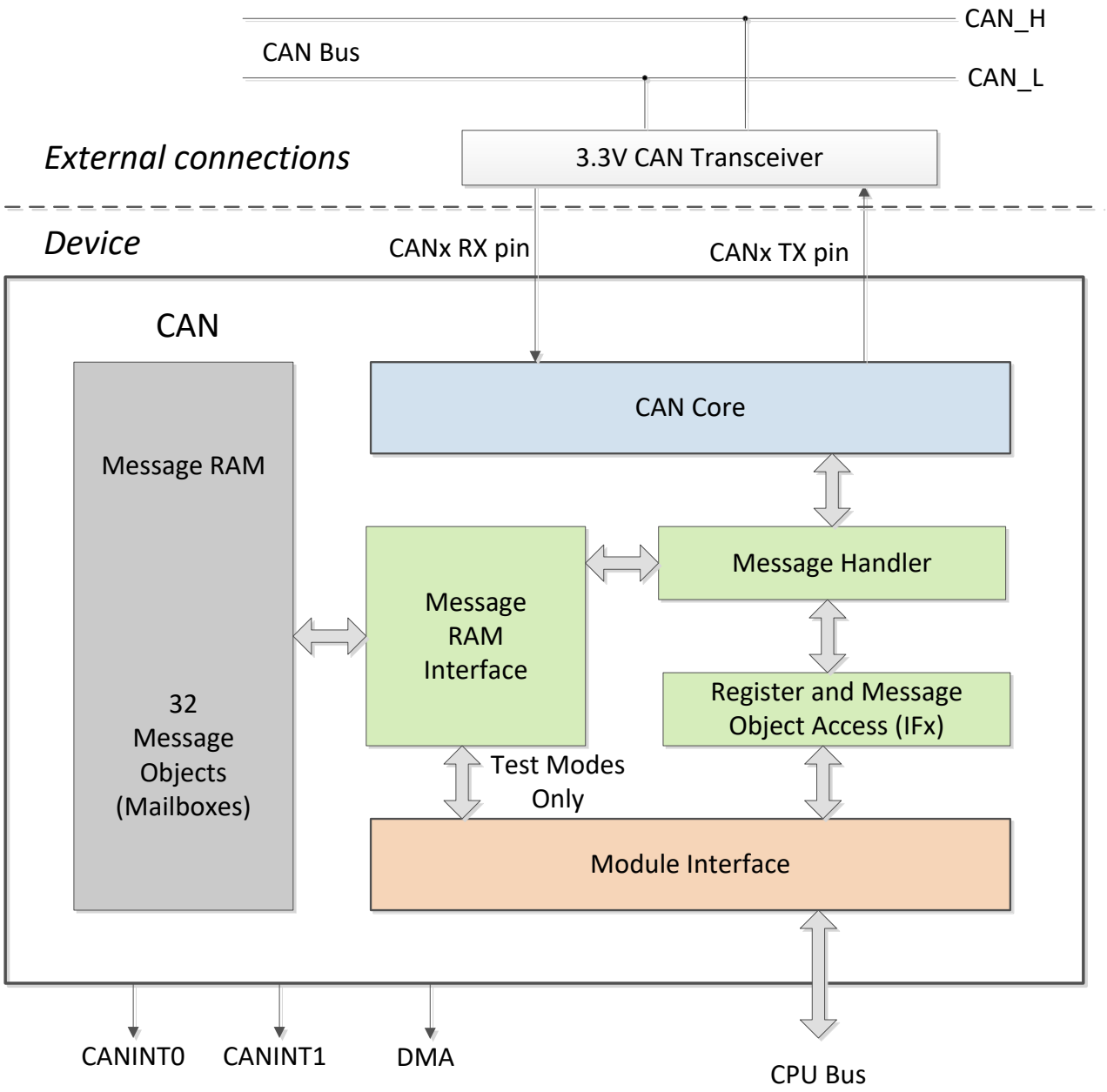


Figure 30-1. CAN Block Diagram

30.1.3.1 CAN Core

The CAN core consists of the CAN Protocol Controller and the Rx/Tx Shift register. It handles all ISO 11898-1 protocol functions.

30.1.3.2 Message Handler

The message handler is a state machine which controls the data transfer between the single-port Message RAM and the CAN Core's Rx/Tx Shift register. It also handles acceptance filtering and the interrupt request generation as programmed in the control registers.

30.1.3.3 Message RAM

The CAN message RAM enables the storage of 32 CAN messages.

30.1.3.4 Registers and Message Object Access (IFx)

Data consistency is provided by indirect accesses to the message objects. During normal operation, all CPU and DMA accesses to the message RAM are done through Interface registers. The IFx registers can be thought of as a "window" through which the message objects (mailboxes) are accessed.

Three Interface register sets control the CPU read and write accesses to the Message RAM, see Figure 30-2. There are two Interface register sets for read/write access (IF1 and IF2) and one Interface register set for read access only (IF3). See also Section 30.13. The Interface registers have the same word length as the message RAM.

In a dedicated test mode, the message RAM is memory-mapped and can be directly accessed.

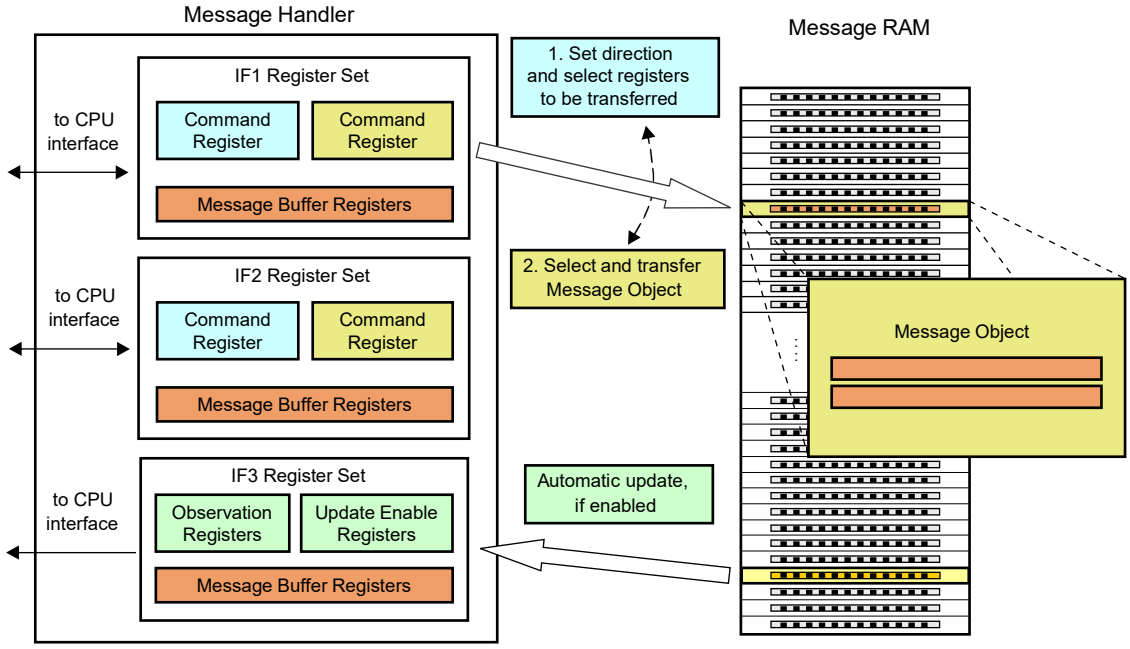


Figure 30-2. Accessing Message Objects Through IFx Registers

30.2 Functional Description

The CAN module performs CAN protocol communication according to ISO 11898-1. The bit rate can be programmed to values up to 1 Mbps. A CAN transceiver chip is required for the connection to the physical layer (CAN bus).

For communication on a CAN network, individual message objects can be configured. The message objects and identifier masks are stored in the Message RAM.

All functions concerning the handling of messages are implemented in the message handler. These functions are: acceptance filtering; the transfer of messages between the CAN Core and the Message RAM; and the handling of transmission requests as well as the generation of interrupts or DMA requests.

The register set of the CAN can be accessed directly by the CPU through the module interface. These registers are used to control and configure the CAN core and the message handler, and to access the message RAM.

30.2.1 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

Some I/O functionality is defined by GPIO register settings independent of this peripheral. For input signals, the GPIO input qualification must be set to asynchronous mode by setting the appropriate GPxQSELn register bits to 11b. The internal pull-ups can be configured in the GPyPUD register.

See the *General-Purpose Input/Output (GPIO)* chapter for more details on GPIO mux and settings.

30.2.2 Address/Data Bus Bridge

The CAN module uses a special addressing scheme to support byte accesses. It is recommended to only use 32-bit accesses to the CAN registers using the *HWREG_BP()* macro that uses the *__byte_peripheral_32()* intrinsic. If 16-bit accesses are to be used, the lower 16-bits must be written to the register's address, and the upper 16-bits must be written to the register's address plus 2.

Because of the bus bridge, the view of the CAN module's register space through the Code Composer Studio™ (CCS) IDE memory window does not always match the actual addressing. When the view mode is 32-bit or 16-bit, even addresses are effectively duplicated; odd addresses can be ignored. When the view mode is 8-bit, even addresses from within the CAN module are duplicated into the odd addresses in the CCS memory view; odd addresses from the module are not displayed.

Table 30-1. CAN Register Access from Software

CAN Register Space			C28x 8-Bit		C28x 16-Bit		C28x 32-Bit	
Address	Name	Data	Access	Data	Access	Data	Address	Data
0x00	CAN_CTL	0x33221100	__byte((int *)0x00,0)	0x0000	(*((short *)(0x00)))	0x1100	(*((long *)(0x00)))	0x33221100
0x04	CAN_ES	0x77665544	__byte((int *)0x01,0)	0x0011	(*((short *)(0x01)))	0x1100	(*((long *)(0x01)))	0x33221100
0x08	CAN_ERRC	0xBBAA9988	__byte((int *)0x02,0)	0x0022	(*((short *)(0x02)))	0x3322	(*((long *)(0x02)))	0x33221100
0x0C	CAN_BTR	0xFFEEDDCC	__byte((int *)0x03,0)	0x0033	(*((short *)(0x03)))	0x3322	(*((long *)(0x03)))	0x33221100
			__byte((int *)0x04,0)	0x0044	(*((short *)(0x04)))	0x5544	(*((long *)(0x04)))	0x77665544
			__byte((int *)0x05,0)	0x0055	(*((short *)(0x05)))	0x5544	(*((long *)(0x05)))	0x77665544
			__byte((int *)0x06,0)	0x0066	(*((short *)(0x06)))	0x7766	(*((long *)(0x06)))	0x77665544
			__byte((int *)0x07,0)	0x0077	(*((short *)(0x07)))	0x7766	(*((long *)(0x07)))	0x77665544
			__byte((int *)0x08,0)	0x0088	(*((short *)(0x08)))	0x9988	(*((long *)(0x08)))	0xBBAA9988
			__byte((int *)0x09,0)	0x0099	(*((short *)(0x09)))	0x9988	(*((long *)(0x09)))	0xBBAA9988
			__byte((int *)0x0A,0)	0x00AA	(*((short *)(0x0A)))	0xBBAA	(*((long *)(0x0A)))	0xBBAA9988
			__byte((int *)0x0B,0)	0x00BB	(*((short *)(0x0B)))	0xBBAA	(*((long *)(0x0B)))	0xBBAA9988
			__byte((int *)0x0C,0)	0x00CC	(*((short *)(0x0C)))	0xDDCC	(*((long *)(0x0C)))	0xFFEEDDCC
			__byte((int *)0x0D,0)	0x00DD	(*((short *)(0x0D)))	0xDDCC	(*((long *)(0x0D)))	0xFFEEDDCC
			__byte((int *)0x0E,0)	0x00EE	(*((short *)(0x0E)))	0xFFEE	(*((long *)(0x0E)))	0xFFEEDDCC
			__byte((int *)0x0F,0)	0x00FF	(*((short *)(0x0F)))	0xFFEE	(*((long *)(0x0F)))	0xFFEEDDCC

Table 30-2. CAN Register Access from Code Composer Studio™ IDE

CCS 8-Bit			CCS 16-Bit			CCS 32-Bit	
Address	Displayed Data		Address	Displayed Data		Address	Displayed Data
0x00	0x00		0x00	0x1100		0x00	0x11001100
0x01	0x00		0x01	0x1100		0x02	0x33223322
0x02	0x22		0x02	0x3322		0x04	0x55445544
0x03	0x22		0x03	0x3322		0x06	0x77667766
0x04	0x44		0x04	0x5544		0x08	0x99889988
0x05	0x44		0x05	0x5544		0x0A	0xBBAABBAA
0x06	0x66		0x06	0x7766		0x0C	0xDDCCDDCC
0x07	0x66		0x07	0x7766		0x0E	0xFFEEFFEE
0x08	0x88		0x08	0x9988			
0x09	0x88		0x09	0x9988			
0x0A	0xAA		0x0A	0xBBAA			
0x0B	0xAA		0x0B	0xBBAA			
0x0C	0xCC		0x0C	0xDDCC			
0x0D	0xCC		0x0D	0xDDCC			
0x0E	0xEE		0x0E	0xFFEE			
0x0F	0xEE		0x0F	0xFFEE			

30.3 Operating Modes

30.3.1 Initialization

The initialization mode is entered either by software (by setting the **Init** bit in the CAN_CTL register), by hardware reset, or by going bus-off. While the Init bit is set, the message transfer from and to the CAN bus is stopped, and the status of the CAN_TX output is recessive (high). The CAN error counters are not updated. Setting the Init bit does not change any other configuration register.

To initialize the CAN Controller, the CPU has to configure the CAN Bit timing and those message objects which have to be used for CAN communication. Message objects which are not needed, can be deactivated with their MsgVal bits cleared.

The access to the Bit Timing Register for the configuration of the bit timing is enabled when both **Init** and CCE bits in the CAN Control register are set.

Clearing the Init bit finishes the software initialization. Afterwards, the bit stream processor (BSP) synchronizes itself to the data transfer on the CAN bus by waiting for the occurrence of a sequence of 11 consecutive recessive bits (= Bus Idle) before the BSP can take part in bus activities and start the message transfer. For more details, see Section 30.12.

The initialization of the message objects is independent of the Init bit; however, all message objects must be configured with particular identifiers or set to "not-valid" before the message transfer is started.

It is possible to change the configuration of message objects during normal operation by the CPU. After setup and subsequent transfer of message object from interface registers to message RAM, the acceptance filtering is applied to it when the modified message object number is same or smaller than the previously found message object. This assures data consistency even when changing message objects; for example, while there is a pending CAN frame reception.

30.3.2 CAN Message Transfer (Normal Operation)

Once the CAN is initialized and the Init bit is reset to zero, the CAN Core synchronizes itself to the CAN bus and is ready for communication.

Received messages are stored into their appropriate message objects, if the messages pass acceptance filtering. The whole message (MSGID, DLC, and up to 8 data bytes) is stored into the message object. As a consequence, for example, if the identifier mask is used, the MSGID bits that are masked to "don't care" can change in the message object when a received message is stored.

The CPU can read or write each message at any time using the interface registers, as the message handler provides data consistency in case of concurrent accesses.

Messages to be transmitted can be updated by the CPU. If a permanent message object (MSGID, control bits set up during configuration, and setup for multiple CAN transfers) exists for the message, it is possible to only update the data bytes. If several transmit messages must be assigned to one message object, the whole message object has to be configured before the transmission of this message is requested.

The transmission of multiple message objects can be requested at the same time. The message objects are subsequently transmitted, according to their internal priority. Messages can be updated or set to 'not valid' at any time, even if a requested transmission is still pending. However, the data bytes are discarded if a message is updated before a pending transmission has started.

Depending on the configuration of the message object, a transmission can be automatically requested by the reception of a remote frame with a matching identifier.

30.3.2.1 Disabled Automatic Retransmission

According to the CAN Specification (see ISO11898, 6.3.3 Recovery Management), the CAN provides a mechanism to automatically retransmit frames that have lost arbitration or have been disturbed by errors during transmission. The frame transmission service is not confirmed to the user before the transmission is successfully completed

By default, this automatic retransmission is enabled and can be disabled by setting the DAR bit in the CAN control register. Further details to this mode are provided in Section 30.11.3.

30.3.2.2 Auto-Bus-On

After the CAN has entered the bus-off state, the CPU can start a bus-off-recovery sequence by resetting the *Init* bit. If this is not done, the module stays in the bus-off state.

The CAN provides an automatic auto-bus-on feature that is enabled by the ABO bit. If set, the CAN automatically starts the bus-off-recovery sequence. The sequence can be delayed by a user-defined number of clock cycles.

Note

If the CAN module goes Bus-Off due to multiple CAN bus errors, the CAN module stops all bus activities and automatically sets the Init bit. Once the Init bit is cleared by the application (or due to the auto-bus-on feature), the device waits for 129 occurrences of Bus Idle (equal to 129 * 11 consecutive recessive bits) before resuming normal operation. The Bus-Off recovery sequence cannot be shortened by setting or resetting the Init bit. At the end of the bus-off recovery sequence, the error counters reset. After the Init bit is reset, each time when a sequence of 11 recessive bits is monitored, a Bit0 Error code is written to the Error and Status Register. This enables the CPU to check whether the CAN bus is stuck at dominant or continuously disturbed, and to monitor the proceeding of the Bus-Off recovery sequence.

30.3.3 Test Modes

The CAN module provides several test modes that are mainly intended for self-test purposes. Figure 30-3 aids in understanding the various test modes. Figure 30-3 must be viewed as representative of the module behavior, and not as a gate-accurate implementation of the module. Figure 30-3 does not include the GPIO muxing or the I/O buffers.

For all test modes, the Test bit in the CAN control register needs to be set to 1 to enable write access to the CAN_TEST register.

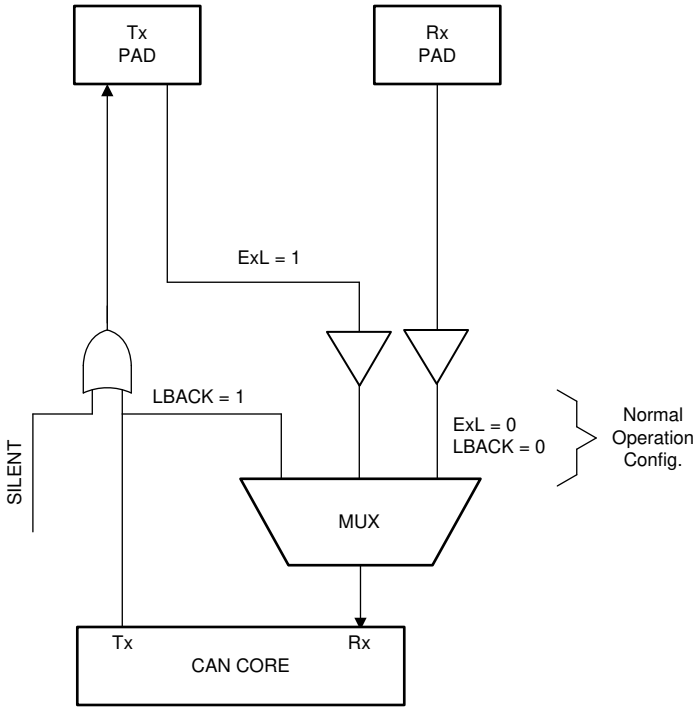


Figure 30-3. CAN_MUX

30.3.3.1 Silent Mode

The silent mode can be used to analyze the traffic on the CAN bus without affecting the CAN by sending dominant bits (for example, acknowledge bit, overload flag, active error flag). The CAN is still able to receive valid data frames and valid remote frames, but the CAN does not send any dominant bits. However, the received frames are internally routed to the CAN Core.

Figure 30-4 shows the connection of signals CAN_TX and CAN_RX to the CAN core in silent mode. Silent mode can be activated by setting the Silent bit in test register (CAN_TEST), to 1. In ISO 11898-1, the silent mode is called the bus monitoring mode.

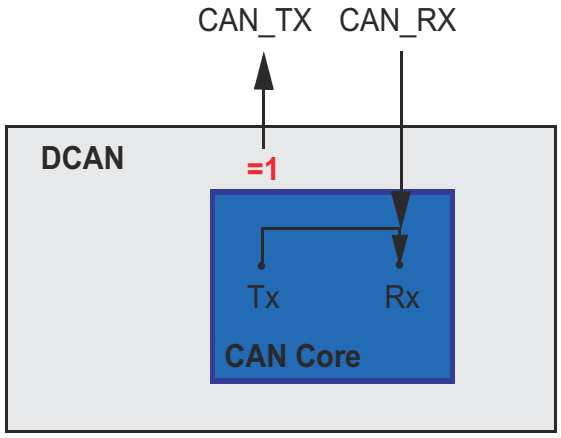


Figure 30-4. CAN Core in Silent Mode

30.3.3.2 Loopback Mode

The loopback mode is mainly intended for hardware self-test functions. In this mode, the CAN core uses internal feedback from Tx output to Rx input. Transmitted messages are treated as received messages, and can be stored into message objects if the messages pass acceptance filtering. The actual value of the CAN_RX input pin is disregarded by the CAN core. Transmitted messages still can be monitored at the CAN_TX pin.

To be independent from external stimulation, the CAN core ignores acknowledge errors (recessive bit sampled in the acknowledge slot of a data/remote frame) in loopback mode.

Figure 30-5 shows the connection of signals CAN_TX and CAN_RX to the CAN core in loopback mode. Loopback mode can be activated by setting the LBack bit in the CAN_TEST register to 1.

Note

In loopback mode, the signal path from the CAN core to the Tx pin, the Tx pin itself, and the signal path from the Tx pin back to the CAN core are disregarded. For including these into the testing, see Section 30.3.3.3.

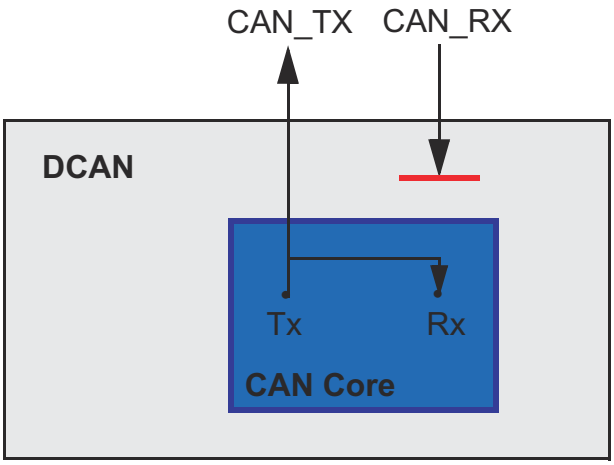


Figure 30-5. CAN Core in Loopback Mode

30.3.3.3 External Loopback Mode

The external loopback mode is similar to the loopback mode; however, the external loopback mode includes the signal path from the CAN core to the Tx pin, the Tx pin itself, and the signal path from the Tx pin back to the CAN core. When the external loopback mode is selected, the CAN core is connected to the input buffer of the Tx pin. With this configuration, the Tx pin IO circuit can be tested. External loopback mode can be activated by setting the ExL bit in Test Register to 1.

Figure 30-6 shows the connection of signals CAN_TX and CAN_RX to the CAN Core in external loopback mode.

Note

When loopback mode is active (LBack bit set), the ExL bit is ignored.

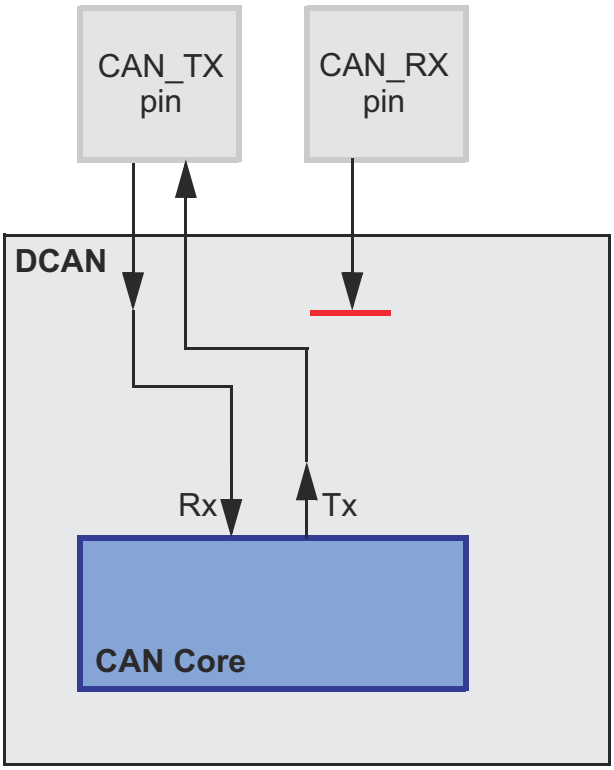


Figure 30-6. CAN Core in External Loopback Mode

30.3.3.4 Loopback Combined with Silent Mode

It is also possible to combine loopback mode and silent mode by setting bits LBack and Silent at the same time. The CAN hardware can be tested without affecting the CAN network. In this mode, the CAN_RX pin is disconnected from the CAN core and no dominant bits are sent on the CAN_TX pin.

Figure 30-7 shows the connection of the signals CAN_TX and CAN_RX to the CAN Core in case of the combination of loopback mode with silent mode.

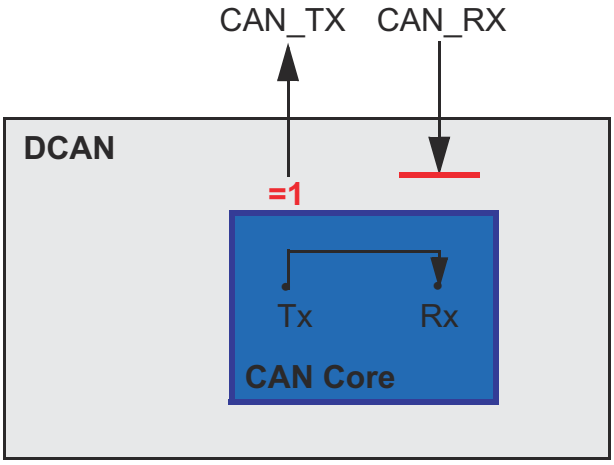


Figure 30-7. CAN Core in Loopback Combined with Silent Mode

30.4 Multiple Clock Source

The CAN bit timing clock is normally derived from the system clock (SYSCLK). If desired, the bit clock can be derived directly from the external clock source (XTAL).

The *System Control and Interrupts* chapter and the device data sheet provide more information on how to configure the relevant clock source registers in the system module.

Note

The CAN core has to be programmed to at least 8 clock cycles per bit time. To achieve a transfer rate of 1Mbps an oscillator frequency of 8 MHz or higher has to be used.

30.5 Interrupt Functionality

Interrupts can be generated on two interrupt lines: CAN0INT and CAN1INT. These lines can be enabled by setting the IE0 and IE1 bits, respectively, in the CAN Control register.

The CAN provides three groups of interrupt sources: message object interrupts, status change interrupts, and error interrupts. The source of an interrupt can be determined by the interrupt identifiers Int0ID and Int1ID in the Interrupt register. When no interrupt is pending, the register holds the value zero. Each interrupt line remains active until the dedicated field in the Interrupt register (Int0ID or Int1ID) again reaches zero (this means the cause of the interrupt is reset), or until IE0 and IE1 are reset. The value 0x8000 in the Int0ID field indicates that an interrupt is pending because the CAN core has updated (not necessarily changed) the Error and Status Register (Error Interrupt or Status Interrupt). This interrupt has the highest priority. The CPU can update (reset) the status bits RxOk, TxOk, and LEC by reading the Error and Status Register, but a write access of the CPU never generates or resets an interrupt.

Values between 1 and the number of the last message object indicates that the source of the interrupt is one of the message objects. INT0ID and INT1ID point to the pending message interrupt with the highest priority. The Message Object 1 has the highest priority, the last message object has the lowest priority.

An interrupt service routine that reads the message from the interrupt source can also read the message and reset the message object's IntPnd at the same time (ClrIntPnd bit in the IF1 or IF2 Command register). When IntPnd is cleared, the Interrupt register points to the next message object with a pending interrupt.

The CAN module features a module-level interrupt enable and acknowledge mechanism. To enable the CAN0 and CAN1 interrupts, you must set the appropriate bits in the CAN_GLB_INT_EN register. When handling an interrupt, the individual message or status change flag must be cleared prior to acknowledging the interrupt using CAN_GLB_INT_CLR and PIEACK.

30.5.1 Message Object Interrupts

Message object interrupts are generated by events from the message objects. They are controlled by the flags IntPND, TxIE and RxIE which are described in Section 30.14.1. Message object interrupts can be routed to the CAN0INT or CAN1INT line, which is controlled by the Interrupt Multiplexer register.

Note that writing to the IntPnd bit in the CAN_IFnMCTL registers can force an interrupt.

30.5.2 Status Change Interrupts

The events RxOk, TxOk, and LEC in the Error and Status register belong to the status change interrupts. The status change interrupt group can be enabled by the SIE bit in the CAN Control Register. If SIE is set, a status change interrupt is generated at each CAN frame, independent of bus errors or valid CAN communication, and also independent of the Message RAM configuration. Status Change interrupts can only be routed to interrupt line CAN0INT which has to be enabled by setting IE0 in the CAN_CTL Register.

30.5.3 Error Interrupts

The events PER, BOff and EWarn, belong to the error interrupts. The error interrupt group can be enabled by setting bit EIE. Also, error interrupts can only be routed to interrupt line CAN0INT, which has to be enabled by setting IE0 in the CAN_CTL register.

30.5.4 Peripheral Interrupt Expansion (PIE) Module Nomenclature for DCAN Interrupts

Table 30-3 shows the Peripheral Interrupt Expansion (PIE) module nomenclature for the interrupts.

Table 30-3. PIE Module Nomenclature for Interrupts

Interrupt	CANA	CANB
CANINT0	CANA_0	CANB_0
CANINT1	CANA_1	CANB_1

30.5.5 Interrupt Topologies

Interrupt topologies for CAN are illustrated in Figure 30-8 and Figure 30-9. Mailbox interrupts for transmit and receive operations can be routed to both CANINT0 and CANINT1. However, error and status interrupts can only be routed to CANINT0.

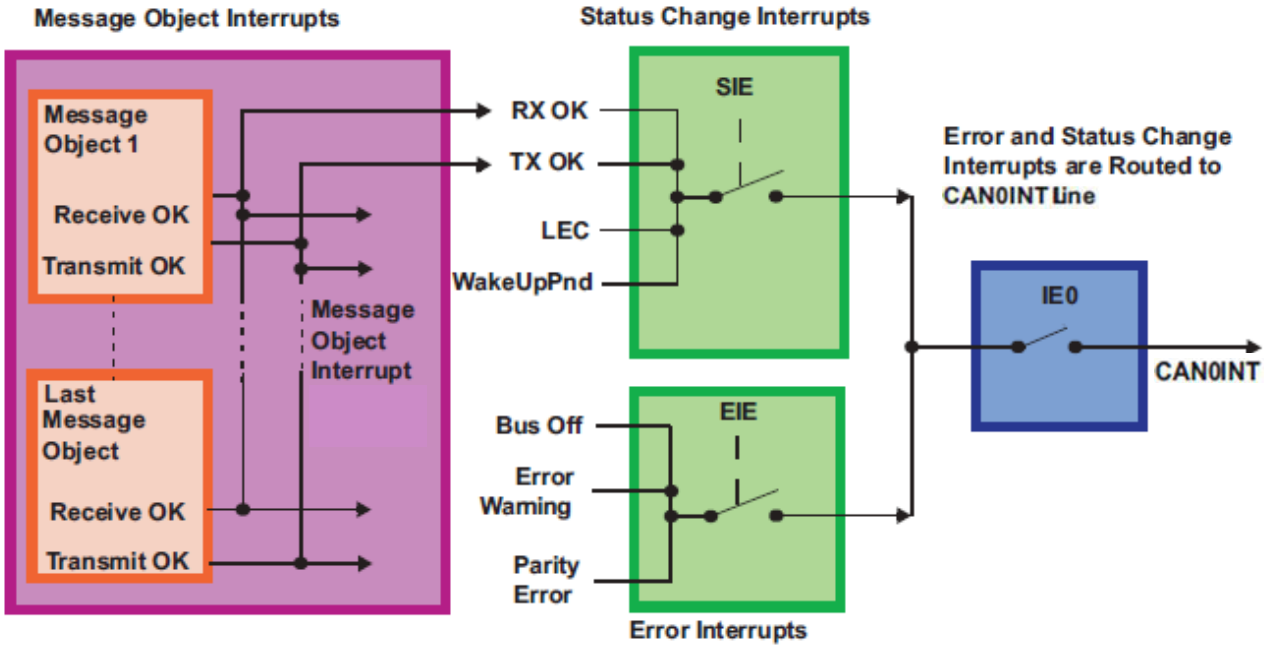


Figure 30-8. CAN Interrupt Topology 1

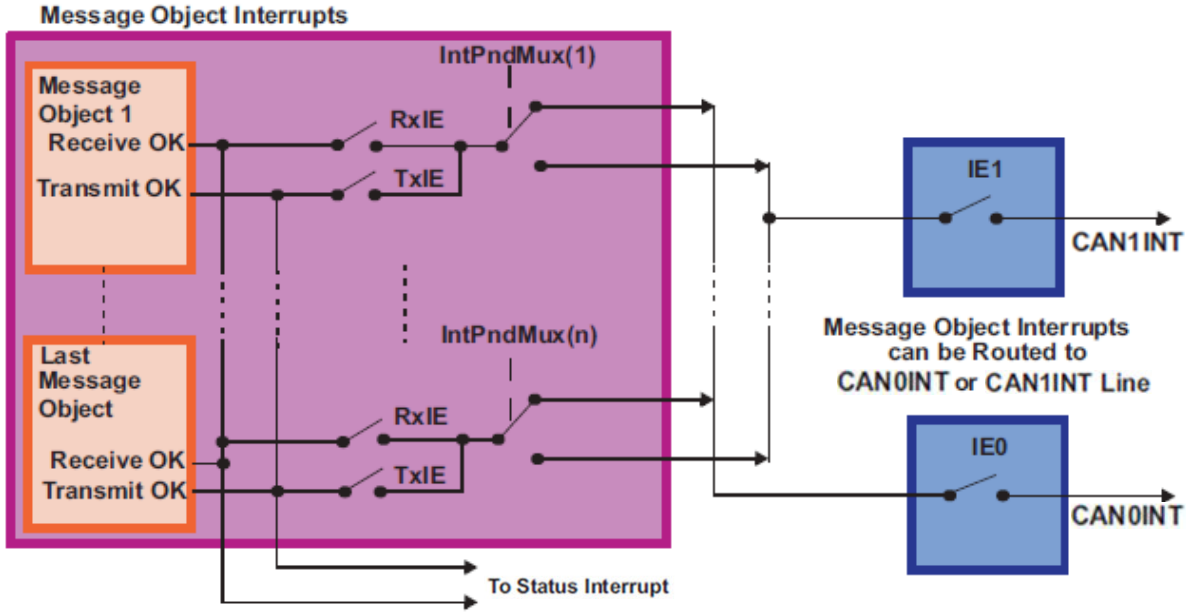


Figure 30-9. CAN Interrupt Topology 2

30.6 DMA Functionality

The CAN module provides three DMA trigger outputs, one for each of the three Interface Registers IF1, IF2 and IF3. These can be enabled using the DE1, DE2, and DE3 bits in the CAN_CTL register.

The Update of IF1 and IF2 registers is initiated by a write access to the IF1 and IF2 Command registers, respectively. Once enabled, setting the DMAactive bit in the IF1CMD or IF2CMD registers cause a DMA request the next time the corresponding interface becomes available.

The IF3 registers content can be automatically updated on reception of CAN messages in message objects which are programmed for automatic IF3 update. That is, when IF3 DMA requests are enabled, all IF3 updates trigger a DMA request.

When a DCAN internal IFx update is complete, a DMA request is activated and stays active until the first access to one of the relevant IFx registers; that is, DMA requests are cleared after the first read or write access to an IF register set.

30.7 Parity Check Mechanism

The CAN provides a parity check mechanism to make sure data integrity of the message RAM data. For each word (32 bits) in Message RAM, one parity bit is calculated.

Parity information is stored in the Message RAM on write accesses and is checked against the stored parity bit from Message RAM on read accesses.

The parity check functionality can be enabled or disabled by the PMD bit field in the CAN control register. In case of a disabled parity check, the parity bits in message RAM are left unchanged on write access to the data area and no check is done on read access.

If parity checking is enabled, parity bits are automatically generated and checked by the CAN. A parity bit is set if the modulo-2-sum of the data bits is 1. This means that if the parity bit is set, then there are an odd number of 1 bits in the data.

30.7.1 Behavior on Parity Error

On any read access to Message RAM, for example, during the start of a CAN frame transmission, the parity of the message object is checked. If a parity error is detected, the PER bit in the Error and Status register is set. If error interrupts are enabled, an interrupt can also be generated. To avoid the transmission of invalid data over the CAN bus, the MsgVal bit of the message object is reset.

The message object data can be read by the CPU, independently of parity errors. Thus, the application has to make sure that the read data is valid, for example, by immediately checking the Parity Error Code register on parity error interrupt.

30.8 Debug Mode

The module supports the usage of an external debug unit by providing functions like pausing CAN activities and making message RAM content accessible from the debugger. Debug mode is entered automatically when an external debugger is connected and the core is halted.

Before entering Debug mode, the circuit waits until a transmission is started, a reception is finished, or the Bus idle state is recognized. If the IDS bit is set, the debugger immediately interrupts the current transmission or reception. Afterwards, the CAN enters Debug mode, indicated by the InitDbg flag, in the CAN Control register. During debug mode, all CAN registers can be accessed. Reading reserved bits returns a 0; writing to reserved bits has no effect. Also, the message RAM is memory-mapped, so this allows the external debug unit to read the message RAM. For the memory organization (see Section 30.14.3).

Note

During debug mode, the Message RAM cannot be accessed using the IFx register sets.

Writing to control registers in Debug mode can influence the CAN state machine and further message handling.

For debug support, the auto clear functionality of the following CAN registers is disabled:

- Error and Status register (clear of status flags by read)
- IF1/IF2 Command registers (clear of DMA Active flag by read and write)

30.9 Module Initialization

After hardware reset, the Init bit in the CAN Control register is set and all CAN protocol functions are disabled. The configuration of the bit timing and of the message objects must be completed before the CAN protocol functions are enabled.

For the configuration of the message objects, see Section 30.10.

For the configuration of the Bit Timing, see Section 30.12.2.

The bits MsgVal, NewDat, IntPnd, and TxRqst of the message objects are reset to 0 by a hardware reset. The configuration of a message object is done by programming Mask, Arbitration, Control and Data bits of one of the IF1/IF2 Interface register sets to the desired values. By writing the message object number to bits [7:0] of the corresponding IF1/IF2 Command register, the IF1/IF2 Interface Register content is loaded into the addressed message object in the Message RAM.

The configuration of the bit timing requires that the CCE bit in the CAN Control register is set additionally to Init. This is not required for the configuration of the message objects.

When the Init bit in the CAN Control register is cleared, the CAN Protocol Controller state machine of the CAN Core and the message handler State Machine start to control the CAN's internal data flow. Received messages which pass the acceptance filtering are stored into the Message RAM; messages with pending transmission request are loaded into the CAN Core's Shift register and are transmitted using the CAN bus.

The CPU can enable the interrupt lines (setting IE0 and IE1 to 1) at the same time when the CPU clears Init and CCE. The status interrupts EIE and SIE can be enabled simultaneously.

The CAN communication can be controlled interrupt-driven or in polling mode. The Interrupt Register points to those message objects with IntPnd = 1. The register is updated even if the interrupt lines to the CPU are disabled (IE0 and IE1 are 0).

The CPU can poll all MessageObject's NewDat and TxRqst bits in parallel from the NewData registers and the Transmission Request registers. Polling can be made easier if all Transmit Objects are grouped at the low numbers; all Receive Objects are grouped at the high numbers.

30.10 Configuration of Message Objects

The entire Message RAM must be configured before the end of the initialization; however, it is also possible to change the configuration of message objects during CAN communication.

30.10.1 Configuration of a Transmit Object for Data Frames

Figure 30-10 shows how a transmit object can be initialized.

Figure 30-10. Initialization of a Transmit Object

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	1	0	0	0	appl.	0	appl.	0

- The arbitration bits (ID[28:0] and Xtd bit) are given by the application. The arbitration bits define the identifier and type of the outgoing message. If an 11-bit Identifier (standard frame) is used (Xtd = 0), the Identifier is programmed to ID[28:18]. In this case, ID[17:0] can be ignored.
- The data registers (DLC[3:0] and Data0-7) are given by the application. TxRqst and RmtEn must not be set before the data is valid.
- If the TxIE bit is set, the IntPnd bit is set after a successful transmission of the message object.
- If the RmtEn bit is set, a matching received remote frame causes the TxRqst bit to be set; the remote frame is autonomously answered by a data frame.
- The Mask bits (Msk[28:0], UMask, MXtd, and MDir bits) can be used (UMask = 1) to allow groups of remote frames with similar identifiers to set the TxRqst bit. The Dir bit must not be masked. For details, see Section 30.11.8. Identifier masking must be disabled (UMask = 0) if no remote frames are allowed to set the TxRqst bit (RmtEn = 0).

30.10.2 Configuration of a Transmit Object for Remote Frames

It is not necessary to configure transmit objects for the transmission of remote frames. Setting TxRqst for a receive object causes the transmission of a remote frame with the same identifier as the data frame for which this receive object is configured.

30.10.3 Configuration of a Single Receive Object for Data Frames

Figure 30-11 shows how a receive object for data frames can be initialized.

Figure 30-11. Initialization of a Single Receive Object for Data Frames

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	0	0	0	appl.	0	0	0	0

- The arbitration bits (ID[28:0] and Xtd bit) are given by the application. The arbitration bits define the identifier and type of accepted received messages. If an 11-bit Identifier (Standard Frame) is used (Xtd = 0), the Identifier is programmed to ID[28:18]. In this case, ID[17:0] can be ignored. When a data frame with an 11-bit Identifier is received, ID[17:0] is set to 0.
- When the message handler stores a data frame in the message object, the message handler stores the received data length code and the corresponding number of data bytes. If the data length code is less than 8, the remaining bytes of the message object can be overwritten by non-specified values.
- The mask bits (Msk[28:0], UMask, MXtd, and MDir bits) can be used (UMask = 1) to allow groups of data frames with similar identifiers to be accepted. The Dir bit must not be masked in typical applications. If some bits of the Mask bits are set to "don't care", the corresponding bits of the Arbitration Register are overwritten by the bits of the stored data frame.
- If the RxIE bit is set, the IntPnd bit is set when a received data frame is accepted and stored in the message object.
- If the TxRqst bit is set, the transmission of a remote frame with the same identifier as stored in the Arbitration bits is triggered. The content of the Arbitration bits can change if the Mask bits are used (UMask = 1) for acceptance filtering.

30.10.4 Configuration of a Single Receive Object for Remote Frames

Figure 30-12 shows how a receive object for remote frames can be initialized.

Figure 30-12. Initialization of a Single Receive Object for Remote Frames

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	1	0	0	appl.	0	0	0	0

- Receive objects for remote frames can be used to monitor remote frames on the CAN bus. The remote frame stored in the receive object does not trigger the transmission of a data frame. Receive objects for remote frames can be expanded to a FIFO buffer, see Section 30.10.5.
- UMask must be set to 1. The Mask bits (Msk[28:0], UMask, MXtd, and MDir bits) can be set to "must-match" or to "don't care", to allow groups of remote frames with similar identifiers to be accepted. The Dir bit must not be masked in typical applications. For details, see Section 30.11.8.
- The arbitration bits (ID[28:0] and Xtd bit) can be given by the application. The arbitration bits define the identifier and type of accepted received remote frames. If some bits of the Mask bits are set to "don't care", the corresponding bits of the arbitration bits are overwritten by the bits of the stored remote frame. If an 11-bit Identifier (standard frame) is used (Xtd = 0), the Identifier is programmed to ID[28:18]. In this case, ID[17:0] can be ignored. When a remote frame with an 11-bit Identifier is received, ID[17:0] is set to 0.
- The data length code (DLC[3:0]) can be given by the application. When the message handler stores a remote frame in the message object, the message handler stores the received data length code. The data bytes of the message object remain unchanged.
- If the RxIE bit is set, the IntPnd bit is set when a received remote frame is accepted and stored in the message object.

30.10.5 Configuration of a FIFO Buffer

With the exception of the EoB bit, the configuration of receive objects belonging to a FIFO buffer is the same as the configuration of a single receive object.

To concatenate multiple message objects to form a FIFO , the identifiers and masks (if used) of these message objects have to be programmed to matching values. Due to the implicit priority of the message objects, the message object with the lowest number is the first message object of the FIFO buffer. The EoB bit of all message objects of a FIFO buffer except the last one must be programmed to 0. The EoB bit of the last message object of a FIFO buffer is set to 1, configuring it as the end of the block.

30.11 Message Handling

When initialization is finished, the CAN module synchronizes itself to the traffic on the CAN bus. The CAN module does acceptance filtering on received messages and stores those frames that are accepted into the designated message objects. The application must update the data of the messages to be transmitted and to enable and request their transmission. The transmission is requested automatically when a matching remote frame is received.

The application can read messages that are received and accepted. Messages that are not read before the next messages are accepted for the same message object are overwritten. Messages can be read interrupt-driven or after polling of NewDat.

30.11.1 Message Handler Overview

The message handler state machine controls the data transfer between the Rx/Tx Shift Register of the CAN Core and the Message RAM. The message handler state machine performs the following tasks:

- Data transfer from Message RAM to CAN Core (messages to be transmitted).
- Data transfer from CAN Core to the Message RAM (received messages).
- Data transfer from CAN Core to the Acceptance Filtering unit.
- Scanning of Message RAM for a matching message object (acceptance filtering).
- Scanning the same message object after being changed by IF1/IF2 registers when priority is same or higher as message the object found by last scanning.
- Handling of TxRqst flags.
- Handling of interrupt flags.

The message handler registers contains status flags of all message objects grouped into the following topics:

- Transmission request flags
- New data flags
- Interrupt pending flags
- Message valid registers

Instead of collecting above listed status information of each message object using IFx registers separately, these message handler registers provide a fast and easy way to get an overview, for example, about all pending transmission requests.

All message handler registers are read-only.

30.11.2 Receive/Transmit Priority

The receive/transmit priority for the message objects is attached to the message number, not to the CAN identifier. Message object 1 has the highest priority, while message object 32 has the lowest priority. If more than one transmission request is pending, the requests are serviced due to the priority of the corresponding message object, so for example, messages with the highest priority can be placed in the message objects with the lowest numbers.

The acceptance filtering for received data frames or remote frames is also done in ascending order of message objects, so a frame that has been accepted by a message object cannot be accepted by another message object with a higher message number. The last message object can be configured to accept any data frame or remote frame that was not accepted by any other message object, for nodes that need to log the complete message traffic on the CAN bus.

30.11.3 Transmission of Messages in Event Driven CAN Communication

If the shift register of the CAN Core is ready for loading and if there is no data transfer between the IFx registers and Message RAM, the MsgVal bits in the Message Valid register and the TxRqst bits in the transmission request register are evaluated. The valid message object with the highest priority pending transmission request is loaded into the shift register by the message handler and the transmission is started. The message object's NewDat bit is reset.

After a successful transmission and if no new data was written to the message object (NewDat = 0) since the start of the transmission, the TxRqst bit is reset. If TxIE is set, IntPnd is set after a successful transmission. If the CAN has lost the arbitration or if an error occurred during the transmission, the message is retransmitted as soon as the CAN bus is free again. If meanwhile the transmission of a message with higher priority has been requested, the messages are transmitted in the order of their priority.

If automatic retransmission mode is disabled by setting the DAR bit in the CAN control register, the behavior of bits TxRqst and NewDat in the Message Control register of the Interface register set is as follows:

- When a transmission starts, the TxRqst bit of the respective Interface register set is reset, while bit NewDat remains set.
- When the transmission has been successfully completed, the NewDat bit is reset.

When a transmission failed (lost arbitration or error) bit NewDat remains set. To restart the transmission, the application has to set TxRqst again.

Received remote frames do not require a receive object and automatically trigger the transmission of a data frame, if in the matching Transmit Object the RmtEn bit is set.

30.11.4 Updating a Transmit Object

The CPU can update the data bytes of a transmit object any time using the IF1 and IF2 interface registers; neither MsgVal nor TxRqst have to be reset before the update.

Even if only a part of the data bytes are to be updated, all four bytes in the corresponding IF1/IF2 Data A register or IF1/IF2 Data B register have to be valid before the content of that register is transferred to the message object. Either the CPU has to write all four bytes into the IF1/IF2 Data register or the message object is transferred to the IF1/IF2 Data Register before the CPU writes the new data bytes.

When only the data bytes are updated, first 0x87 can be written to bits [23:16] of the Command register and then the number of the message object is written to bits [7:0] of the Command register, concurrently updating the data bytes and setting TxRqst with NewDat.

To prevent the reset of TxRqst at the end of a transmission that can already be in progress while the data is updated, NewDat has to be set together with TxRqst in event driven CAN communication. For details see Section 30.11.3.

When NewDat is set together with TxRqst, NewDat is reset as soon as the new transmission has started.

30.11.5 Changing a Transmit Object

If the number of implemented message objects is not sufficient to be used as permanent message objects only, the transmit objects can be managed dynamically. The CPU can write the whole message (arbitration, control, and data) into the Interface register. The bits [23:16] of the Command register can be set to 0xB7 for the transfer of the whole message object content into the message object. Neither MsgVal nor TxRqst have to be reset before this operation.

If a previously requested transmission of this message object is not completed but already in progress, the transmission is continued; however, the transmission is not repeated if the transmission is disturbed.

To update only the data bytes of a message being transmitted, set bits [23:16] of the Command register to 0x87.

Note

After the update of the transmit object, the interface register set contains a copy of the actual contents of the object, including the part that had not been updated.

30.11.6 Acceptance Filtering of Received Messages

When the arbitration and control bits (Identifier + IDE + RTR + DLC) of an incoming message is completely shifted into the shift register of the CAN Core, the message handler starts to scan the message RAM for a matching valid message object:

- The acceptance filtering unit is loaded with the arbitration bits from the CAN Core shift register.
- Then the arbitration and mask bits (including MsgVal, UMask, NewDat, and EoB) of Message Object 1 are loaded into the Acceptance Filtering unit and are compared with the arbitration bits from the shift register. This is repeated for all following message objects until a matching message object is found, or until the end of the Message RAM is reached.
- If a match occurs, the scanning is stopped and the message handler proceeds depending on the type of the frame (data frame or remote frame) received.

30.11.7 Reception of Data Frames

The message handler stores the message from the CAN Core shift register into the respective message object in the Message RAM. Not only the data bytes, but all arbitration bits and the data length code are stored into the corresponding message object. This makes sure that the data bytes stay associated to the identifier even if arbitration mask registers are used.

The NewDat bit is set to indicate that new data (not yet seen by the CPU) has been received. The CPU must reset the NewDat bit when the CPU reads the message object. If at the time of the reception the NewDat bit was already set, MsgLst is set to indicate that the previous data (not seen by the CPU) is lost. If the RxIE bit is set, the IntPnd bit is set, causing the Interrupt Register to point to this message object.

The TxRqst bit of this message object is reset to prevent the transmission of a remote frame, while the requested data frame has just been received.

30.11.8 Reception of Remote Frames

When a remote frame is received, three different configurations of the matching message object have to be considered:

1. Dir = '1' (direction = transmit), RmtEn = '1', UMask = '1' or '0'
   The TxRqst bit of this message object is set at the reception of a matching remote frame. The rest of the message object remains unchanged.
2. Dir = '1' (direction = transmit), RmtEn = '0', UMask = '0'
   The remote frame is ignored, this message object remains unchanged.
3. Dir = '1' (direction = transmit), RmtEn = '0', UMask = '1'
   The remote frame is treated similar to a received data frame. At the reception of a matching remote frame, the TxRqst bit of this message object is reset. The arbitration and control bits (Identifier + IDE + RTR + DLC) from the shift register are stored in the message object in the Message RAM and the NewDat bit of this message object is set. The data bytes of the message object remain unchanged

30.11.9 Reading Received Messages

The CPU can read a received message any time using the IFx interface registers, the data consistency is provided by the message handler state machine.

Typically the CPU writes 0x7F to bits [23:16] and then the number of the message object to bits [7:0] of the Command Register. That combination transfers the whole received message from the Message RAM into the Interface Register set. Additionally, the bits NewDat and IntPnd are cleared in the Message RAM (not in the Interface Register set). The values of these bits in the Message Control Register always reflect the status before resetting the bits.

If the message object uses masks for acceptance filtering, the arbitration bits show which of the different matching messages has been received.

The actual value of NewDat shows whether a new message has been received since the last time when this message object was read. The actual value of MsgLst shows whether more than one message have been received since the last time when this message object was read. MsgLst is not automatically reset.

30.11.10 Requesting New Data for a Receive Object

By means of a remote frame, the CPU can request another CAN node to provide new data for a receive object. Setting the TxRqst bit of a receive object causes the transmission of a remote frame with the receive object's identifier. This remote frame triggers the other CAN node to start the transmission of the matching data frame. If the matching data frame is received before the remote frame can be transmitted, the TxRqst bit is automatically reset.

Setting the TxRqst bit without changing the contents of a message object requires the value 0x84 in bits [23:16] of the Command Register.

30.11.11 Storing Received Messages in FIFO Buffers

Several message objects can be grouped to form one or more FIFO Buffers. Each FIFO Buffer configured to store received messages with a particular (group of) Identifiers. Arbitration and Mask registers of the FIFO Buffer's message objects are identical. The EoB (End of Buffer) bits of all but the last of the FIFO Buffer's message objects are 0, in the last bit the EoB bit is 1.

Received messages with identifiers matching to a FIFO Buffer are stored into a message object of this FIFO Buffer, starting with the message object with the lowest message number.

When a message is stored into a message object of a FIFO Buffer the NewDat bit of this message object is set. By setting NewDat while EoB is 0, the message object is locked for further write accesses by the message handler until the CPU has cleared the NewDat bit.

Messages are stored into a FIFO Buffer until the last message object of this FIFO Buffer is reached. If none of the preceding message objects is released by writing NewDat to 0, all further messages for this FIFO Buffer are written into the last message object of the FIFO Buffer (EoB = 1) and therefore overwrite previous messages in this message object.

30.11.12 Reading from a FIFO Buffer

Several messages can be accumulated in a set of message objects that are concatenated to form a FIFO buffer before the application program is required (to avoid the loss of data) to empty the buffer. A FIFO buffer of length N stores N-1 plus the last received message since the last time the FIFO buffer was cleared. A FIFO buffer is cleared by reading and resetting the NewDat bits of all the message objects, starting at the FIFO object with the lowest message number. This can be done in a subroutine following the example shown in Figure 30-13.

Note

All message objects of a FIFO buffer needs to be read and cleared before the next batch of messages can be stored. Otherwise, true FIFO functionality cannot be maintained, since the message objects of a partly read buffer are refilled according to the normal (descending) priority.

Reading from a FIFO Buffer message object and resetting the NewDat bit is handled the same way as reading from a single message object.

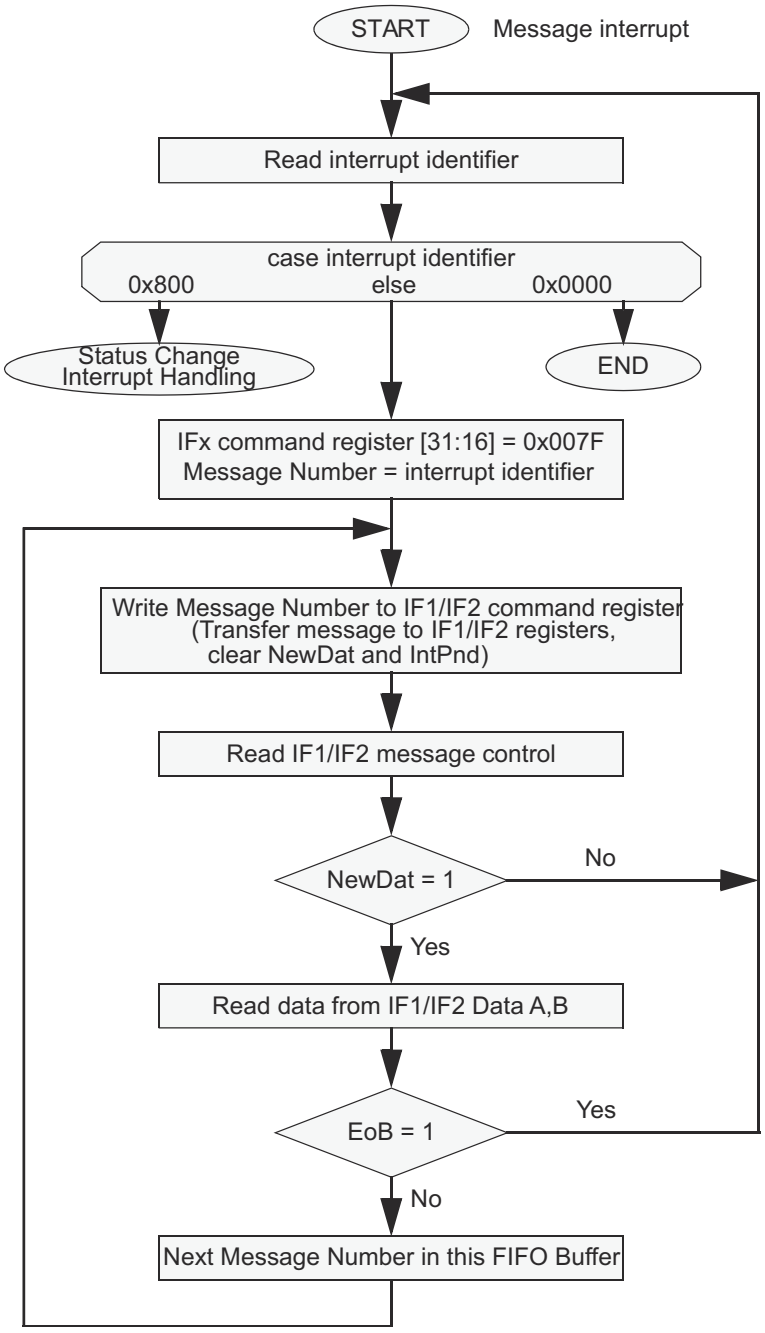


Figure 30-13. CPU Handling of a FIFO Buffer (Interrupt Driven)

30.12 CAN Bit Timing

The CAN supports bit rates up to 1000 kBit/s.

Each CAN node has its own clock generator, typically derived from a crystal oscillator. The bit timing parameters can be configured individually for each CAN node, creating a common Bit rate even though the CAN nodes' oscillator periods (F_{osc}) can be different.

The frequencies of these oscillators are not absolutely stable. Small variations are caused by changes in temperature or voltage and by deteriorating components. As long as the variations remain inside a specific oscillator tolerance range (df), the CAN nodes are able to compensate for the different bit rates by resynchronizing to the bit stream.

In many cases, the CAN bit synchronization amends a faulty configuration of the CAN bit timing to such a degree that only occasionally an error frame is generated. In the case of arbitration, when two or more CAN nodes simultaneously try to transmit a frame, a misplaced sample point can cause one of the transmitters to become error passive.

The analysis of such sporadic errors requires a detailed knowledge of the CAN bit synchronization inside a CAN node and of the CAN nodes' interaction on the CAN bus.

Even if minor errors in the configuration of the CAN bit timing do not result in immediate failure, the performance of a CAN network can be reduced significantly.

30.12.1 Bit Time and Bit Rate

According to the CAN specification, the Bit time is divided into four segments (see Figure 30-14):

- Synchronization Segment (Sync_Seg)
- Propagation Time Segment (Prop_Seg)
- Phase Buffer Segment 1 (Phase_Seg1)
- Phase Buffer Segment 2 (Phase_Seg2)

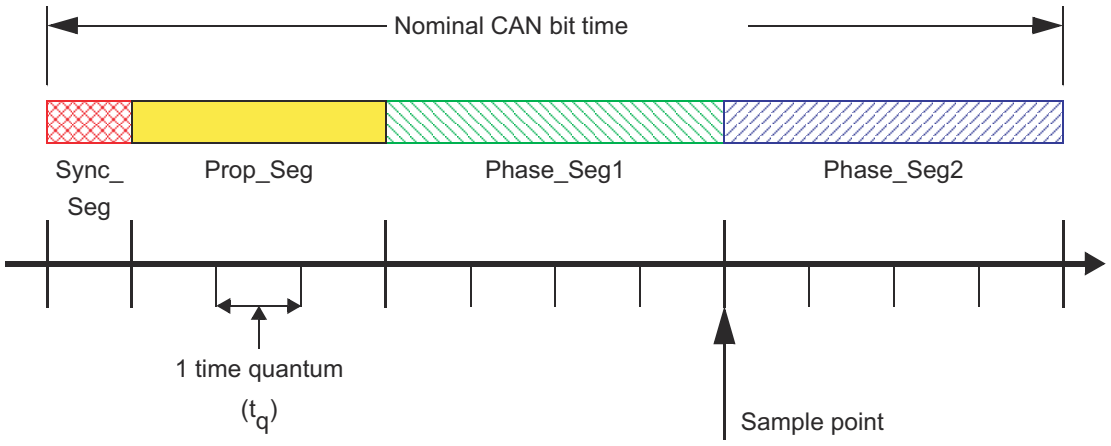


Figure 30-14. Bit Timing

Each segment consists of a specific number of time quanta. The length of one time quantum (t_q), which is the basic time unit of the bit time, is given by the CAN_CLK and the Baud Rate Prescalers (BRPE and BRP). With these two Baud Rate Prescalers combined, divider values from 1 to 1024 can be programmed:

t_q = Baud Rate Prescaler/CAN_CLK

Apart from the fixed length of the synchronization segment, these numbers are programmable. Table 30-4 describes the minimum programmable ranges required by the CAN protocol.

A given bit rate can be met by different bit time configurations.

Table 30-4. Programmable Ranges Required by CAN Protocol

Parameter	Range	Remark
Sync_Seg	1 t_q (fixed)	Synchronization of bus input to CAN_CLK
Prop_Seg	[1 … 8] t_q	Compensates for the physical delay times
Phase_Seg1	[1 … 8] t_q	Can be lengthened temporarily by synchronization
Phase_Seg2	[1 … 8] t_q	Can be shortened temporarily by synchronization
Synchronization Jump Width (SJW)	[1 … 4] t_q	Cannot be longer than either phase buffer segment

Note

For proper functionality of the CAN network, the physical delay times and the oscillator's tolerance range have to be considered.

30.12.1.1 Synchronization Segment

The Synchronization Segment (Sync_Seg) is the part of the bit time where edges of the CAN bus level are expected to occur. If an edge occurs outside of Sync_Seg, its distance to the Sync_Seg is called the phase error of this edge.

30.12.1.2 Propagation Time Segment

This part of the bit time is used to compensate physical delay times within the CAN network. These delay times consist of the signal propagation time on the bus and the internal delay time of the CAN nodes.

Any CAN node synchronized to the bit stream on the CAN bus can be out of phase with the transmitter of the bit stream, caused by the signal propagation time between the two nodes. The CAN protocol's nondestructive bitwise arbitration and the dominant acknowledge bit provided by receivers of CAN messages require that a CAN node transmitting a bit stream must also be able to receive dominant bits transmitted by other CAN nodes that are synchronized to that bit stream. The example in Figure 30-15 shows the phase shift and propagation times between two CAN nodes.

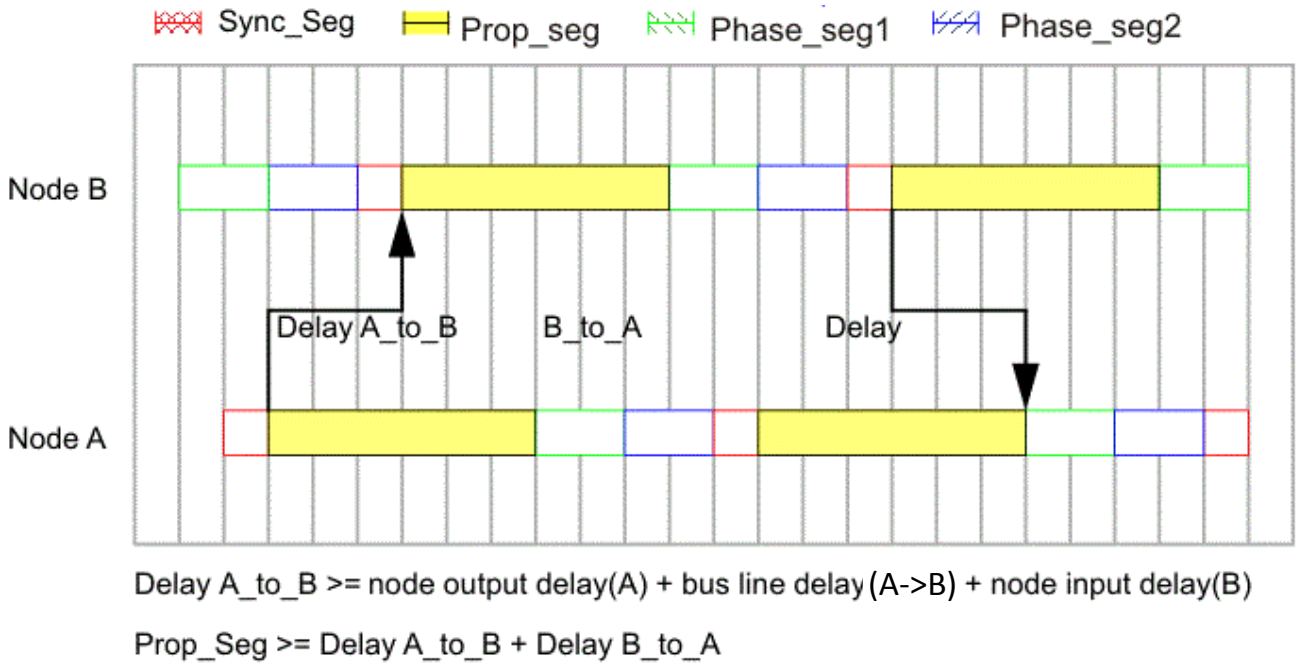


Figure 30-15. Propagation Time Segment

In this example, both nodes A and B are transmitters performing an arbitration for the CAN bus. Node A has sent a Start of Frame bit less than one bit time earlier than node B, therefore node B has synchronized itself to the received edge from recessive to dominant. Since node B has received this edge delay (A_to_B) after the bit has been transmitted, node B's bit timing segments are shifted with regard to node A. Node B sends an identifier with higher priority, so node B wins the arbitration at a specific identifier bit when node B transmits a dominant bit while node A transmits a recessive bit. The dominant bit transmitted by node B arrives at node A after the delay (B_to_A).

Due to oscillator tolerances, the actual position of node A's Sample Point can be anywhere inside the nominal range of node A's Phase Buffer Segments, so the bit transmitted by node B must arrive at node A before the start of Phase_Seg1. This condition defines the length of Prop_Seg.

If the edge from recessive to dominant transmitted by node B arrives at node A after the start of Phase_Seg1, node A could potentially sample a recessive bit instead of a dominant bit, resulting in a bit error and the destruction of the current frame by an error flag.

This error only occurs when two nodes arbitrate for the CAN bus, which have oscillators of opposite ends of the tolerance range and are separated by a long bus line; this is an example of a minor error in the bit timing configuration (Prop_Seg too short) that causes sporadic bus errors.

Some CAN implementations provide an optional 3-Sample Mode. The CAN module on this device does not. In this mode, the CAN bus input signal passes a digital low-pass filter, using three samples and a majority logic to determine the valid bit value. This results in an additional input delay of 1 t_q, requiring a longer Prop_Seg.

30.12.1.3 Phase Buffer Segments and Synchronization

The phase buffer segments (Phase_Seg1 and Phase_Seg2) and the synchronization jump width (SJW) are used to compensate for the oscillator tolerance.

The phase buffer segments surround the sample point. The phase buffer segments can be lengthened or shortened by synchronization.

The synchronization jump width (SJW) defines how far the resynchronizing mechanism can move the sample point inside the limits defined by the phase buffer segments to compensate for edge phase errors.

Synchronizations occur on edges from recessive to dominant. Their purpose is to control the distance between edges and sample points.

Edges are detected by sampling the actual bus level in each time quantum and comparing the sample with the bus level at the previous sample point. A synchronization can be done only if a recessive bit was sampled at the previous sample point and if the actual time quantum's bus level is dominant.

An edge is synchronous if the edge occurs inside of Sync_Seg; otherwise, the distance to the Sync_Seg is the edge phase error, measured in time quanta. If the edge occurs before Sync_Seg, the phase error is negative; else, the phase error is positive.

Two types of synchronization exist: hard synchronization and resynchronization. A hard synchronization is done once at the start of a frame; inside a frame, only resynchronization is possible.

- **Hard Synchronization:** After a hard synchronization, the bit time is restarted with the end of Sync_Seg, regardless of the edge phase error. Thus hard synchronization forces the edge which has caused the hard synchronization to lie within the synchronization segment of the restarted bit time.
- **Bit Resynchronization:** Resynchronization leads to a shortening or lengthening of the bit time such that the position of the sample point is shifted with regard to the edge.

  When the phase error of the edge that causes resynchronization is positive, Phase_Seg1 is lengthened. If the magnitude of the phase error is less than SJW, Phase_Seg1 is lengthened by the magnitude of the phase error; else, Phase_Seg1 is lengthened by SJW.

  When the phase error of the edge that causes resynchronization is negative, Phase_Seg2 is shortened. If the magnitude of the phase error is less than SJW, Phase_Seg2 is shortened by the magnitude of the phase error; else, Phase_Seg2 is shortened by SJW.

If the magnitude of the phase error of the edge is less than or equal to the programmed value of SJW, the results of hard synchronization and resynchronization are the same. If the magnitude of the phase error is larger than SJW, the resynchronization cannot compensate the phase error completely, and an error of (phase error - SJW) remains.

Only one synchronization can be done between two sample points. The synchronizations maintain a minimum distance between edges and sample points, giving the bus level time to stabilize and filtering out spikes that are shorter than (Prop_Seg + Phase_Seg1).

Apart from noise spikes, most synchronizations are caused by arbitration. All nodes synchronize "hard" on the edge transmitted by the "leading" transceiver that started transmitting first, but due to propagation delay times, the nodes cannot become ideally synchronized. The "leading" transmitter does not necessarily win the arbitration; therefore, the receivers have to synchronize themselves to different transmitters that subsequently "take the lead" and that are differently synchronized to the previously "leading" transmitter. The same happens at the acknowledge field, where the transmitter and some of the receivers have to synchronize to that receiver that "takes the lead" in the transmission of the dominant acknowledge bit.

Synchronizations after the end of the arbitration are caused by oscillator tolerance, when the differences in the oscillator's clock periods of transmitter and receivers sum up during the time between synchronizations (at most 10 bits). These summarized differences cannot be longer than the SJW, limiting the oscillator's tolerance range.

The examples in Figure 30-16 show how the phase buffer segments are used to compensate for phase errors. There are three drawings of each two consecutive bit timings. The upper drawing shows the synchronization on a "late" edge, the lower drawing shows the synchronization on an "early" edge, and the middle drawing is the reference without synchronization.

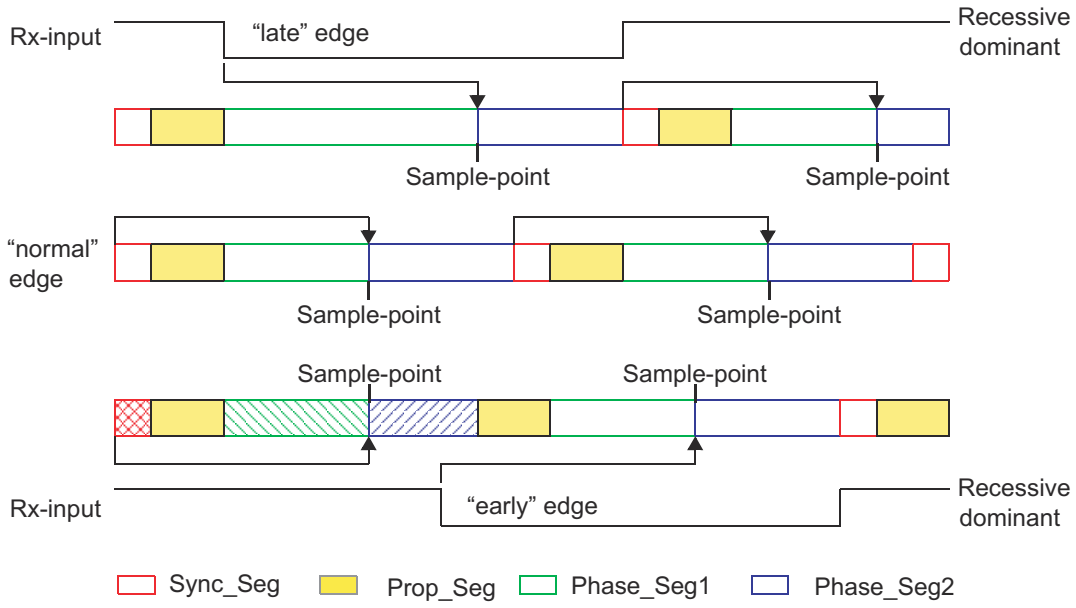


Figure 30-16. Synchronization on Late and Early Edges

In the first example, an edge from recessive to dominant occurs at the end of Prop_Seg. The edge is "late" since the edge occurs after the Sync_Seg. Reacting to the "late" edge, Phase_Seg1 is lengthened so that the distance from the edge to the sample point is the same as would have been from the Sync_Seg to the sample point if no edge had occurred. The phase error of this "late" edge is less than SJW, so it is fully compensated and the edge from dominant to recessive at the end of the bit, which is one nominal bit time long, occurs in the Sync_Seg.

In the second example, an edge from recessive to dominant occurs during Phase_Seg2. The edge is "early" since it occurs before a Sync_Seg. Reacting to the "early" edge, Phase_Seg2 is shortened and Sync_Seg is omitted, so that the distance from the edge to the sample point is the same as would have been from a Sync_Seg to the sample point if no edge had occurred. As in the previous example, the magnitude of this "early" edge's phase error is less than SJW, so it is fully compensated.

The phase buffer segments are lengthened or shortened temporarily only; at the next bit time, the segments return to their nominal programmed values.

In these examples, the bit timing is seen from the point of view of the CAN implementation's state machine, where the bit time starts and ends at the sample points. The state machine omits Sync_Seg when synchronizing on an "*early*" edge because the state machine cannot subsequently redefine that time quantum of Phase_Seg2 where the edge occurs to be the Sync_Seg.

The examples in Figure 30-17 show how short dominant noise spikes are filtered by synchronizations. In both examples, the spike starts at the end of Prop_Seg and has the length of (Prop_Seg + Phase_Seg1).

In the first example, the synchronization jump width is greater than or equal to the phase error of the spike's edge from recessive to dominant. Therefore the sample point is shifted after the end of the spike; a recessive bus level is sampled.

In the second example, SJW is shorter than the phase error, so the sample point cannot be shifted far enough; the dominant spike is sampled as actual bus level.

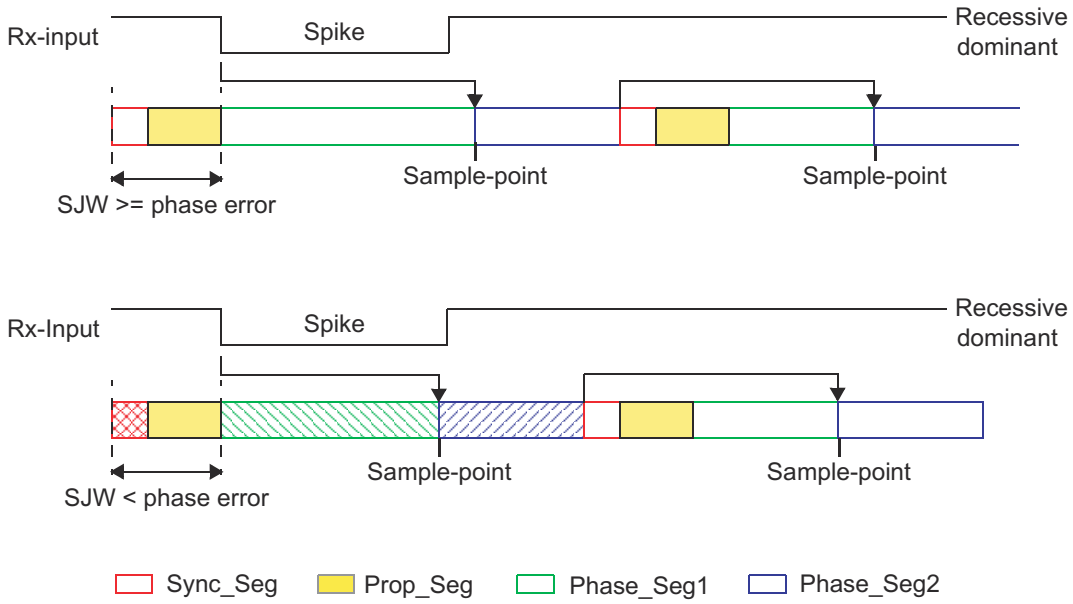


Figure 30-17. Filtering of Short Dominant Spikes

30.12.1.4 Oscillator Tolerance Range

With the introduction of CAN protocol version 1.2, the option to synchronize on edges from dominant to recessive became obsolete. Only edges from recessive to dominant are considered for synchronization. The protocol update to version 2.0 (A and B) had no influence on the oscillator tolerance.

The tolerance range df for an oscillator's frequency f_{osc} around the nominal frequency f_{nom} with:

$$(1 - df) * f_{nom} \leq f_{osc} \leq (1 + df) * f_{nom}$$

depends on the proportions of Phase_Seg1, Phase_Seg2, SJW, and the bit time. The maximum tolerance df is defined by two conditions (both shall be met):

$$df \leq \frac{\min(Tseg1, Tseg2)}{2((13 \times bit\ time) - Tseg2)}$$

$$df \leq \frac{SJW}{20 \times bit\_time}$$

It has to be considered that SJW cannot be larger than the smaller of the phase buffer segments and that the propagation time segment limits that are part of the bit time can be used for the phase buffer segments.

The combination Prop_Seg = 1 and Phase_Seg1 = Phase_Seg2 = SJW = 4 allows the largest possible oscillator tolerance of 1.58%. This combination with a Propagation Time Segment of only 10% of the bit time is not suitable for short bit times; it can be used for bit rates of up to 125 kBit/s (bit time = 8 µs) with a bus length of 40 m.

30.12.2 Configuration of the CAN Bit Timing

In the CAN, the bit timing configuration is programmed in two register bytes, additionally a third byte for a baud rate prescaler extension of 4 bits (BRPE) is provided. The sum of Prop_Seg and Phase_Seg1 (as TSEG1) is combined with Phase_Seg2 (as TSEG2) in one byte, SJW and BRP (plus BRPE in third byte) are combined in the other byte (see Figure 30-18).

In this bit timing register, the components TSEG1, TSEG2, SJW and BRP have to be programmed to a numerical value that is one less than the functional value; so instead of values in the range of [1…n], values in the range of [0…n-1] are programmed. That way, for example, SJW (functional range of [1…4]) is represented by only two bits.

Therefore the length of the bit time is either:

- (programmed values) [TSEG1 + TSEG2 + 3] t_q
- (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] t_q

The data in the Bit Timing Register is the configuration input of the CAN protocol controller. The baud rate prescaler (configured by BRPE/BRP) defines the length of the time quantum (the basic time unit of the bit time); the bit timing logic (configured by TSEG1, TSEG2, and SJW) defines the number of time quanta in the bit time.

The processing of the bit time, the calculation of the position of the Sample Point, and occasional synchronizations are controlled by the Bit timing state machine, which is evaluated once each time quantum. The rest of the CAN protocol controller, the Bit Stream Processor (BSP) state machine, is evaluated once each bit time, at the Sample Point.

The Shift register serializes the messages to be sent and parallelizes received messages. Loading and shifting is controlled by the BSP.

The BSP translates messages into frames and conversely. The BSP generates and discards the enclosing fixed format bits, inserts and extracts stuff bits, calculates and checks the CRC code, performs the error management, and decides which type of synchronization is to be used. It is evaluated at the sample point and processes the sampled bus input bit. The time after the sample point that is needed to calculate the next bit to be sent (for example, data bit, CRC bit, stuff bit, error flag, or idle) is called the Information Processing Time (IPT), which is 0 t_q for the CAN.

Generally, the IPT is CAN controller specific, but cannot be longer than 2 t_q. The IPT length is the lower limit of the programmed length of Phase_Seg2. In case of a synchronization, Phase_Seg2 can be shortened to a value less than IPT, which does not affect bus timing.

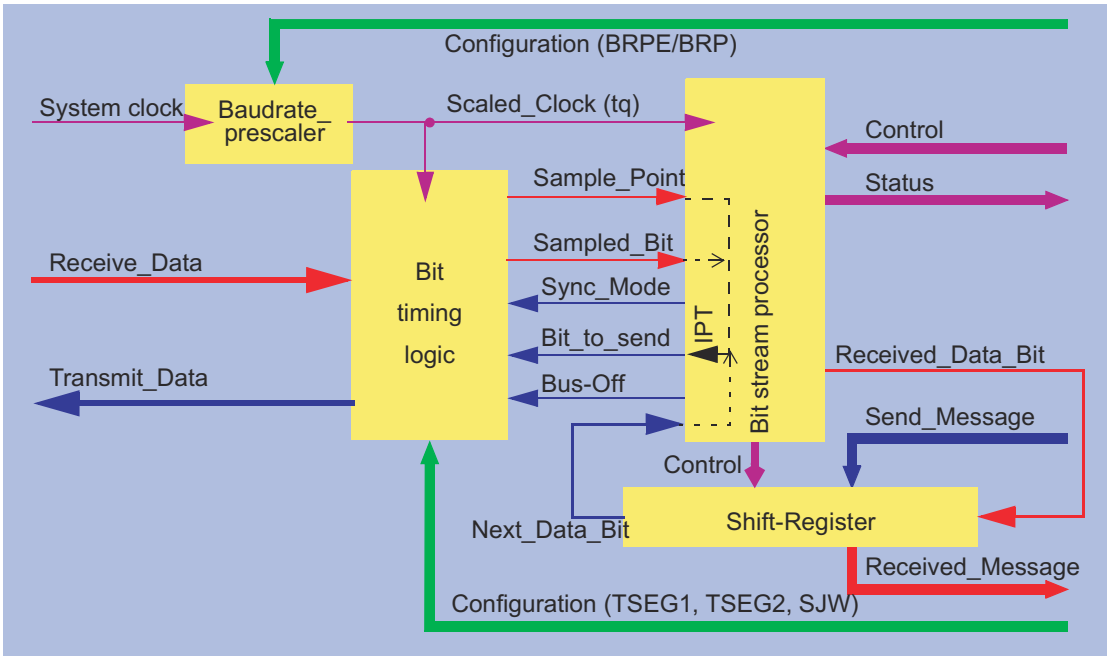


Figure 30-18. Structure of the CAN Core's CAN Protocol Controller

30.12.2.1 Calculation of the Bit Timing Parameters

Usually, the calculation of the bit timing configuration starts with a desired bit rate or bit time. The resulting Bit time (1/Bit rate) must be an integer multiple of the CAN clock period.

Note

8 MHz is the minimum CAN clock frequency required to operate the CAN at a bit rate of 1 MBit/s.

The bit time can consist of 8 to 25 time quanta. The length of the time quantum t_q is defined by the Baud Rate Prescaler with t_q = (Baud Rate Prescaler) / CAN_CLK. Several combinations can lead to the desired bit time, allowing iterations of the following steps.

The first part of the bit time to be defined is the Prop_Seg. The length depends on the delay times measured in the system. A maximum bus length as well as a maximum node delay has to be defined for expandable CAN bus systems. The resulting time for Prop_Seg is converted into time quanta (rounded up to the nearest integer multiple of t_q).

The Sync_Seg is 1 t_q long (fixed), leaving (bit time - Prop_Seg - 1) t_q for the two Phase Buffer Segments. If the number of remaining t_q is even, the Phase Buffer Segments have the same length, Phase_Seg2 = Phase_Seg1; else, Phase_Seg2 = Phase_Seg1 + 1.

The minimum nominal length of Phase_Seg2 has to be regarded as well. Phase_Seg2 cannot be shorter than any CAN controller Information Processing Time in the network, which is device dependent and can be in the range of [0 to 2] t_q.

The length of the synchronization jump width is set to the maximum value, which is the minimum of 4 and Phase_Seg1.

The oscillator tolerance range necessary for the resulting configuration is calculated by the formulas given in Section 30.12.1.4.

If more than one configurations are possible to reach a certain bit rate, it is recommended to choose the configuration that allows the highest oscillator tolerance range.

CAN nodes with different clocks require different configurations to come to the same bit rate. The calculation of the propagation time in the CAN network, based on the nodes with the longest delay times, is done once for the whole network.

The CAN system oscillator tolerance range is limited by the node with the lowest tolerance range.

The calculation can show that bus length or bit rate has to be decreased or that the oscillator frequency stability has to be increased to find a protocol compliant configuration of the CAN bit timing.

The resulting configuration is written into the Bit Timing register:

(Phase_Seg2-1)&(Phase_Seg1+Prop_Seg-1)&(SynchronizationJumpWidth-1)&(Prescaler-1)

30.12.2.2 Example for Bit Timing at High Baudrate

In this example, the frequency of CAN_CLK is 10 MHz, BRP is 0, the bit rate is 1 MBit/s.

t_q	100	ns	=	t_{CAN_CLK}
delay of bus driver	90	ns	=	
delay of receiver circuit	40	ns	=	
delay of bus line (40m)	220	ns	=	
t_{Prop}	700	ns	=	2*delays = 7 • t_q
t_{SJW}	100	ns	=	1 • t_q
t_{TSeg1}	800	ns	=	t_{Prop} + t_{SJW}
t_{TSeg2}	100	ns	=	Information Processing Time + 1 • t_q
$t_{Sync-Seg}$	100	ns	=	1 • t_q
bit time	1000	ns	=	$t_{Sync-Seg}$ + t_{TSeg1} + t_{TSeg2}
tolerance for CAN_CLK	0.35	%	=	$\frac{\min(Tseg1, Tseg2)}{2((13 \times bit\ time) - Tseg2)}$ $= \frac{0.1\,\mu s}{2((13 \times 1\,\mu s) - 0.1\,\mu s)}$

In this example, the concatenated bit time parameters are $(1\text{-}1)_3$&$(8\text{-}1)_4$&$(1\text{-}1)_2$&$(1\text{-}1)_6$, so the Bit Timing Register is programmed to = 0x00000700.

30.12.2.3 Example for Bit Timing at Low Baudrate

In this example, the frequency of CAN_CLK is 2 MHz, BRP is 1, the bit rate is 100 KBit/s.

t_q	1	µs	=	2 • t_{CAN_CLK}
delay of bus driver	200	ns	=	
delay of receiver circuit	80	ns	=	
delay of bus line (40m)	220	ns	=	
t_{Prop}	1	µs	=	1 • t_q
t_{SJW}	4	µs	=	4 • t_q
t_{TSeg1}	5	µs	=	t_{Prop} + t_{SJW}
t_{TSeg2}	4	µs	=	Information Processing Time + 4 • t_q
$t_{Sync-Seg}$	1	µs	=	1 • t_q
bit time	10	µs	=	$t_{Sync-Seg}$ + t_{TSeg1} + t_{TSeg2}
tolerance for CAN_CLK	1.58	%	=	$\frac{\min(Tseg1, Tseg2)}{2((13 \times bit\ time) - Tseg2)}$ $= \frac{4\,\mu s}{2((13 \times 10\,\mu s) - 4\,\mu s)}$

In this example, the concatenated bit time parameters are $(4\text{-}1)_3$&$(5\text{-}1)_4$&$(4\text{-}1)_2$&$(2\text{-}1)_6$, so the Bit Timing register is programmed to = 0x000034C1.

30.13 Message Interface Register Sets

The interface register sets control the CPU read and write accesses to the Message RAM. There are two interface register sets for read and write access (IF1 and IF2) and one Interface Register Set for read access only (IF3).

Due to the structure of the Message RAM, it is not possible to change single bits or bytes of a message object. Instead, always a complete message object in the Message RAM is accessed. Therefore the data transfer from the IF1/IF2 registers to the Message RAM requires the message handler to perform a read-modifywrite cycle. First those parts of the message object that are not to be changed are read from the Message RAM into the Interface Register set, and after the update the whole content of the Interface Register set is written into the message object.

After the partial write of a message object, those parts of the Interface Register set that are not selected in the Command Register are set to the actual contents of the selected message object. After the partial read of a message object, those parts of the Interface Register set that are not selected in the Command Register are left unchanged.

By buffering the data to be transferred, the Interface Register sets avoid conflicts between concurrent CPU accesses to the Message RAM and CAN message reception and transmission. A complete message object (see Section 30.14.1) or parts of the message object can be transferred between the Message RAM and the IF1/IF2 Register set in one single transfer. This transfer, performed in parallel on all selected parts of the message object, maintains the data consistency of the CAN message.

There is one condition that can cause a write access to the message RAM to be lost. If MsgVal = 1 for the message object that is accessed and CAN communication is ongoing, a transfer from the IFx register to message RAM can be lost. The reason for this might happen is that the IFx register write to the message RAM occurs in between a read-modify-write access of the Host Message Handler when it is in the process of receiving a message for the same message object.

To avoid this issue with receive mail boxes, reset MsgVal before changing any of the following: Id28-0, Xtd, Dir, DLC3-0, RxIE, TxIE, RmtEn, EoB, Umask, Msk28-0, MXtd, and MDir.

To avoid this issue with transmit mail boxes, reset MsgVal before changing any of the following: Dir, RxIE, TxIE, RmtEn, EoB, Umask, Msk28-0, MXtd, and MDir. Other fields not listed above, like Data, can be changed without fear of losing a write to the message RAM.

30.13.1 Message Interface Register Sets 1 and 2 (IF1 and IF2)

The IF1 and IF2 register sets allow data transfers to and from the message objects. The IFxCMD register for an interface control the direction of the data transfer. If the IFxCMD register is set to write, then the message object fields selected by the IFxCMD register are overwritten by values taken from the other IFx registers. If the IFxCMD register is set to read, then the message object fields selected by the IFxCMD register is copied from the message object to the other IFx registers. The interfaces allow for transfers of a complete message object as well as individual parts. The transfer begins with the desired message object number is written to bits 7:0 of the IFxCMD register.

When the CPU initiates a data transfer between the IF1/IF2 registers and Message RAM, the message handler sets the Busy bit in the respective Command Register to 1. After the transfer has completed, the Busy bit is set back to 0 (see Figure 30-19).

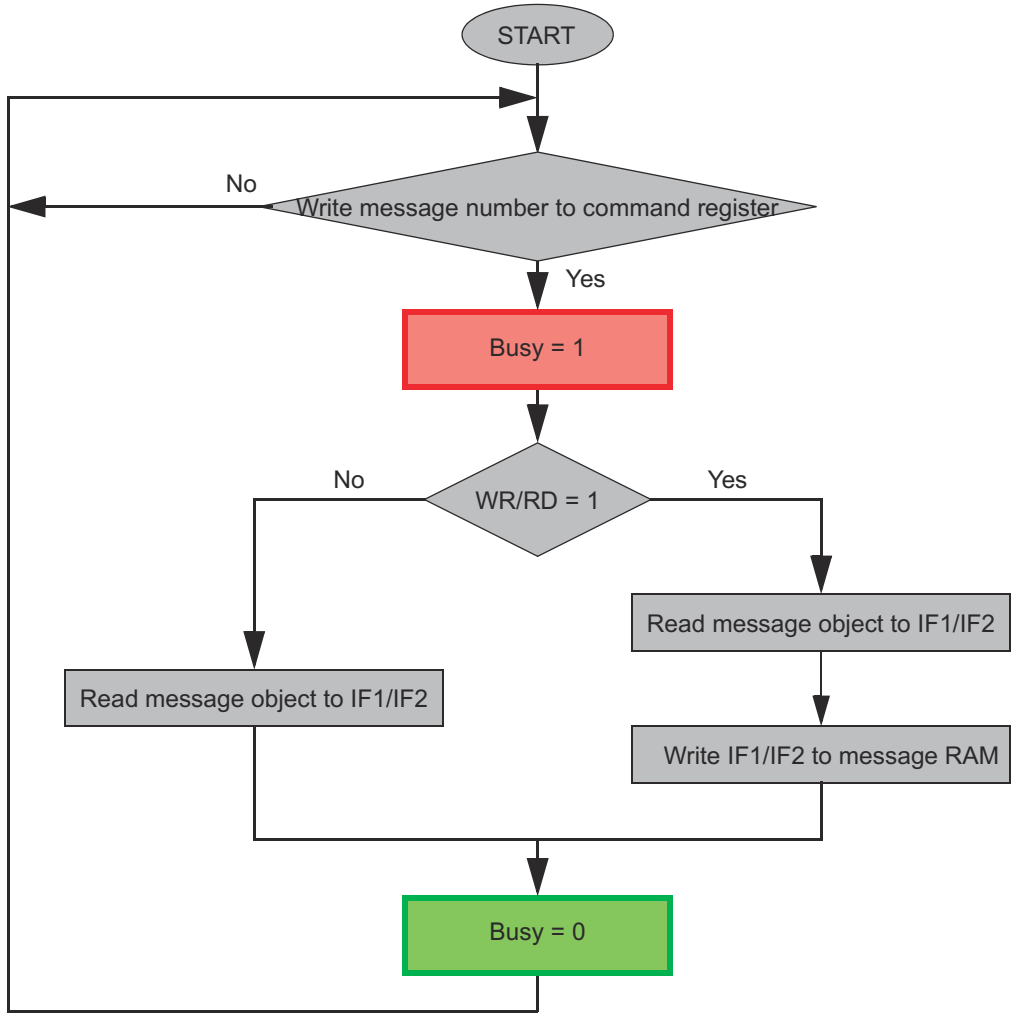


Figure 30-19. Data Transfer Between IF1 / IF2 Registers and Message RAM

30.13.2 Message Interface Register Set 3 (IF3)

The IF3 register set can automatically be updated with received message objects without the need to initiate the transfer from Message RAM by CPU. The automatic update functionality can be programmed for each message object (see the IF3 Update Enable register).

All valid message objects in Message RAM that are configured for automatic update are checked for active NewDat flags. If such a message object is found, the message objects are transferred to the IF3 register (if no previous DMA transfers are ongoing), controlled by IF3 Observation register. If more than one NewDat flag is active, the message object with the lowest number has the highest priority for automatic IF3 update.

The NewDat bit in the message object is reset by a transfer to IF3.

If DCAN internal IF3 update is complete, a DMA is requested. The DMA request stays active until the first read access to one of the IF3 registers. The DMA functionality has to be enabled by setting bit DE3 in the CAN Control register.

Note

The IF3 register set cannot be used for transferring data into message objects.

30.14 Message RAM

The CAN Message RAM contains message objects and parity bits for the message objects. There are 32 message objects in the Message RAM.

During normal operation, accesses to the Message RAM are performed using the Interface Register sets, and the CPU cannot directly access the Message RAM.

The Interface Register sets IF1 and IF2 provide indirect read/write access from the CPU to the Message RAM. The IF1 and IF2 register sets can buffer control and user data to be transferred to and from the message objects.

The third Interface Register set IF3 can be configured to automatically receive control and user data from the Message RAM when a message object has been updated after reception of a CAN message. The CPU does not need to initiate the transfer from Message RAM to IF3 Register set.

The message handler avoids potential conflicts between concurrent accesses to Message RAM and CAN frame reception/transmission.

The message RAM can only be accessed in debug mode. The message RAM base address is 0x1000 above the base address of the CAN peripheral.

30.14.1 Structure of Message Objects

Figure 30-20 shows the structure of a message object.

The grayed fields are those parts of the message object which are represented in dedicated registers. For example, the transmit request flags of all message objects are represented in centralized transmit request registers.

Figure 30-20. Structure of a Message Object

Message Object												
UMask	Msk[28:0]	MXtd	MDir	EoB	unused	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
MsgVal	ID[28:0]	Xtd	Dir	DLC[3:0]	Data 0	Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7

Table 30-5. Message Object Field Descriptions

Name	Value	Description
MsgVal		Message valid
	0	The message object is ignored by the message handler.
	1	The message object is to be used by the message handler.
		Note: This bit can be kept at level 1 even when the identifier bits ID[28:0], the control bits Xtd, Dir, or the data length code DLC[3:0] are changed.
UMask		Use Acceptance Mask
	0	Mask bits (Msk[28:0], MXtd and MDir) are ignored and not used for acceptance filtering.
	1	Mask bits are used for acceptance filtering.
		Note: If the UMask bit is set to 1, the message object's mask bits have to be programmed during initialization of the message object before MsgVal is set to 1.
ID[28:0]		Message Identifier
	ID[28:0]	29-bit ("extended") identifier bits
	ID[28:18]	11-bit ("standard") identifier bits

Table 30-5. Message Object Field Descriptions (continued)

Name	Value	Description
Msk[28:0]		Identifier Mask
	0	The corresponding bit in the message identifier is not used for acceptance filtering (don't care).
	1	The corresponding bit in the message identifier is used for acceptance filtering.
		Note: The bit functionality in the DCAN module is the opposite of the Local Acceptance Mask bit functionality in the eCAN module found in older C28x devices, where a 1 means the corresponding bit is not used for filtering, and a 0 means the bit is used.
Xtd		Extended Identifier
	0	The 11-bit ("standard") identifier is used for this message object.
	1	The 29-bit ("extended") identifier is used for this message object.
MXtd		Mask Extended Identifier
	0	The extended identifier bit (IDE) has no effect on the acceptance filtering.
	1	The extended identifier bit (IDE) is used for acceptance filtering.
		Note: When 11-bit ("standard") Identifiers are used for a message object, the identifiers of received data frames are written into bits ID[28:18]. For acceptance filtering, only these bits together with mask bits Msk[28:18] are considered.
Dir		Message Direction
	0	Direction = receive: On TxRqst, a remote frame with the identifier of this message object is transmitted. On reception of a data frame with matching identifier, the message is stored in this message object.
	1	Direction = transmit: On TxRqst, a data frame is transmitted. On reception of a remote frame with matching identifier, the TxRqst bit of this message object is set (if RmtEn = 1).
MDir		Mask Message Direction
	0	The message direction bit (Dir) has no effect on the acceptance filtering.
	1	The message direction bit (Dir) is used for acceptance filtering.
EOB		End of Block
	0	The message object is part of a FIFO Buffer block and is not the last message object of this FIFO Buffer block.
	1	The message object is a single message object or the last message object in a FIFO Buffer Block.
		Note: This bit is used to concatenate multiple message objects to build a FIFO Buffer. For single message objects (not belonging to a FIFO Buffer), this bit must always be set to 1.
NewDat		New Data
	0	No new data has been written into the data bytes of this message object by the message handler since the last time when this flag was cleared by the CPU.
	1	The message handler or the CPU has written new data into the data bytes of this message object.
MsgLst		Message Lost (only valid for Message Objects with direction = receive)
	0	No message was lost since the last time when this bit was reset by the CPU.
	1	The message handler stored a new message into this message object when NewDat was still set, so the previous message has been overwritten.
RxIE		Receive Interrupt Enable
	0	IntPnd is not triggered after the successful reception of a frame.
	1	IntPnd is triggered after the successful reception of a frame.
TxIE		Transmit Interrupt Enable
	0	IntPnd is not triggered after the successful transmission of a frame.
	1	IntPnd is triggered after the successful transmission of a frame.
IntPnd		Interrupt Pending
	0	This message object is not the source of an interrupt.
	1	This message object is the source of an interrupt. The Interrupt Identifier in the Interrupt Register point to this message object, if there is no other interrupt source with higher priority.

Table 30-5. Message Object Field Descriptions (continued)

Name	Value	Description
RmtEn		Remote Enable
	0	At the reception of a remote frame, TxRqst is not changed.
	1	At the reception of a remote frame, TxRqst is set.
		Note: See Section 30.11.8 for details on the setup of RmtEn and UMask for remote frames.
TxRqst		Transmit Request
	0	This message object is not waiting for a transmission.
	1	The transmission of this message object is requested and is not yet done.
DLC[3:0]		Data length code
	0-8	Data frame has 0-8 data bytes.
	9-15	Data frame has 8 data bytes.
		Note: The data length code of a message object must be defined to the same value as in the corresponding objects with the same identifier at other nodes. When the message handler stores a data frame, the message handler writes the DLC to the value given by the received message.
Data 0		1st data byte of a CAN data frame
Data 1		2nd data byte of a CAN data frame
Data 2		3rd data byte of a CAN data frame
Data 3		4th data byte of a CAN data frame
Data 4		5th data byte of a CAN data frame
Data 5		6th data byte of a CAN data frame
Data 6		7th data byte of a CAN data frame
Data 7		8th data byte of a CAN data frame
		Note: Byte Data 0 is the first data byte shifted into the shift register of the CAN core during a reception, byte Data 7 is the last. When the message handler stores a data frame, the message handler writes all the eight data bytes into a message object. If the data length code is less than 8, the remaining bytes of the message object can be overwritten by undefined values.

30.14.2 Addressing Message Objects in RAM

The starting location of a particular message object in RAM is:

Message RAM base address + (message object number) * 0x20

This means that Message Object 1 starts at offset 0x0020; Message Object 2 starts at offset 0x0040, and so on.

Note

A 0 is not a valid message object number. At address 0x0000, the last message object (32) (with the lowest priority) is located. Writing to the address of an unimplemented message object can overwrite an implemented message object.

Message Object number 1 has the highest priority.

Table 30-6. Message RAM Addressing in Debug Mode

Message Object Number	Offset From Base Address	Word Number	Debug Mode(1)
last implemented (here:32)	0x0000	1	Parity
	0x0004	2	MXtd,MDir,Mask
	0x0008	3	Xtd,Dir,ID
	0x000C	4	Ctrl
	0x0010	5	Data Bytes 3-0
	0x0014	6	Data Bytes 7-4
1	0x0020	1	Parity
	0x0024	2	MXtd,MDir,Mask
	0x0028	3	Xtd,Dir,ID
	0x002C	4	Ctrl
	0x0030	5	Data Bytes 3-0
	0x0034	6	Data Bytes 7-4
2	0x0040	1	Parity
	0x0044	2	MXtd,MDir,Mask
	0x0048	3	Xtd,Dir,ID
	0x004C	4	Ctrl
	0x0050	5	Data Bytes 3-0
	0x0054	6	Data Bytes 7-4
…	…	…	…
31	0x03E0	1	Parity
	0x03E4	2	MXtd,MDir,Mask
	0x03E8	3	Xtd,Dir,ID
	0x03EC	4	Ctrl
	0x03F0	5	Data Bytes 3-0
	0x03F4	6	Data Bytes 7-4

(1) See Section 30.14.3.

30.14.3 Message RAM Representation in Debug Mode

In debug mode, the Message RAM is memory-mapped. This allows the external debug unit to access the Message RAM.

Note

During debug mode, the Message RAM cannot be accessed using the IFx register sets.

Figure 30-21. Message RAM Representation in Debug Mode

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

MsgAddr + 0x00

Reserved	
Reserved	Parity[4:0]

MsgAddr + 0x04

MXtd	MDir	Rsvd	Msk[28:16]
Msk[15:0]			

MsgAddr + 0x08

Rsvd	Xtd	Dir	ID[28:16]
ID[15:0]			

MsgAddr + 0x0C

Reserved										
Rsvd	MsgLst	Rsvd	UMask	TxIE	RxIE	RmtEn	Rsvd	EOB	Reserved	DLC[3:0]

MsgAddr + 0x10

Data 3	Data 2
Data 1	Data 0

MsgAddr + 0x14

Data 7	Data 6
Data 5	Data 4

30.15 Software

30.15.1 CAN Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/can

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

30.15.1.1 CAN External Loopback

FILE: can_ex1_loopback.c

This example shows the basic setup of CAN in order to transmit and receive messages on the CAN bus. The CAN peripheral is configured to transmit messages with a specific CAN ID. A message is then transmitted once per second, using a simple delay loop for timing. The message that is sent is a 2 byte message that contains an incrementing pattern.

This example sets up the CAN controller in External Loopback test mode. Data transmitted is visible on the CANTXA pin and is received internally back to the CAN Core. Please refer to details of the External Loopback Test Mode in the CAN Chapter in the Technical Reference Manual. Refer to *Programming Examples and Debug Strategies for the DCAN Module* for useful information about this example

External Connections

- None.

Watch Variables

- msgCount - A counter for the number of successful messages received
- txMsgData - An array with the data being sent
- rxMsgData - An array with the data that was received

30.15.1.2 CAN Loopback - CM

FILE: can_ex1_polling.c

This example shows the basic setup of CAN in order to transmit and receive messages on the CAN bus. The CAN peripheral is configured to transmit messages with a specific CAN ID. The message that is sent is a 2 byte message that contains an incrementing pattern.

This example sets up the CAN controller in External Loopback test mode. Data transmitted is visible on the CANTXA pin and is received internally back to the CAN Core. Please refer to details of the External Loopback Test Mode in the CAN Chapter in the Technical Reference Manual.

Before running this example, please run the can_config_c28x example from the c28x folder. It will initialize the clock, configure the GPIOs and allocate CAN A to CM.

External Connections

- None.

Watch Variables

- msgCount - A counter for the number of successful messages received.
- rxMsgData - An array with the data that was received.
- txMsgData - An array with the data being sent.

30.15.1.3 CAN External Loopback with Interrupts

FILE: can_ex2_loopback_interrupts.c

This example shows the basic setup of CAN in order to transmit and receive messages on the CAN bus. The CAN peripheral is configured to transmit messages with a specific CAN ID. A message is then transmitted once per second, using a simple delay loop for timing. The message that is sent is a 4 byte message that contains an incrementing pattern. A CAN interrupt handler is used to confirm message transmission and count the number of messages that have been sent.

This example sets up the CAN controller in External Loopback test mode. Data transmitted is visible on the CANTXA pin and is received internally back to the CAN Core. Please refer to details of the External Loopback Test Mode in the CAN Chapter in the Technical Reference Manual. Refer to *Programming Examples and Debug Strategies for the DCAN Module* for useful information about this example

External Connections

- None.

Watch Variables

- txMsgCount - A counter for the number of messages sent
- rxMsgCount - A counter for the number of messages received
- txMsgData - An array with the data being sent
- rxMsgData - An array with the data that was received
- errorFlag - A flag that indicates an error has occurred

30.15.1.4 CAN External Loopback with Interrupts - CM

FILE: can_ex2_loopback_interrupts.c

This example shows the basic setup of CAN in order to transmit and receive messages on the CAN bus. The CAN peripheral is configured to transmit messages with a specific CAN ID. A message is then transmitted once per second, using a simple delay loop for timing. The message that is sent is a 4 byte message that contains an incrementing pattern. A CAN interrupt handler is used to confirm message transmission and count the number of messages that have been sent.

This example sets up the CAN controller in External Loopback test mode. Data transmitted is visible on the CANTXA pin and is received internally back to the CAN Core. Please refer to details of the External Loopback Test Mode in the CAN Chapter in the Technical Reference Manual.

Before running this example, please run the can_config_c28x example from the c28x folder. It will initialize the clock, configure the GPIOs and allocate CAN A to CM.

External Connections

- None.

Watch Variables

- txMsgCount - A counter for the number of messages sent
- rxMsgCount - A counter for the number of messages received
- txMsgData - An array with the data being sent
- rxMsgData - An array with the data that was received
- errorFlag - A flag that indicates an error has occurred

30.15.1.5 CAN-A to CAN-B External Transmit

FILE: can_ex3_external_transmit.c

This example initializes CAN module A and CAN module B for external communication. CAN-A module is setup to transmit incrementing data for "n" number of times to the CAN-B module, where "n" is the value of TXCOUNT. CAN-B module is setup to trigger an interrupt service routine (ISR) when data is received. An error flag will be set if the transmitted data doesn't match the received data.

Both CAN modules on the device need to be connected to each other via CAN transceivers. *Hardware Required*

- A C2000 board with two CAN transceivers

External Connections

- ControlCARD CANA is on DEVICE_GPIO_PIN_CANTXA (CANTXA)
- and DEVICE_GPIO_PIN_CANRXA (CANRXA)
- ControlCARD CANB is on DEVICE_GPIO_PIN_CANTXB (CANTXB)
- and DEVICE_GPIO_PIN_CANRXB (CANRXB)

Watch Variables

- TXCOUNT - Adjust to set the number of messages to be transmitted
- txMsgCount - A counter for the number of messages sent
- rxMsgCount - A counter for the number of messages received
- txMsgData - An array with the data being sent
- rxMsgData - An array with the data that was received
- errorFlag - A flag that indicates an error has occurred

30.15.1.6 CAN-A to CAN-B External Transmit - CM

FILE: can_ex3_external_transmit.c

This example initializes CAN module A and CAN module B for external communication. CAN-A module is setup to transmit incrementing data for "n" number of times to the CAN-B module, where "n" is the value of TXCOUNT. CAN-B module is setup to trigger an interrupt service routine (ISR) when data is received. An error flag will be set if the transmitted data doesn't match the received data.

Before running this example, please run the can_config_c28x example from the c28x folder. It will initialize the clock, configure the GPIOs and allocate CAN A and CAN B to CM.

Both CAN modules on the device need to be connected to each other via CAN transceivers. *Hardware Required*

- A C2000 board with two CAN transceivers

External Connections

- ControlCARD CANA is on GPIO37 (CANTXA) and GPIO36 (CANRXA)
- ControlCARD CANB is on GPIO12 (CANTXB) and GPIO10 (CANRXB)

Watch Variables

- TXCOUNT - Adjust to set the number of messages to be transmitted
- txMsgCount - A counter for the number of messages sent
- rxMsgCount - A counter for the number of messages received
- txMsgData - An array with the data being sent
- rxMsgData - An array with the data that was received
- errorFlag - A flag that indicates an error has occurred

30.15.1.7 CAN External Loopback with DMA

FILE: can_ex4_loopback_dma.c

This example sets up the CAN module to transmit and receive messages on the CAN bus. The CAN module is set to transmit a 4 byte message internally. An interrupt is used to assert the DMA request line which then triggers the DMA to transfer the received data from the CAN interface register to the receive buffer array. A data check is performed once the transfer is complete.

This example sets up the CAN controller in External Loopback test mode. Data transmitted is visible on the CANTXA pin and is received internally back to the CAN Core. Please refer to details of the External Loopback Test Mode in the CAN Chapter in the Technical Reference Manual. Please refer to the appnote Programming Examples and Debug Strategies for the DCAN Module (www.ti.com/lit/SPRACE5) for useful information about this example

External Connections

- None.

Watch Variables

- txMsgCount - A counter for the number of messages sent
- rxMsgCount - A counter for the number of messages received
- txMsgData - An array with the data being sent
- rxMsgData - An array with the data that was received

30.15.1.8 CAN Transmit and Receive Configurations - CM

FILE: can_ex4_transmit_receive.c

This example shows the basic setup of CAN in order to transmit or receive messages on the CAN bus with a specific Message ID. The CAN Controller is configured according to the selection of the define.

When the TRANSMIT define is selected, the CAN Controller acts as a Transmitter and sends data to the second CAN Controller connected externally.If TRANMSIT is not defined the CAN Controller acts as a Receiver and waits for message to be transmitted by the External CAN Controller.

Before running this example, please run the can_config_c28x example from the c28x folder. It will initialize the clock, configure the GPIOs and allocate CAN A to CM.

CAN modules on the device need to be connected to via CAN transceivers. *Hardware Required*

- A C2000 board with CAN transceiver.

External Connections

- ControlCARD CANA is on GPIO37 (CANTXA) and GPIO36 (CANRXA)

Watch Variables Transmit \Configuration

- MSGCOUNT - Adjust to set the number of messages
- txMsgCount - A counter for the number of messages sent
- txMsgData - An array with the data being sent
- errorFlag - A flag that indicates an error has occurred
- rxMsgCount - Has the initial value as No. of Messages to be received and decrements with each message.

30.15.1.9 CAN Transmit and Receive Configurations

FILE: can_ex5_transmit_receive.c

This example shows the basic setup of CAN in order to transmit or receive messages on the CAN bus with a specific Message ID. The CAN Controller is configured according to the selection of the define.

When the TRANSMIT define is selected, the CAN Controller acts as a Transmitter and sends data to the second CAN Controller connected externally.If TRANMSIT is not defined the CAN Controller acts as a Receiver and waits for message to be transmitted by the External CAN Controller. Refer to *Programming Examples and Debug Strategies for the DCAN Module* for useful information about this example

CAN modules on the device need to be connected to via CAN transceivers. *Hardware Required*

- A C2000 board with CAN transceiver.

External Connections

- ControlCARD CANA is on DEVICE_GPIO_PIN_CANTXA (CANTXA)
- and DEVICE_GPIO_PIN_CANRXA (CANRXA)

Watch Variables Transmit \Configuration

- MSGCOUNT - Adjust to set the number of messages
- txMsgCount - A counter for the number of messages sent
- txMsgData - An array with the data being sent
- errorFlag - A flag that indicates an error has occurred
- rxMsgCount - Has the initial value as No. of Messages to be received and decrements with each message.

30.15.1.10 CAN Error Generation Example

FILE: can_ex6_error_generation.c

This example demonstrates the ways of handling CAN Error conditions It generates the CAN Packets and sends them over GPIO It is looped back externally to be received in CAN module The CAN Interrupt service routine reads the Error status and demonstrates how different Error conditions can be detected

Change ERR_CFG define to the different Error Scenarios and run the example. The corresponding Error Flag will be set in status variable of canaISR() routine. Uses a CPU Timer(Timer 0) for periodic timer interrupt of CANBITRATE uSec On the Timer interrupt it sends the required CAN Frame type with the specified error conditions CAN modules on the device need to be connected to via CAN transceivers. Please refer to

the application note titled "Configurable Error Generator for Controller Area Network" at *Configurable Error Generator for Controller Area Network* for further details on this example

External Connections

- ControlCARD GPIOTX_PIN should be connected to
- DEVICE_GPIO_PIN_CANRXA(CANRXA)

Watch Variables Transmit \Configuration

- status - variable in canaISR for checking error Status

30.15.1.11 CAN Remote Request Loopback

FILE: can_ex7_loopback_tx_rx_remote_frame.c

This example shows the basic setup of CAN in order to transmit a remote frame and get a response for the remote frame and store it in a receive Object. The CAN peripheral is configured to transmit remote request frame and a remote answer frame messages with a specific CAN ID. Message object 3 is configured to transmit a remote request. Message object 2 is configured as a remote answer object with filter mask such that it accepts remote frame with any message ID and transmit's remote answer with message ID 7 and data length 8. Message object 1 is configured as a received object with filter message ID 7 so as to store the remote answer data transmitted by message object 2.

This example sets up the CAN controller in External Loopback test mode. Data transmitted is visible on the CANTXA pin and is received internally back to the CAN Core. Please refer to details of the External Loopback Test Mode in the CAN Chapter in the Technical Reference Manual.

External Connections

- None.

Watch Variables

- txMsgData - An array with the data being sent
- rxMsgData - An array with the data that was received

30.15.1.12 CAN example that illustrates the usage of Mask registers

FILE: can_ex8_mask.c

This example initializes CAN module A for Reception. When a frame with a matching filter criterion is received, the data will be copied in mailbox 1 and LED will be toggled a few times and the code gets ready for the next frame. If a message of any other MSGID is received, an ACK will be provided Completion of reception is determined by polling CAN_NDAT_21 register. No interrupts are used. Refer to *Programming Examples and Debug Strategies for the DCAN Module* for useful information about this example

Hardware Required

- An external CAN node that transmits to CAN-A on the C2000 MCU

Watch Variables

- rxMsgCount - A counter for the number of messages received
- rxMsgData - An array with the data that was received

30.16 CAN Registers

This section describes the Controller Area Network registers.

30.16.1 CAN Base Address Table (C28)

Table 30-7. CAN Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
CanaRegs	CAN_REGS	CANA_BASE	0x0004_8000	YES	YES	YES	-	YES
CanbRegs	CAN_REGS	CANB_BASE	0x0004_A000	YES	YES	YES	-	YES

30.16.2 CM CAN Base Address Table (CM)

Table 30-8. CM CAN Base Address Table (CM)

DriverLib Name	Base Address	µDMA Access	Ethernet DMA Access
CANA_BASE	0x4007_0000	-	-
CANB_BASE	0x4007_4000	-	-

30.16.3 CAN_REGS Registers

Table 30-9 lists the memory-mapped registers for the CAN_REGS registers. All register offset addresses not listed in Table 30-9 should be considered as reserved locations and the register contents should not be modified.

Table 30-9. CAN_REGS Registers

Offset (x8)	Offset (x16)	Acronym	Register Name	Write Protection	Section
0h	0h	CAN_CTL	CAN Control Register		Go
8h	4h	CAN_ES	Error and Status Register		Go
10h	8h	CAN_ERRC	Error Counter Register		Go
18h	Ch	CAN_BTR	Bit Timing Register		Go
20h	10h	CAN_INT	Interrupt Register		Go
28h	14h	CAN_TEST	Test Register		Go
38h	1Ch	CAN_PERR	CAN Parity Error Code Register		Go
80h	40h	CAN_RAM_INIT	CAN RAM Initialization Register		Go
A0h	50h	CAN_GLB_INT_EN	CAN Global Interrupt Enable Register		Go
A8h	54h	CAN_GLB_INT_FLG	CAN Global Interrupt Flag Register		Go
B0h	58h	CAN_GLB_INT_CLR	CAN Global Interrupt Clear Register		Go
100h	80h	CAN_ABOTR	Auto-Bus-On Time Register		Go
108h	84h	CAN_TXRQ_X	CAN Transmission Request Register		Go
110h	88h	CAN_TXRQ_21	CAN Transmission Request 2_1 Register		Go
130h	98h	CAN_NDAT_X	CAN New Data Register		Go
138h	9Ch	CAN_NDAT_21	CAN New Data 2_1 Register		Go
158h	ACh	CAN_IPEN_X	CAN Interrupt Pending Register		Go
160h	B0h	CAN_IPEN_21	CAN Interrupt Pending 2_1 Register		Go
180h	C0h	CAN_MVAL_X	CAN Message Valid Register		Go
188h	C4h	CAN_MVAL_21	CAN Message Valid 2_1 Register		Go
1B0h	D8h	CAN_IP_MUX21	CAN Interrupt Multiplexer 2_1 Register		Go
200h	100h	CAN_IF1CMD	IF1 Command Register		Go
208h	104h	CAN_IF1MSK	IF1 Mask Register		Go
210h	108h	CAN_IF1ARB	IF1 Arbitration Register		Go
218h	10Ch	CAN_IF1MCTL	IF1 Message Control Register		Go
220h	110h	CAN_IF1DATA	IF1 Data A Register		Go
228h	114h	CAN_IF1DATB	IF1 Data B Register		Go
240h	120h	CAN_IF2CMD	IF2 Command Register		Go
248h	124h	CAN_IF2MSK	IF2 Mask Register		Go
250h	128h	CAN_IF2ARB	IF2 Arbitration Register		Go
258h	12Ch	CAN_IF2MCTL	IF2 Message Control Register		Go
260h	130h	CAN_IF2DATA	IF2 Data A Register		Go
268h	134h	CAN_IF2DATB	IF2 Data B Register		Go
280h	140h	CAN_IF3OBS	IF3 Observation Register		Go
288h	144h	CAN_IF3MSK	IF3 Mask Register		Go
290h	148h	CAN_IF3ARB	IF3 Arbitration Register		Go
298h	14Ch	CAN_IF3MCTL	IF3 Message Control Register		Go
2A0h	150h	CAN_IF3DATA	IF3 Data A Register		Go
2A8h	154h	CAN_IF3DATB	IF3 Data B Register		Go
2C0h	160h	CAN_IF3UPD	IF3 Update Enable Register		Go

Complex bit access types are encoded to fit into small table cells. Table 30-10 shows the codes that are used for access types in this section.

Table 30-10. CAN_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

30.16.3.1 CAN_CTL Register (Offset (x8) = 0h, Offset (x16) = 0h) [Reset = 1401h]

CAN_CTL is shown in Figure 30-22 and described in Table 30-11.

Return to the Summary Table.

This register is used for configuring the CAN module in terms of interrupts, parity, debug-mode behavior etc.

Figure 30-22. CAN_CTL Register

31	30	29	28	27	26	25	24
RESERVED						RESERVED	RESERVED
R-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED			DE3	DE2	DE1	IE1	INITDBG
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h
15	14	13	12	11	10	9	8
SWR	RESERVED	PMD				ABO	IDS
R-0/W1C-0h	R-0h	R/W-5h				R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
Test	CCE	DAR	RESERVED	EIE	SIE	IE0	Init
R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h

Table 30-11. CAN_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23-21	RESERVED	R	0h	Reserved
20	DE3	R/W	0h	Enable DMA request line for IF3 0 Disabled 1 Enabled Note: A pending DMA request for IF3 remains active until first access to one of the IF3 registers. Reset type: SYSRSn
19	DE2	R/W	0h	Enable DMA request line for IF2 0 Disabled 1 Enabled Note: A pending DMA request for IF1 remains active until first access to one of the IF2 registers. Reset type: SYSRSn
18	DE1	R/W	0h	Enable DMA request line for IF1 0 Disabled 1 Enabled Note: A pending DMA request for IF1 remains active until first access to one of the IF1 registers. Reset type: SYSRSn
17	IE1	R/W	0h	Interrupt line 1 Enable 0 CANINT1 is disabled. 1 CANINT1 is enabled. Interrupts will assert CANINT1 line to 1 line remains active until pending interrupts are processed. Reset type: SYSRSn
16	INITDBG	R	0h	Debug Mode Status Bit: This bit indicates the internal init state for a debug access 0 Not in debug mode, or debug mode requested but not entered. 1 Debug mode requested and internally entered the CAN module is ready for debug accesses. Reset type: SYSRSn

Table 30-11. CAN_CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15	SWR	R-0/W1C	0h	Software Reset Enable Bit: This bit activates the software reset. 0 Normal Operation. 1 Module is forced to reset state. This bit will get cleared automatically one clock cycle after execution of software reset. Note: To execute software reset, the following procedure is necessary: 1. Set INIT bit to shut down CAN communication. 2. Set SWR bit. Note: This bit is write-protected by Init bit. If module is reset using the SWR bit, no user configuration is lost. Only status bits get reset along with logic which needs to be reset for the next CAN transaction. If module is reset using SOFTPRES register, entire module will get reset, including configuration registers. Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-10	PMD	R/W	5h	Parity on/off 0101 Parity function disabled Any other value - Parity function enabled Reset type: SYSRSn
9	ABO	R/W	0h	Auto-Bus-On Enable 0 The Auto-Bus-On feature is disabled 1 The Auto-Bus-On feature is enabled Reset type: SYSRSn
8	IDS	R/W	0h	Interruption Debug Support Enable 0 When Debug mode is requested, the CAN module will wait for a started transmission or reception to be completed before entering Debug mode 1 When Debug mode is requested, the CAN module will interrupt any transmission or reception, and enter Debug mode immediately. Reset type: SYSRSn
7	Test	R/W	0h	Test Mode Enable 0 Disable Test Mode (Normal operation) 1 Enable Test Mode Reset type: SYSRSn
6	CCE	R/W	0h	Configuration Change Enable 0 The CPU has no write access to the configuration registers. 1 The CPU has write access to the configuration registers (when Init bit is set). Reset type: SYSRSn
5	DAR	R/W	0h	Disable Automatic Retransmission 0 Automatic Retransmission of "not successful" messages enabled. 1 Automatic Retransmission disabled. Reset type: SYSRSn
4	RESERVED	R	0h	Reserved
3	EIE	R/W	0h	Error Interrupt Enable 0 Disabled - PER, BOff and EWarn bits cannot generate an interrupt. 1 Enabled - PER, BOff and EWarn bits can generate an interrupt at CANINT0 line and affect the Interrupt Register. Reset type: SYSRSn
2	SIE	R/W	0h	Status Change Interrupt Enable 0 Disabled - RxOk, TxOk and LEC bits cannot generate an interrupt. 1 Enabled - RxOk, TxOk and LEC can generate an interrupt on the CANINT0 line Reset type: SYSRSn
1	IE0	R/W	0h	Interrupt line 0 Enable 0 CANINT0 is disabled. 1 CANINT0 is enabled. Interrupts will assert CANINT0 line to 1 line remains active until pending interrupts are processed. Reset type: SYSRSn

Table 30-11. CAN_CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	Init	R/W	1h	Initialization Mode This bit is used to keep the CAN module inactive during bit timing configuration and message RAM initialization. It is set automatically during a bus off event. Clearing this bit will not shorten the bus recovery time. 0 CAN module processes messages normally 1 CAN module ignores bus activity Reset type: SYSRSn

30.16.3.2 CAN_ES Register (Offset (x8) = 8h, Offset (x16) = 4h) [Reset = 7h]

CAN_ES is shown in Figure 30-23 and described in Table 30-12.

Return to the Summary Table.

This register indicates error conditions, if any, of the CAN module. Interrupts are generated by PER, BOff and EWarn bits (if EIE bit in CAN Control Register is set) and by RxOk, TxOk, and LEC bits (if SIE bit in CAN Control Register is set). A change of bit EPass will not generate an Interrupt.
Reading the Error and Status Register clears the PER, RxOk and TxOk bits and sets the LEC to value '7'. Additionally, the Status Interrupt value (0x8000) in the Interrupt Register will be replaced by the next lower priority interrupt value.
For debug support, the auto clear functionality of Error and Status Register (clear of status flags by read) is disabled when in Debug/Suspend mode.

Figure 30-23. CAN_ES Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					RESERVED	RESERVED	PER
R-0h					R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
BOff	EWarn	EPass	RxOk	TxOk	LEC		
R-0h	R-0h	R-0h	R-0h	R-0h	R-7h		

Table 30-12. CAN_ES Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Reserved
8	PER	R	0h	Parity Error Detected: This bit will be reset after the CPU reads the register. 0 No parity error has been detected since last read access. 1 The parity check mechanism has detected a parity error in the Message RAM. Reset type: SYSRSn
7	BOff	R	0h	Bus-off Status Bit: 0 The CAN module is not in Bus-Off state. 1 The CAN module is in Bus-Off state. Reset type: SYSRSn
6	EWarn	R	0h	Warning State Bit: 0 Both error counters are below the error warning limit of 96. 1 At least one of the error counters has reached the error warning limit of 96. Reset type: SYSRSn
5	EPass	R	0h	Error Passive State 0 On CAN Bus error, the CAN could send active error frames. 1 The CAN Core is in the error passive state as defined in the CAN Specification. Reset type: SYSRSn

Table 30-12. CAN_ES Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	RxOk	R	0h	Reception status Bit: This bit indicates the status of reception. The bit will be reset after the CPU reads the register. 0 No message has been successfully received since the last time when this bit was read by the CPU. This bit is never reset by CAN internal events. 1 A message has been successfully received since the last time when this bit was reset by a read access of the CPU. This bit will be set independent of the result of acceptance filtering. Reset type: SYSRSn
3	TxOk	R	0h	Transmission status Bit: This bit indicates the status of transmission. The bit will be reset after the CPU reads the register. 0 No message has been successfully transmitted since the last time when this bit was read by the CPU. This bit is never reset by CAN internal events. 1 A message has been successfully transmitted (error free and acknowledged by at least one other node) since the last time when this bit was cleared by a read access of the CPU. Reset type: SYSRSn
2-0	LEC	R	7h	Last Error Code The LEC field indicates the type of the last error on the CAN bus. This field will be cleared to '0' when a message has been transferred (reception or transmission) without error. This field will be reset to '7' whenever the CPU reads the register. 0 No Error 1 Stuff Error: More than five equal bits in a row have been detected in a part of a received message where this is not allowed. 2 Form Error: A fixed format part of a received frame has the wrong format. 3 Ack Error: The message this CAN Core transmitted was not acknowledged by another node. 4 Bit1 Error: During the transmission of a message (with the exception of the arbitration field), the device wanted to send a recessive level (bit of logical value '1'), but the monitored bus value was dominant. 5 Bit0 Error: During the transmission of a message (or acknowledge bit, or active error flag, or overload flag), the device wanted to send a dominant level (logical value '0'), but the monitored bus level was recessive. During Bus-Off recovery, this status is set each time a sequence of 11 recessive bits has been monitored. This enables the CPU to monitor the proceeding of the Bus-Off recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed). 6 CRC Error: In a received message, the CRC check sum was incorrect. (CRC received for an incoming message does not match the calculated CRC for the received data). 7 No CAN bus event was detected since the last time when CPU has read the Error and Status Register. Any read access to the Error and Status Register re-initializes the LEC to value '7'. Reset type: SYSRSn

30.16.3.3 CAN_ERRC Register (Offset (x8) = 10h, Offset (x16) = 8h) [Reset = 0h]

CAN_ERRC is shown in Figure 30-24 and described in Table 30-13.

Return to the Summary Table.

This register reflects the value of the Transmit and Receive error counters

Figure 30-24. CAN_ERRC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RP	REC							TEC							
R-0h	R-0h							R-0h							

Table 30-13. CAN_ERRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	RP	R	0h	Receive Error Passive 0 The Receive Error Counter is below the error passive level. 1 The Receive Error Counter has reached the error passive level as defined in the CAN Specification. Reset type: SYSRSn
14-8	REC	R	0h	Receive Error Counter Actual state of the Receive Error Counter (values from 0 to 127). Reset type: SYSRSn
7-0	TEC	R	0h	Transmit Error Counter Actual state of the Transmit Error Counter. (values from 0 to 255). Reset type: SYSRSn

30.16.3.4 CAN_BTR Register (Offset (x8) = 18h, Offset (x16) = Ch) [Reset = 2301h]

CAN_BTR is shown in Figure 30-25 and described in Table 30-14.

Return to the Summary Table.

This register is used to configure the bit-timing parameters for the CAN module. This register is only writable if CCE and Init bits in the CAN Control Register are set.
The CAN bit time may be programmed in the range of 8 to 25 time quanta.
The CAN time quantum may be programmed in the range of 1 to1024 CAN_CLK periods.

Figure 30-25. CAN_BTR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				BRPE			
R-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED	TSEG2			TSEG1			
R-0h	R/W-2h			R/W-3h			
7	6	5	4	3	2	1	0
SJW		BRP					
R/W-0h		R/W-1h					

Table 30-14. CAN_BTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-16	BRPE	R/W	0h	Baud Rate Prescaler Extension Valid programmed values are 0 to 15. By programming BRPE the Baud Rate Prescaler can be extended to values up to 1024. Note: This bit is Write Protected by CCE bit. Reset type: SYSRSn
15	RESERVED	R	0h	Reserved
14-12	TSEG2	R/W	2h	Time segment after the sample point Valid programmed values are 0 to 7. The actual TSeg2 value which is interpreted for the Bit Timing will be the programmed TSeg2 value + 1. Note: This bit is Write Protected by CCE bit. Reset type: SYSRSn
11-8	TSEG1	R/W	3h	Time segment before the sample point Valid programmed values are 1 to 15. The actual TSeg1 value interpreted for the Bit Timing will be the programmed TSeg1 value + 1. Note: This bit is Write Protected by CCE bit. Reset type: SYSRSn
7-6	SJW	R/W	0h	Synchronization Jump Width Valid programmed values are 0 to 3. The actual SJW value interpreted for the Synchronization will be the programmed SJW value + 1. Note: This bit is Write Protected by CCE bit. Reset type: SYSRSn

Table 30-14. CAN_BTR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	BRP	R/W	1h	Baud Rate Prescaler- Value by which the CAN_CLK frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid programmed values are 0 to 63. The actual BRP value interpreted for the Bit Timing will be the programmed BRP value + 1. Note: This bit is Write Protected by CCE bit. Reset type: SYSRSn

30.16.3.5 CAN_INT Register (Offset (x8) = 20h, Offset (x16) = 10h) [Reset = 0h]

CAN_INT is shown in Figure 30-26 and described in Table 30-15.

Return to the Summary Table.

This register is used to identify the source of the interrupt(s).

Figure 30-26. CAN_INT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								INT1ID								INT0ID															
R-0h								R-0h								R-0h															

Table 30-15. CAN_INT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	INT1ID	R	0h	Interrupt 1 Cause 0x00 No interrupt is pending. 0x01-0x20 Number of message object (mailbox) which caused the interrupt. 0x21-0xFF Unused. If several interrupts are pending, the CAN Interrupt Register will point to the pending interrupt with the highest priority. Note: The CANINT1 interrupt line remains active until INT1ID reaches value 0 (the cause of the interrupt is reset) or until IE0 is cleared. A message interrupt is cleared by clearing the mailbox's IntPnd bit. Among the message interrupts, the mailbox's interrupt priority decreases with increasing message number. Reset type: SYSRSn
15-0	INT0ID	R	0h	Interrupt 0 Cause 0x0000 - No interrupt is pending. 0x0001 - 0x0020 - Number of message object which caused the interrupt. 0x0021 - 0x7FFF - Unused. 0x8000 - Error and Status Register value is not 0x07. 0x8001 - 0xFFFF - Unused. If several interrupts are pending, the CAN Interrupt Register will point to the pending interrupt with the highest priority. Note: The CANINT0 interrupt line remains active until INT0ID reaches value 0 (the cause of the interrupt is reset) or until IE0 is cleared. The Status Interrupt has the highest priority. Among the message interrupts, the message object's interrupt priority decreases with increasing message number. Reset type: SYSRSn

30.16.3.6 CAN_TEST Register (Offset (x8) = 28h, Offset (x16) = 14h) [Reset = 0h]

CAN_TEST is shown in Figure 30-27 and described in Table 30-16.

Return to the Summary Table.

This register is used to configure the various test options supported. For all test modes, the Test bit in CAN Control Register needs to be set to one. If Test bit is set, the RDA, EXL, Tx1, Tx0, LBack and Silent bits are writable. Bit Rx monitors the state of CANRX pin and therefore is only readable. All Test Register functions are disabled when Test bit is cleared.
Note: Setting Tx[1:0] other than '00' will disturb message transfer.
Note: When the internal loop back mode is active (bit LBack is set), bit EXL will be ignored.

Figure 30-27. CAN_TEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						RDA	EXL
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RX	TX		LBACK	SILENT	RESERVED		
R-0h	R/W-0h		R/W-0h	R/W-0h	R-0h		

Table 30-16. CAN_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	RDA	R/W	0h	RAM Direct Access Enable: 0 Normal Operation. 1 Direct access to the RAM is enabled while in Test Mode. Reset type: SYSRSn
8	EXL	R/W	0h	External Loop Back Mode: 0 Disabled. 1 Enabled. Reset type: SYSRSn
7	RX	R	0h	Monitors the actual value of the CANRX pin: 0 The CAN bus is dominant. 1 The CAN bus is recessive. Reset type: SYSRSn
6-5	TX	R/W	0h	Control of CANTX pin: 00 Normal operation, CANTX is controlled by the CAN Core. 01 Sample Point can be monitored at CANTX pin. 10 CANTX pin drives a dominant value. 11 CANTX pin drives a recessive value. Reset type: SYSRSn
4	LBACK	R/W	0h	Loop Back Mode: 0 Disabled. 1 Enabled. Reset type: SYSRSn
3	SILENT	R/W	0h	Silent Mode: 0 Disabled. 1 Enabled. Reset type: SYSRSn

Table 30-16. CAN_TEST Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	RESERVED	R	0h	Reserved

30.16.3.7 CAN_PERR Register (Offset (x8) = 38h, Offset (x16) = 1Ch) [Reset = 100h]

CAN_PERR is shown in Figure 30-28 and described in Table 30-17.

Return to the Summary Table.

This register indicates the Word/Mailbox number where a parity error has been detected. If a parity error is detected, the PER flag will be set in the Error and Status Register. This bit is not reset by the parity check mechanism
it must be reset by reading the Error and Status Register. In addition to the PER flag, the Parity Error Code Register will indicate the memory area where the parity error has been detected. If more than one word with a parity error was detected, the highest word number with a parity error will be displayed. After a parity error has been detected, the register will hold the last error code until power is removed.

Figure 30-28. CAN_PERR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					WORD_NUM			MSG_NUM							
R-0h					R-1h			R-0h							

Table 30-17. CAN_PERR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Reserved
10-8	WORD_NUM	R	1h	0x01-0x05 Word number where parity error has been detected. RDA word number (1 to 5) of the mailbox (according to the Message RAM representation in RDA mode). Reset type: SYSRSn
7-0	MSG_NUM	R	0h	0x01-0x21 Mailbox number where parity error has been detected Reset type: SYSRSn

30.16.3.8 CAN_RAM_INIT Register (Offset (x8) = 80h, Offset (x16) = 40h) [Reset = 5h]

CAN_RAM_INIT is shown in Figure 30-29 and described in Table 30-18.

Return to the Summary Table.

This register is used to initialize the Mailbox RAM. It clears the entire mailbox RAM, including the MsgVal bits.

Figure 30-29. CAN_RAM_INIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RAM_INIT_DONE	CAN_RAM_INIT	KEY3	KEY2	KEY1	KEY0
R-0h		R-0h	R/W-0h	R/W-0h	R/W-1h	R/W-0h	R/W-1h

Table 30-18. CAN_RAM_INIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RAM_INIT_DONE	R	0h	CAN Mailbox RAM initialization status: 0 Read: Initialization is on-going or initialization not initiated. 1 Read: Initialization complete Reset type: SYSRSn
4	CAN_RAM_INIT	R/W	0h	Initiate CAN Mailbox RAM initialization: 0 Read: Initialization complete or initialization not initiated. Write: No action 1 Read: Initialization is on-going Write: Initiate CAN Mailbox RAM initialization. After initialization, this bit will be automatically cleared to 0. Reset type: SYSRSn
3	KEY3	R/W	0h	See Key 0 Reset type: SYSRSn
2	KEY2	R/W	1h	See Key 0 Reset type: SYSRSn
1	KEY1	R/W	0h	See Key 0 Reset type: SYSRSn
0	KEY0	R/W	1h	KEY3-KEY0 should be 1010 for any write to this register to be valid. These bits will be restored to their reset state after the CAN RAM initialization is complete. Reset type: SYSRSn

30.16.3.9 CAN_GLB_INT_EN Register (Offset (x8) = A0h, Offset (x16) = 50h) [Reset = 0h]

CAN_GLB_INT_EN is shown in Figure 30-30 and described in Table 30-19.

Return to the Summary Table.

This register is used to enable the interrupt lines to the PIE.

Figure 30-30. CAN_GLB_INT_EN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						GLBINT1_EN	GLBINT0_EN
R-0h						R/W-0h	R/W-0h

Table 30-19. CAN_GLB_INT_EN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	GLBINT1_EN	R/W	0h	Global Interrupt Enable for CANINT1 0 CANINT1 does not generate interrupt to PIE 1 CANINT1 generates interrupt to PIE if interrupt condition occurs Reset type: SYSRSn
0	GLBINT0_EN	R/W	0h	Global Interrupt Enable for CANINT0 0 CANINT0 does not generate interrupt to PIE 1 CANINT0 generates interrupt to PIE if interrupt condition occurs Reset type: SYSRSn

30.16.3.10 CAN_GLB_INT_FLG Register (Offset (x8) = A8h, Offset (x16) = 54h) [Reset = 0h]

CAN_GLB_INT_FLG is shown in Figure 30-31 and described in Table 30-20.

Return to the Summary Table.

This register indicates if and when the interrupt line to the PIE is active.

Figure 30-31. CAN_GLB_INT_FLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						INT1_FLG	INT0_FLG
R-0h						R-0h	R-0h

Table 30-20. CAN_GLB_INT_FLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	INT1_FLG	R	0h	CANINT1 Flag 0 No interrupt generated 1 Interrupt is generated due to CANINT1 (refer to CAN Interrupt Status Register for the condition) Reset type: SYSRSn
0	INT0_FLG	R	0h	CANINT0 Flag 0 No interrupt generated 1 Interrupt is generated due to CANINT0 (refer to CAN Interrupt Status Register for the condition) Reset type: SYSRSn

30.16.3.11 CAN_GLB_INT_CLR Register (Offset (x8) = B0h, Offset (x16) = 58h) [Reset = 0h]

CAN_GLB_INT_CLR is shown in Figure 30-32 and described in Table 30-21.

Return to the Summary Table.

This register is used to clear the interrupt to the PIE.

Figure 30-32. CAN_GLB_INT_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						INT1_FLG_CLR	INT0_FLG_CLR
R-0h						W-0h	W-0h

Table 30-21. CAN_GLB_INT_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	INT1_FLG_CLR	W	0h	Global Interrupt flag clear for CANINT1 0 No effect 1 Write 1 to clear the corresponding bit of the Global Interrupt Flag Register and allow the PIE to receive another interrupt from CANINT1. Reset type: SYSRSn
0	INT0_FLG_CLR	W	0h	Global Interrupt flag clear for CANINT0 0 No effect 1 Write 1 to clear the corresponding bit of the Global Interrupt Flag Register and allow the PIE to receive another interrupt from CANINT0. Reset type: SYSRSn

30.16.3.12 CAN_ABOTR Register (Offset (x8) = 100h, Offset (x16) = 80h) [Reset = 0h]

CAN_ABOTR is shown in Figure 30-33 and described in Table 30-22.

Return to the Summary Table.

This register is used to introduce a variable delay before the Bus-off recovery sequence is started.

Figure 30-33. CAN_ABOTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ABO_Time																															
R/W-0h																															

Table 30-22. CAN_ABOTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ABO_Time	R/W	0h	Auto-Bus-On Timer Number of clock cycles before a Bus-Off recovery sequence is started by clearing the Init bit. "Clock" refers to the input clock to the CAN module. This function has to be enabled by setting bit ABO in CAN Control Register. The Auto-Bus-On timer is realized by a 32-bit counter which starts to count down to zero when the module goes Bus-Off. The counter will be reloaded with the preload value of the ABO Time register after this phase. NOTE: On write access to the CAN Control register while Auto-Bus-On timer is running, the Auto-Bus-On procedure will be aborted. NOTE: During Debug mode, running Auto-Bus-On timer will be paused. Reset type: SYSRSn

30.16.3.13 CAN_TXRQ_X Register (Offset (x8) = 108h, Offset (x16) = 84h) [Reset = 0h]

CAN_TXRQ_X is shown in Figure 30-34 and described in Table 30-23.

Return to the Summary Table.

With these bits, the CPU can detect if one or more bits in the CAN Transmission Request 21 Register (CAN_TXRQ_21) is set. Each bit in this register represents a group of eight mailboxes. If at least one of the TxRqst bits of these message objects is set, the corresponding bit in this register will be set.

Figure 30-34. CAN_TXRQ_X Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				TxRqstReg2		TxRqstReg1	
R-0h				R-0h		R-0h	

Table 30-23. CAN_TXRQ_X Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-2	TxRqstReg2	R	0h	Transmit Request Register 2 flag: Bit 2 represents byte 2 of CAN_TXRQ_21. If one or more bits in that byte are set, then bit 2 will be set. Bit 3 represents byte 3 of CAN_TXRQ_21 Register. If one or more bits in that byte are set, then bit 3 will be set. Reset type: SYSRSn
1-0	TxRqstReg1	R	0h	Transmit Request Register 1 flag: Bit 0 represents byte 0 of CAN_TXRQ_21 Register. If one or more bits in that byte are set, then bit 0 will be set. Bit 1 represents byte 1 of CAN_TXRQ_21 Register. If one or more bits in that byte are set, then bit 1 will be set. Reset type: SYSRSn

30.16.3.14 CAN_TXRQ_21 Register (Offset (x8) = 110h, Offset (x16) = 88h) [Reset = 0h]

CAN_TXRQ_21 is shown in Figure 30-35 and described in Table 30-24.

Return to the Summary Table.

This register holds the TxRqst bits of the mailboxes. By reading out these bits, the CPU can check for pending transmission requests. The TxRqst bit in a specific mailbox can be set/reset by the CPU via the IF1/IF2 message interface registers, or by the message handler after reception of a remote frame or after a successful transmission.

Figure 30-35. CAN_TXRQ_21 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TxRqst																															
R-0h																															

Table 30-24. CAN_TXRQ_21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TxRqst	R	0h	Transmission Request Bits (for all message objects) 0 No transmission has been requested for this message object. 1 The transmission of this message object is requested and is not yet done. Note: Bit 0 is for mailbox 1, Bit 1 is for mailbox 2, Bit 2 is for mailbox 3,..., Bit 31 is for mailbox 32 Reset type: SYSRSn

30.16.3.15 CAN_NDAT_X Register (Offset (x8) = 130h, Offset (x16) = 98h) [Reset = 0h]

CAN_NDAT_X is shown in Figure 30-36 and described in Table 30-25.

Return to the Summary Table.

With these bits, the CPU can detect if one or more bits in the CAN New Data 21 Register (CAN_NDAT _21) is set. Each bit in this register represents a group of eight mailboxes. If at least one of the NewDat bits of these mailboxes are set, the corresponding bit in this register will be set.

Figure 30-36. CAN_NDAT_X Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				NewDatReg2		NewDatReg1	
R-0h				R-0h		R-0h	

Table 30-25. CAN_NDAT_X Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-2	NewDatReg2	R	0h	New Data Register 2 flag: Bit 2 represents byte 2 of CAN_NDAT _21 Register. If one or more bits in that byte are set, then bit 2 will be set. Bit 3 represents byte 3 of CAN_NDAT _21 Register. If one or more bits in that byte are set, then bit 3 will be set. Reset type: SYSRSn
1-0	NewDatReg1	R	0h	New Data Register 1 flag: Bit 0 represents byte 0 of CAN_NDAT _21 Register. If one or more bits in that byte are set, then bit 0 will be set. Bit 1 represents byte 1 of CAN_NDAT _21 Register. If one or more bits in that byte are set, then bit 1 will be set. Reset type: SYSRSn

30.16.3.16 CAN_NDAT_21 Register (Offset (x8) = 138h, Offset (x16) = 9Ch) [Reset = 0h]

CAN_NDAT_21 is shown in Figure 30-37 and described in Table 30-26.

Return to the Summary Table.

This register holds the NewDat bits of all mailboxes. By reading out the NewDat bits, the CPU can check for which mailboxes the data portion was updated. The NewDat bit of a specific mailbox can be set/reset by the CPU via the IFx "Message Interface" Registers or by the Message Handler after reception of a Data Frame or after a successful transmission.

Figure 30-37. CAN_NDAT_21 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NewDat																															
R-0h																															

Table 30-26. CAN_NDAT_21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NewDat	R	0h	New Data Bits (for all message objects) 0 No new data has been written into the data portion of this message object by the message handler since the last time when this flag was cleared by the CPU. 1 The message handler or the CPU has written new data into the data portion of this message object. Note: Bit 0 is for mailbox 1, Bit 1 is for mailbox 2, Bit 2 is for mailbox 3,..., Bit 31 is for mailbox 32 Reset type: SYSRSn

30.16.3.17 CAN_IPEN_X Register (Offset (x8) = 158h, Offset (x16) = ACh) [Reset = 0h]

CAN_IPEN_X is shown in Figure 30-38 and described in Table 30-27.

Return to the Summary Table.

With these bits, the CPU can detect if one or more bits in the CAN Interrupt Pending 21 Register (CAN_IPEN_21) is set. Each bit in this register represents a group of eight mailboxes. If at least one of the IntPnd bits of these mailboxes are set, the corresponding bit in this register will be set.

Figure 30-38. CAN_IPEN_X Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				IntPndReg2		IntPndReg1	
R-0h				R-0h		R-0h	

Table 30-27. CAN_IPEN_X Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-2	IntPndReg2	R	0h	Interrupt Pending Register 2 flag: Bit 2 represents byte 2 of CAN_IPEN_21 Register. If one or more bits in that byte are set, then bit 2 will be set. Bit 3 represents byte 3 of CAN_IPEN_21 Register. If one or more bits in that byte are set, then bit 3 will be set. Reset type: SYSRSn
1-0	IntPndReg1	R	0h	Interrupt Pending Register 1 flag: Bit 0 represents byte 0 of CAN_IPEN_21 Register. If one or more bits in that byte are set, then bit 0 will be set. Bit 1 represents byte 1 of CAN_IPEN_21 Register. If one or more bits in that byte are set, then bit 1 will be set. Reset type: SYSRSn

30.16.3.18 CAN_IPEN_21 Register (Offset (x8) = 160h, Offset (x16) = B0h) [Reset = 0h]

CAN_IPEN_21 is shown in Figure 30-39 and described in Table 30-28.

Return to the Summary Table.

This register holds the IntPnd bits of the mailboxes. By reading out these bits, the CPU can check for pending interrupts in the mailboxes. The IntPnd bit of a specific mailbox can be set/reset by the CPU via the IF1/IF2 interface register sets, or by the message handler after a reception or a successful transmission.

Figure 30-39. CAN_IPEN_21 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IntPnd																															
R-0h																															

Table 30-28. CAN_IPEN_21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IntPnd	R	0h	Interrupt Pending bits: This register contains the bits that indicate the pending interrupts in each one of the 32 mailboxes. 0 This mailbox is not the source of an interrupt. 1 This mailbox is the source of an interrupt. Note: Bit 0 is for mailbox 1, Bit 1 is for mailbox 2, Bit 2 is for mailbox 3,..., Bit 31 is for mailbox 32 Reset type: SYSRSn

30.16.3.19 CAN_MVAL_X Register (Offset (x8) = 180h, Offset (x16) = C0h) [Reset = 0h]

CAN_MVAL_X is shown in Figure 30-40 and described in Table 30-29.

Return to the Summary Table.

With these bits, the CPU can detect if one or more bits in the CAN Message Valid 2_1 Register (CAN_MVAL_21) is set.Each bit in this register represents a group of eight mailboxes. If at least one of the MsgVal bits of these mailboxes are set, the corresponding bit in this register will be set.

Figure 30-40. CAN_MVAL_X Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MsgValReg2		MsgValReg1	
R-0h				R-0h		R-0h	

Table 30-29. CAN_MVAL_X Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-2	MsgValReg2	R	0h	Message Valid Register 2 flag: Bit 2 represents byte 2 of CAN_ MVAL _21 Register. If one or more bits in that byte are set, then bit 2 will be set. Bit 3 represents byte 3 of CAN_ MVAL _21 Register. If one or more bits in that byte are set, then bit 3 will be set. Reset type: SYSRSn
1-0	MsgValReg1	R	0h	Message Valid Register 1 flag: Bit 0 represents byte 0 of CAN_ MVAL _21 Register. If one or more bits in that byte are set, then bit 0 will be set. Bit 1 represents byte 1 of CAN_ MVAL _21 Register. If one or more bits in that byte are set, then bit 1 will be set. Reset type: SYSRSn

30.16.3.20 CAN_MVAL_21 Register (Offset (x8) = 188h, Offset (x16) = C4h) [Reset = 0h]

CAN_MVAL_21 is shown in Figure 30-41 and described in Table 30-30.

Return to the Summary Table.

This registers hold the MsgVal bits of all mailboxes. By reading out the MsgVal bits, the CPU can check which mailbox is valid. The MsgVal bit of a specific mailbox can be set/reset by the CPU via the IF1/2 "Message Interface" Registers.

Figure 30-41. CAN_MVAL_21 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MsgValReg																															
R-0h																															

Table 30-30. CAN_MVAL_21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MsgValReg	R	0h	Message Valid Bits (for all message objects) 0 This message object is ignored by the message handler. 1 This message object is configured and will be considered by the message handler. Note: Bit 0 is for mailbox 1, Bit 1 is for mailbox 2, Bit 2 is for mailbox 3,..., Bit 31 is for mailbox 32 Reset type: SYSRSn

30.16.3.21 CAN_IP_MUX21 Register (Offset (x8) = 1B0h, Offset (x16) = D8h) [Reset = 0h]

CAN_IP_MUX21 is shown in Figure 30-42 and described in Table 30-31.

Return to the Summary Table.

The IntMux bit determines for each mailbox, which of the two interrupt lines (CANINT0 or CANINT1) will be asserted when the IntPnd bit of that mailbox is set. Both interrupt lines can be globally enabled or disabled by setting or clearing IE0 and IE1 bits in CAN Control Register. This will also affect the INT0ID or INT1ID flags in the Interrupt Register.

Figure 30-42. CAN_IP_MUX21 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IntMux																															
R/W-0h																															

Table 30-31. CAN_IP_MUX21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IntMux	R/W	0h	Interrupt Mux bits: 0 CANINT0 line is active if corresponding IntPnd flag is one. 1 CANINT1 line is active if corresponding IntPnd flag is one. Note: Bit 0 is for mailbox 32, Bit 1 is for mailbox 1, Bit 2 is for mailbox 2,..., Bit 31 is for mailbox 31 Reset type: SYSRSn

30.16.3.22 CAN_IF1CMD Register (Offset (x8) = 200h, Offset (x16) = 100h) [Reset = 1h]

CAN_IF1CMD is shown in Figure 30-43 and described in Table 30-32.

Return to the Summary Table.

The IF1/IF2 Command Registers configure and initiate the transfer between the IF1/IF2 Register sets and the Message RAM. It is configurable which portions of the message object should be transferred. A transfer is started when the CPU writes the message number to bits [7:0] of the IF1/IF2 Command Register. With this write operation, the Busy bit is automatically set to '1' to indicate that a transfer is in progress. After 4 to 14 clock cycles, the transfer between the Interface Register and the Message RAM will be completed and the Busy bit is cleared. The maximum number of cycles is needed when the message transfer coincides with a CAN message transmission, acceptance filtering, or message storage.
If the CPU writes to both IF1/IF2 Command Registers consecutively (request of a second transfer while first transfer is still in progress), the second transfer will start after the first one has been completed. The following points must be borne in mind while writing to this register: (1) Do not write zeros to the whole register. (2) Write to the register in a single 32-bit write or write the upper 16-bits before writing to the lower 16- bits.
Note: While Busy bit is one, IF1/IF2 Register sets are write protected.
Note: For debug support, the auto clear functionality of the IF1/IF2 Command Registers (clear of DMAactive flag by R/W, for devices with DMA support) is disabled during Debug/Suspend mode.
Note: If an invalid Message Number is written to bits [7:0] of the IF1/IF2 Command Register, the Message Handler may access an implemented (valid) message object instead.

Figure 30-43. CAN_IF1CMD Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
DIR	Mask	Arb	Control	ClrIntPnd	TXRQST	DATA_A	DATA_B
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
Busy	DMAactive	RESERVED					
R-0h	R/W-0h	R-0h					
7	6	5	4	3	2	1	0
MSG_NUM							
R/W-1h							

Table 30-32. CAN_IF1CMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23	DIR	R/W	0h	Write/Read 0 Direction = Read: Transfer direction is from the message object addressed by Message Number (Bits [7:0]) to the IF1/IF2 Register set. That is, transfer data from the mailbox into the selected IF1/IF2 Message Buffer Registers. 1 Direction = Write: Transfer direction is from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]) . That is, transfer data from the selected IF1/IF2 Message Buffer Registers to the mailbox. The other bits of IF1/IF2 Command Mask Register have different functions depending on the transfer direction. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

Table 30-32. CAN_IF1CMD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22	Mask	R/W	0h	Access Mask Bits 0 Mask bits will not be changed 1 (Direction = Read): The Mask bits (Identifier Mask + MDir + MXtd) will be transferred from the message object addressed by Message Number (Bits [7:0]) to the IF1/IF2 Register set. 1 (Direction = Write): The Mask bits (Identifier Mask + MDir + MXtd) will be transferred from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]). Note: This bit is write protected by Busy bit. Reset type: SYSRSn
21	Arb	R/W	0h	Access Arbitration Bits 0 Arbitration bits will not be changed 1 (Direction = Read): The Arbitration bits (Identifier + Dir + Xtd + MsgVal) will be transferred from the message object addressed by Message Number (Bits [7:0]) to the corresponding IF1/IF2 Register set. 1 (Direction = Write): The Arbitration bits (Identifier + Dir + Xtd + MsgVal) will be transferred from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]). Note: This bit is write protected by Busy bit. Reset type: SYSRSn
20	Control	R/W	0h	Access control bits. If the TxRqst/NewDat bit in this register(Bit [18]) is set, the TxRqst/NewDat bit in the IF1 message control register will be ignored. 0 Control bits will not be changed. 1 (Direction = Read): The message control bits will be transferred from the message object addressed by message number (Bits [7:0]) to the IF1 register set. 1 (Direction = Write): The message control bits will be transferred from the IF1 register set to the message object addressed by message number (Bits [7:0]). Note: This bit is write protected by the Busy bit. Reset type: SYSRSn
19	ClrIntPnd	R/W	0h	Clear Interrupt Pending Bit 0 IntPnd bit will not be changed 1 (Direction = Read): Clears IntPnd bit in the message object. 1 (Direction = Write): This bit is ignored. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
18	TXRQST	R/W	0h	Access Transmission Request (TxRqst) / New Data (NewDat) Bit 0 (Direction = Read): NewDat bit will not be changed. 0 (Direction = Write): TxRqst/NewDat bit will be handled according to the Control bit. 1 (Direction = Read): Clears NewDat bit in the message object. 1 (Direction = Write): Sets TxRqst/NewDat in message object. Note: If a CAN transmission is requested by setting TxRqst/NewDat in this register, the TxRqst/NewDat bits in the message object will be set to one independent of the values in IF1/IF2 Message Control Register. Note: A read access to a message object can be combined with the reset of the control bits IntPnd and NewDat. The values of these bits transferred to the IF1/IF2 Message Control Register always reflect the status before resetting them. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

Table 30-32. CAN_IF1CMD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	DATA_A	R/W	0h	Access Data Bytes 0-3 0 Data Bytes 0-3 will not be changed. 1 (Direction = Read): The Data Bytes 0-3 will be transferred from the message object addressed by the Message Number (Bits [7:0]) to the corresponding IF1/IF2 Register set. 1 (Direction = Write): The Data Bytes 0-3 will be transferred from the IF1/IF2 Register set to the message object addressed by the Message Number (Bits [7:0]). Note: The duration of the message transfer is independent of the number of bytes to be transferred. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
16	DATA_B	R/W	0h	Access Data Bytes 4-7 0 Data Bytes 4-7 will not be changed. 1 (Direction = Read): The Data Bytes 4-7 will be transferred from the message object addressed by Message Number (Bits [7:0]) to the corresponding IF1/IF2 Register set. 1 (Direction = Write): The Data Bytes 4-7 will be transferred from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]). Note: The duration of the message transfer is independent of the number of bytes to be transferred. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
15	Busy	R	0h	Busy Flag 0 No transfer between IF1/IF2 Register Set and Message RAM is in progress. 1 Transfer between IF1/IF2 Register Set and Message RAM is in progress. This bit is set to one after the message number has been written to bits [7:0]. IF1/IF2 Register Set will be write protected. The bit is cleared after read/write action has been finished. Reset type: SYSRSn
14	DMAactive	R/W	0h	DMA trigger status due to IF1 update. 0 No IF1 DMA request is active. 1 DMA is requested after a completed transfer between IF1 and the message RAM. The DMA request remains active until the first read or write to one of the IF1 registers an exception is a write to Message Number (Bits [7:0]) when DMAactive is one. Note: Due to the auto reset feature of the DMAactive bit, this bit has to be set for each subsequent DMA cycle separately. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
13-8	RESERVED	R	0h	Reserved
7-0	MSG_NUM	R/W	1h	Number of message object in Message RAM which is used for data transfer 0x00 Invalid message number 0x01-0x20 Valid message numbers 0x21-0xFF Invalid message numbers Note: This bit is write protected by Busy bit. Reset type: SYSRSn

30.16.3.23 CAN_IF1MSK Register (Offset (x8) = 208h, Offset (x16) = 104h) [Reset = FFFFFFFFh]

CAN_IF1MSK is shown in Figure 30-44 and described in Table 30-33.

Return to the Summary Table.

The bits of the IF1/IF2 Mask Registers mirror the mask bits of a message object.
Note: While Busy bit of IF1/IF2 Command Register is one, IF1/IF2 Register Set is write-protected.

Figure 30-44. CAN_IF1MSK Register

31	30	29	28	27	26	25	24
MXtd	MDir	RESERVED	Msk				
R/W-1h	R/W-1h	R-1h	R/W-1FFFFFFFh				
23	22	21	20	19	18	17	16
Msk							
R/W-1FFFFFFFh							
15	14	13	12	11	10	9	8
Msk							
R/W-1FFFFFFFh							
7	6	5	4	3	2	1	0
Msk							
R/W-1FFFFFFFh							

Table 30-33. CAN_IF1MSK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MXtd	R/W	1h	Mask Extended Identifier 0 The extended identifier bit (Xtd) has no effect on the acceptance filtering. 1 The extended identifier bit (Xtd) is used for acceptance filtering. When 11-bit ("standard") identifiers are used for a message object, the identifiers of received data frames are written into bits ID[28:18]. For acceptance filtering, only these bits together with mask bits Msk[28:18] are considered. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
30	MDir	R/W	1h	Mask Message Direction 0 The message direction bit (Dir) has no effect on the acceptance filtering. 1 The message direction bit (Dir) is used for acceptance filtering. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
29	RESERVED	R	1h	Reserved
28-0	Msk	R/W	1FFFFFFFh	Identifier Mask- 0 The corresponding bit in the identifier of the message object is not used for acceptance filtering (don't care). 1 The corresponding bit in the identifier of the message object is used for acceptance filtering. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

30.16.3.24 CAN_IF1ARB Register (Offset (x8) = 210h, Offset (x16) = 108h) [Reset = 0h]

CAN_IF1ARB is shown in Figure 30-45 and described in Table 30-34.

Return to the Summary Table.

The bits of the IF1/IF2 Arbitration Registers mirror the arbitration bits of a message object. The Arbitration bits ID[28:0], Xtd, and Dir are used to define the identifier and type of outgoing messages and (together with the Mask bits Msk[28:0], MXtd, and MDir) for acceptance filtering of incoming messages.
A received message is stored into the valid message object with matching identifier and Direction = receive (Data Frame) or Direction = transmit (Remote Frame).
Extended frames can be stored only in message objects with Xtd = one, standard frames in message objects with Xtd = zero.
If a received message (Data Frame or Remote Frame) matches more than one valid message objects, it is stored into the one with the lowest message number.
Note: While Busy bit of IF1/IF2 Command Register is one, IF1/IF2 Register Set is write-protected.

Figure 30-45. CAN_IF1ARB Register

31	30	29	28	27	26	25	24
MsgVal	Xtd	Dir	ID				
R/W-0h	R/W-0h	R/W-0h	R/W-0h				
23	22	21	20	19	18	17	16
ID							
R/W-0h							
15	14	13	12	11	10	9	8
ID							
R/W-0h							
7	6	5	4	3	2	1	0
ID							
R/W-0h							

Table 30-34. CAN_IF1ARB Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MsgVal	R/W	0h	Message Valid 0 The mailbox is disabled. (The message object is ignored by the message handler). 1 The mailbox is enabled. (The message object is to be used by the message handler). The CPU should reset the MsgVal bit of all unused Messages Objects during the initialization before it resets the Init bit in the CAN Control Register. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
30	Xtd	R/W	0h	Extended Identifier 0 The 11-bit ("standard") Identifier is used for this message object. 1 The 29-bit ("extended") Identifier is used for this message object. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

Table 30-34. CAN_IF1ARB Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
29	Dir	R/W	0h	Message Direction 0 Direction = receive: On TxRqst, a remote frame with the identifier of this message object is transmitted. On reception of a data frame with matching identifier, that frame is stored in this message object. 1 Direction = transmit: On TxRqst, the respective message object is transmitted as a data frame. On reception of a remote frame with matching identifier, the TxRqst bit of this message object is set (if RmtEn = one). Note: This bit is write protected by Busy bit. Reset type: SYSRSn
28-0	ID	R/W	0h	Message Identifier ID[28:0] 29-bit Identifier ("Extended Frame") ID[28:18] 11-bit Identifier ("Standard Frame") Note: This bit is write protected by Busy bit. Reset type: SYSRSn

30.16.3.25 CAN_IF1MCTL Register (Offset (x8) = 218h, Offset (x16) = 10Ch) [Reset = 0h]

CAN_IF1MCTL is shown in Figure 30-46 and described in Table 30-35.

Return to the Summary Table.

The bits of the IF1/IF2 Message Control Registers mirror the message control bits of a message object. This register has control/status bits pertaining to interrupts, acceptance mask, remote frames and FIFO option.

Figure 30-46. CAN_IF1MCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
EoB	RESERVED			DLC			
R/W-0h	R-0h			R/W-0h			

Table 30-35. CAN_IF1MCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	NewDat	R/W	0h	New Data 0 No new data has been written into the data portion of this message object by the message handler since the last time when this flag was cleared by the CPU. 1 The message handler or the CPU has written new data into the data portion of this message object. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
14	MsgLst	R/W	0h	Message Lost (only valid for message objects with direction = receive) 0 No message lost since the last time when this bit was reset by the CPU. 1 The message handler stored a new message into this object when NewDat was still set, so the previous message has been overwritten. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
13	IntPnd	R/W	0h	Interrupt Pending 0 This message object is not the source of an interrupt. 1 This message object is the source of an interrupt. The Interrupt Identifier in the Interrupt Register will point to this message object if there is no other interrupt source with higher priority. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
12	UMask	R/W	0h	Use Acceptance Mask 0 Mask ignored 1 Use Mask (Msk[28:0], MXtd, and MDir) for acceptance filtering If the UMask bit is set to one, the message object's mask bits have to be programmed during initialization of the message object before MsgVal is set to one. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

Table 30-35. CAN_IF1MCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	TxIE	R/W	0h	Transmit Interrupt Enable 0 IntPnd will not be triggered after the successful transmission of a frame. 1 IntPnd will be triggered after the successful transmission of a frame. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
10	RxIE	R/W	0h	Receive Interrupt Enable 0 IntPnd will not be triggered after the successful reception of a frame. 1 IntPnd will be triggered after the successful reception of a frame. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
9	RmtEn	R/W	0h	Remote Enable 0 At the reception of a remote frame, TxRqst is not changed. 1 At the reception of a remote frame, TxRqst is set. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
8	TxRqst	R/W	0h	Transmit Request 0 This message object is not waiting for a transmission. 1 The transmission of this message object is requested and is not yet done. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
7	EoB	R/W	0h	End of Block 0 The message object is part of a FIFO Buffer block and is not the last message object of the FIFO Buffer block. 1 The message object is a single message object or the last message object in a FIFO Buffer Block. Note: This bit is used to concatenate multiple message objects to build a FIFO Buffer. For single message objects (not belonging to a FIFO Buffer), this bit must always be set to one. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
6-4	RESERVED	R	0h	Reserved
3-0	DLC	R/W	0h	Data length code 0-8 Data frame has 0-8 data bytes. 9-15 Data frame has 8 data bytes. Note: The data length code of a message object must be defined the same as in all the corresponding objects with the same identifier at other nodes. When the message handler stores a data frame, it will write the DLC to the value given by the received message. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

30.16.3.26 CAN_IF1DATA Register (Offset (x8) = 220h, Offset (x16) = 110h) [Reset = 0h]

CAN_IF1DATA is shown in Figure 30-47 and described in Table 30-36.

Return to the Summary Table.

This register provides a window to the data bytes of the CAN message. The data bytes of CAN messages are stored in the IF1/IF2 registers in the following order. In a CAN Data Frame, Data 0 is the first, and Data 7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first. All bits in this register are write-protected by the Busy bit.

Figure 30-47. CAN_IF1DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data_3								Data_2								Data_1								Data_0							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 30-36. CAN_IF1DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	Data_3	R/W	0h	Data Byte 3 Reset type: SYSRSn
23-16	Data_2	R/W	0h	Data Byte 2 Reset type: SYSRSn
15-8	Data_1	R/W	0h	Data Byte 1 Reset type: SYSRSn
7-0	Data_0	R/W	0h	Data Byte 0 Reset type: SYSRSn

30.16.3.27 CAN_IF1DATB Register (Offset (x8) = 228h, Offset (x16) = 114h) [Reset = 0h]

CAN_IF1DATB is shown in Figure 30-48 and described in Table 30-37.

Return to the Summary Table.

This register provides a window to the data bytes of the CAN message. The data bytes of CAN messages are stored in the IF1/IF2 registers in the following order. In a CAN Data Frame, Data 0 is the first, and Data 7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first. All bits in this register are write-protected by the Busy bit.

Figure 30-48. CAN_IF1DATB Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data_7								Data_6								Data_5								Data_4							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 30-37. CAN_IF1DATB Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	Data_7	R/W	0h	Data Byte 7 Reset type: SYSRSn
23-16	Data_6	R/W	0h	Data Byte 6 Reset type: SYSRSn
15-8	Data_5	R/W	0h	Data Byte 5 Reset type: SYSRSn
7-0	Data_4	R/W	0h	Data Byte 4 Reset type: SYSRSn

30.16.3.28 CAN_IF2CMD Register (Offset (x8) = 240h, Offset (x16) = 120h) [Reset = 1h]

CAN_IF2CMD is shown in Figure 30-49 and described in Table 30-38.

Return to the Summary Table.

The IF1/IF2 Command Registers configure and initiate the transfer between the IF1/IF2 Register sets and the Message RAM. It is configurable which portions of the message object should be transferred. A transfer is started when the CPU writes the message number to bits [7:0] of the IF1/IF2 Command Register. With this write operation, the Busy bit is automatically set to '1' to indicate that a transfer is in progress. After 4 to 14 clock cycles, the transfer between the Interface Register and the Message RAM will be completed and the Busy bit is cleared. The maximum number of cycles is needed when the message transfer coincides with a CAN message transmission, acceptance filtering, or message storage.
If the CPU writes to both IF1/IF2 Command Registers consecutively (request of a second transfer while first transfer is still in progress), the second transfer will start after the first one has been completed. The following points must be borne in mind while writing to this register: (1) Do not write zeros to the whole register. (2) Write to the register in a single 32-bit write or write the upper 16-bits before writing to the lower 16- bits.
Note: While Busy bit is one, IF1/IF2 Register sets are write protected.
Note: For debug support, the auto clear functionality of the IF1/IF2 Command Registers (clear of DMAactive flag by R/W, for devices with DMA support) is disabled during Debug/Suspend mode.
Note: If an invalid Message Number is written to bits [7:0] of the IF1/IF2 Command Register, the Message Handler may access an implemented (valid) message object instead.

Figure 30-49. CAN_IF2CMD Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
DIR	Mask	Arb	Control	ClrIntPnd	TxRqst	DATA_A	DATA_B
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
Busy	DMAactive	RESERVED					
R-0h	R/W-0h	R-0h					
7	6	5	4	3	2	1	0
MSG_NUM							
R/W-1h							

Table 30-38. CAN_IF2CMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23	DIR	R/W	0h	Write/Read 0 Direction = Read: Transfer direction is from the message object addressed by Message Number (Bits [7:0]) to the IF1/IF2 Register set. That is, transfer data from the mailbox into the selected IF1/IF2 Message Buffer Registers. 1 Direction = Write: Transfer direction is from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]) . That is, transfer data from the selected IF1/IF2 Message Buffer Registers to the mailbox. The other bits of IF1/IF2 Command Mask Register have different functions depending on the transfer direction. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

Table 30-38. CAN_IF2CMD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22	Mask	R/W	0h	Access Mask Bits 0 Mask bits will not be changed 1 (Direction = Read): The Mask bits (Identifier Mask + MDir + MXtd) will be transferred from the message object addressed by Message Number (Bits [7:0]) to the IF1/IF2 Register set. 1 (Direction = Write): The Mask bits (Identifier Mask + MDir + MXtd) will be transferred from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]). Note: This bit is write protected by Busy bit. Reset type: SYSRSn
21	Arb	R/W	0h	Access Arbitration Bits 0 Arbitration bits will not be changed 1 (Direction = Read): The Arbitration bits (Identifier + Dir + Xtd + MsgVal) will be transferred from the message object addressed by Message Number (Bits [7:0]) to the corresponding IF1/IF2 Register set. 1 (Direction = Write): The Arbitration bits (Identifier + Dir + Xtd + MsgVal) will be transferred from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]). Note: This bit is write protected by Busy bit. Reset type: SYSRSn
20	Control	R/W	0h	Access control bits. If the TxRqst/NewDat bit in this register(Bit [18]) is set, the TxRqst/NewDat bit in the IF1 message control register will be ignored. 0 Control bits will not be changed. 1 (Direction = Read): The message control bits will be transferred from the message object addressed by message number (Bits [7:0]) to the IF1 register set. 1 (Direction = Write): The message control bits will be transferred from the IF1 register set to the message object addressed by message number (Bits [7:0]). Note: This bit is write protected by the Busy bit. Reset type: SYSRSn
19	ClrIntPnd	R/W	0h	Clear Interrupt Pending Bit 0 IntPnd bit will not be changed 1 (Direction = Read): Clears IntPnd bit in the message object. 1 (Direction = Write): This bit is ignored. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
18	TxRqst	R/W	0h	Access Transmission Request (TxRqst) / New Data (NewDat) Bit 0 (Direction = Read): NewDat bit will not be changed. 0 (Direction = Write): TxRqst/NewDat bit will be handled according to the Control bit. 1 (Direction = Read): Clears NewDat bit in the message object. 1 (Direction = Write): Sets TxRqst/NewDat in message object. Note: If a CAN transmission is requested by setting TxRqst/NewDat in this register, the TxRqst/NewDat bits in the message object will be set to one independent of the values in IF1/IF2 Message Control Register. Note: A read access to a message object can be combined with the reset of the control bits IntPnd and NewDat. The values of these bits transferred to the IF1/IF2 Message Control Register always reflect the status before resetting them. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

Table 30-38. CAN_IF2CMD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	DATA_A	R/W	0h	Access Data Bytes 0-3 0 Data Bytes 0-3 will not be changed. 1 (Direction = Read): The Data Bytes 0-3 will be transferred from the message object addressed by the Message Number (Bits [7:0]) to the corresponding IF1/IF2 Register set. 1 (Direction = Write): The Data Bytes 0-3 will be transferred from the IF1/IF2 Register set to the message object addressed by the Message Number (Bits [7:0]). Note: The duration of the message transfer is independent of the number of bytes to be transferred. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
16	DATA_B	R/W	0h	Access Data Bytes 4-7 0 Data Bytes 4-7 will not be changed. 1 (Direction = Read): The Data Bytes 4-7 will be transferred from the message object addressed by Message Number (Bits [7:0]) to the corresponding IF1/IF2 Register set. 1 (Direction = Write): The Data Bytes 4-7 will be transferred from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]). Note: The duration of the message transfer is independent of the number of bytes to be transferred. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
15	Busy	R	0h	Busy Flag 0 No transfer between IF1/IF2 Register Set and Message RAM is in progress. 1 Transfer between IF1/IF2 Register Set and Message RAM is in progress. This bit is set to one after the message number has been written to bits [7:0]. IF1/IF2 Register Set will be write protected. The bit is cleared after read/write action has been finished. Reset type: SYSRSn
14	DMAactive	R/W	0h	DMA trigger status due to IF1 update. 0 No IF1 DMA request is active. 1 DMA is requested after a completed transfer between IF1 and the message RAM. The DMA request remains active until the first read or write to one of the IF1 registers an exception is a write to Message Number (Bits [7:0]) when DMAactive is one. Note: Due to the auto reset feature of the DMAactive bit, this bit has to be set for each subsequent DMA cycle separately. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
13-8	RESERVED	R	0h	Reserved
7-0	MSG_NUM	R/W	1h	Number of message object in Message RAM which is used for data transfer 0x00 Invalid message number 0x01-0x20 Valid message numbers 0x21-0xFF Invalid message numbers Note: This bit is write protected by Busy bit. Reset type: SYSRSn

30.16.3.29 CAN_IF2MSK Register (Offset (x8) = 248h, Offset (x16) = 124h) [Reset = FFFFFFFFh]

CAN_IF2MSK is shown in Figure 30-50 and described in Table 30-39.

Return to the Summary Table.

The bits of the IF1/IF2 Mask Registers mirror the mask bits of a message object.
Note: While Busy bit of IF1/IF2 Command Register is one, IF1/IF2 Register Set is write-protected.

Figure 30-50. CAN_IF2MSK Register

31	30	29	28	27	26	25	24
MXtd	MDir	RESERVED	Msk				
R/W-1h	R/W-1h	R-1h	R/W-1FFFFFFFh				
23	22	21	20	19	18	17	16
Msk							
R/W-1FFFFFFFh							
15	14	13	12	11	10	9	8
Msk							
R/W-1FFFFFFFh							
7	6	5	4	3	2	1	0
Msk							
R/W-1FFFFFFFh							

Table 30-39. CAN_IF2MSK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MXtd	R/W	1h	Mask Extended Identifier 0 The extended identifier bit (Xtd) has no effect on the acceptance filtering. 1 The extended identifier bit (Xtd) is used for acceptance filtering. When 11-bit ("standard") identifiers are used for a message object, the identifiers of received data frames are written into bits ID[28:18]. For acceptance filtering, only these bits together with mask bits Msk[28:18] are considered. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
30	MDir	R/W	1h	Mask Message Direction 0 The message direction bit (Dir) has no effect on the acceptance filtering. 1 The message direction bit (Dir) is used for acceptance filtering. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
29	RESERVED	R	1h	Reserved
28-0	Msk	R/W	1FFFFFFFh	Identifier Mask 0 The corresponding bit in the identifier of the message object is not used for acceptance filtering (don't care). 1 The corresponding bit in the identifier of the message object is used for acceptance filtering. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

30.16.3.30 CAN_IF2ARB Register (Offset (x8) = 250h, Offset (x16) = 128h) [Reset = 0h]

CAN_IF2ARB is shown in Figure 30-51 and described in Table 30-40.

Return to the Summary Table.

The bits of the IF1/IF2 Arbitration Registers mirror the arbitration bits of a message object. The Arbitration bits ID[28:0], Xtd, and Dir are used to define the identifier and type of outgoing messages and (together with the Mask bits Msk[28:0], MXtd, and MDir) for acceptance filtering of incoming messages.
A received message is stored into the valid message object with matching identifier and Direction = receive (Data Frame) or Direction = transmit (Remote Frame).
Extended frames can be stored only in message objects with Xtd = one, standard frames in message objects with Xtd = zero.
If a received message (Data Frame or Remote Frame) matches more than one valid message objects, it is stored into the one with the lowest message number.
Note: While Busy bit of IF1/IF2 Command Register is one, IF1/IF2 Register Set is write-protected.

Figure 30-51. CAN_IF2ARB Register

31	30	29	28	27	26	25	24
MsgVal	Xtd	Dir	ID				
R/W-0h	R/W-0h	R/W-0h	R/W-0h				
23	22	21	20	19	18	17	16
ID							
R/W-0h							
15	14	13	12	11	10	9	8
ID							
R/W-0h							
7	6	5	4	3	2	1	0
ID							
R/W-0h							

Table 30-40. CAN_IF2ARB Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MsgVal	R/W	0h	Message Valid 0 The mailbox is disabled. (The message object is ignored by the message handler). 1 The mailbox is enabled. (The message object is to be used by the message handler). The CPU should reset the MsgVal bit of all unused Messages Objects during the initialization before it resets the Init bit in the CAN Control Register. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
30	Xtd	R/W	0h	Extended Identifier 0 The 11-bit ("standard") Identifier is used for this message object. 1 The 29-bit ("extended") Identifier is used for this message object. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

Table 30-40. CAN_IF2ARB Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
29	Dir	R/W	0h	Message Direction 0 Direction = receive: On TxRqst, a remote frame with the identifier of this message object is transmitted. On reception of a data frame with matching identifier, that frame is stored in this message object. 1 Direction = transmit: On TxRqst, the respective message object is transmitted as a data frame. On reception of a remote frame with matching identifier, the TxRqst bit of this message object is set (if RmtEn = one). Note: This bit is write protected by Busy bit. Reset type: SYSRSn
28-0	ID	R/W	0h	Message Identifier ID[28:0] 29-bit Identifier ("Extended Frame") ID[28:18] 11-bit Identifier ("Standard Frame") Note: This bit is write protected by Busy bit. Reset type: SYSRSn

30.16.3.31 CAN_IF2MCTL Register (Offset (x8) = 258h, Offset (x16) = 12Ch) [Reset = 0h]

CAN_IF2MCTL is shown in Figure 30-52 and described in Table 30-41.

Return to the Summary Table.

The bits of the IF1/IF2 Message Control Registers mirror the message control bits of a message object. This register has control/status bits pertaining to interrupts, acceptance mask, remote frames and FIFO option.

Figure 30-52. CAN_IF2MCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
EoB	RESERVED			DLC			
R/W-0h	R-0h			R/W-0h			

Table 30-41. CAN_IF2MCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	NewDat	R/W	0h	New Data 0 No new data has been written into the data portion of this message object by the message handler since the last time when this flag was cleared by the CPU. 1 The message handler or the CPU has written new data into the data portion of this message object. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
14	MsgLst	R/W	0h	Message Lost (only valid for message objects with direction = receive) 0 No message lost since the last time when this bit was reset by the CPU. 1 The message handler stored a new message into this object when NewDat was still set, so the previous message has been overwritten. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
13	IntPnd	R/W	0h	Interrupt Pending 0 This message object is not the source of an interrupt. 1 This message object is the source of an interrupt. The Interrupt Identifier in the Interrupt Register will point to this message object if there is no other interrupt source with higher priority. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
12	UMask	R/W	0h	Use Acceptance Mask 0 Mask ignored 1 Use Mask (Msk[28:0], MXtd, and MDir) for acceptance filtering If the UMask bit is set to one, the message object's mask bits have to be programmed during initialization of the message object before MsgVal is set to one. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

Table 30-41. CAN_IF2MCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	TxIE	R/W	0h	Transmit Interrupt Enable 0 IntPnd will not be triggered after the successful transmission of a frame. 1 IntPnd will be triggered after the successful transmission of a frame. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
10	RxIE	R/W	0h	Receive Interrupt Enable 0 IntPnd will not be triggered after the successful reception of a frame. 1 IntPnd will be triggered after the successful reception of a frame. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
9	RmtEn	R/W	0h	Remote Enable 0 At the reception of a remote frame, TxRqst is not changed. 1 At the reception of a remote frame, TxRqst is set. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
8	TxRqst	R/W	0h	Transmit Request 0 This message object is not waiting for a transmission. 1 The transmission of this message object is requested and is not yet done. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
7	EoB	R/W	0h	End of Block 0 The message object is part of a FIFO Buffer block and is not the last message object of the FIFO Buffer block. 1 The message object is a single message object or the last message object in a FIFO Buffer Block. Note: This bit is used to concatenate multiple message objects to build a FIFO Buffer. For single message objects (not belonging to a FIFO Buffer), this bit must always be set to one. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
6-4	RESERVED	R	0h	Reserved
3-0	DLC	R/W	0h	Data length code 0-8 Data frame has 0-8 data bytes. 9-15 Data frame has 8 data bytes. Note: The data length code of a message object must be defined the same as in all the corresponding objects with the same identifier at other nodes. When the message handler stores a data frame, it will write the DLC to the value given by the received message. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

30.16.3.32 CAN_IF2DATA Register (Offset (x8) = 260h, Offset (x16) = 130h) [Reset = 0h]

CAN_IF2DATA is shown in Figure 30-53 and described in Table 30-42.

Return to the Summary Table.

This register provides a window to the data bytes of the CAN message. The data bytes of CAN messages are stored in the IF1/IF2 registers in the following order. In a CAN Data Frame, Data 0 is the first, and Data 7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first. All bits in this register are write-protected by the Busy bit.

Figure 30-53. CAN_IF2DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data_3								Data_2								Data_1								Data_0							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 30-42. CAN_IF2DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	Data_3	R/W	0h	Data Byte 3 Reset type: SYSRSn
23-16	Data_2	R/W	0h	Data Byte 2 Reset type: SYSRSn
15-8	Data_1	R/W	0h	Data Byte 1 Reset type: SYSRSn
7-0	Data_0	R/W	0h	Data Byte 0 Reset type: SYSRSn

30.16.3.33 CAN_IF2DATB Register (Offset (x8) = 268h, Offset (x16) = 134h) [Reset = 0h]

CAN_IF2DATB is shown in Figure 30-54 and described in Table 30-43.

Return to the Summary Table.

This register provides a window to the data bytes of the CAN message. The data bytes of CAN messages are stored in the IF1/IF2 registers in the following order. In a CAN Data Frame, Data 0 is the first, and Data 7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first. All bits in this register are write-protected by the Busy bit.

Figure 30-54. CAN_IF2DATB Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data_7								Data_6								Data_5								Data_4							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 30-43. CAN_IF2DATB Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	Data_7	R/W	0h	Data Byte 7 Reset type: SYSRSn
23-16	Data_6	R/W	0h	Data Byte 6 Reset type: SYSRSn
15-8	Data_5	R/W	0h	Data Byte 5 Reset type: SYSRSn
7-0	Data_4	R/W	0h	Data Byte 4 Reset type: SYSRSn

30.16.3.34 CAN_IF3OBS Register (Offset (x8) = 280h, Offset (x16) = 140h) [Reset = 0h]

CAN_IF3OBS is shown in Figure 30-55 and described in Table 30-44.

Return to the Summary Table.

The IF3 register set can automatically be updated with received message objects without the need to initiate the transfer from Message RAM by CPU.
The observation flags (Bits [4:0]) in the IF3 Observation register are used to determine, which data sections of the IF3 Interface Register set have to be read in order to complete a DMA read cycle. After all marked data sections are read, the DCAN is enabled to update the IF3 Interface Register set with new data.
Any access order of single bytes or half-words is supported. When using byte or half-word accesses, a data section is marked as completed, if all bytes are read.
Note: If IF3 Update Enable is used and no Observation flag is set, the corresponding message objects will be copied to IF3 without activating the DMA request line and without waiting for DMA read accesses.
A write access to this register aborts a pending DMA cycle by resetting the DMA line and enables updating of IF3 Interface Register set with new data. To avoid data inconsistency, the DMA controller should be disabled before reconfiguring IF3 observation register. The status of the current read-cycle can be observed via status flags (Bits [12:8]).
With this, the observation status bits and the IF3Upd bit could be used by the application to realize the notification about new IF3 content in polling or interrupt mode

Figure 30-55. CAN_IF3OBS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
IF3Upd	RESERVED		IF3SDB	IF3SDA	IF3SC	IF3SA	IF3SM
R-0h	R-0h		R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED			Data_B	Data_A	Ctrl	Arb	Mask
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 30-44. CAN_IF3OBS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	IF3Upd	R	0h	IF3 Update Data 0 No new data has been loaded since last IF3 read. 1 New data has been loaded since last IF3 read. Reset type: SYSRSn
14-13	RESERVED	R	0h	Reserved
12	IF3SDB	R	0h	IF3 Status of Data B read access 0 All Data B bytes are already read out, or are not marked to be read. 1 Data B section has still data to be read out. Reset type: SYSRSn
11	IF3SDA	R	0h	IF3 Status of Data A read access 0 All Data A bytes are already read out, or are not marked to be read. 1 Data A section has still data to be read out. Reset type: SYSRSn

Table 30-44. CAN_IF3OBS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	IF3SC	R	0h	IF3 Status of Control bits read access 0 All Control section bytes are already read out, or are not marked to be read. 1 Control section has still data to be read out. Reset type: SYSRSn
9	IF3SA	R	0h	IF3 Status of Arbitration data read access 0 All Arbitration data bytes are already read out, or are not marked to be read. 1 Arbitration section has still data to be read out. Reset type: SYSRSn
8	IF3SM	R	0h	IF3 Status of Mask data read access 0 All Mask data bytes are already read out, or are not marked to be read. 1 Mask section has still data to be read out. Reset type: SYSRSn
7-5	RESERVED	R	0h	Reserved
4	Data_B	R/W	0h	Data B read observation 0 Data B section not to be read. 1 Data B section has to be read to enable next IF3 update. Reset type: SYSRSn
3	Data_A	R/W	0h	Data A read observation 0 Data A section not to be read. 1 Data A section has to be read to enable next IF3 update. Reset type: SYSRSn
2	Ctrl	R/W	0h	Ctrl read observation 0 Ctrl section not to be read. 1 Ctrl section has to be read to enable next IF3 update. Reset type: SYSRSn
1	Arb	R/W	0h	Arbitration data read observation 0 Arbitration data not to be read. 1 Arbitration data has to be read to enable next IF3 update. Reset type: SYSRSn
0	Mask	R/W	0h	Mask data read observation 0 Mask data not to be read. 1 Mask data has to be read to enable next IF3 update. Reset type: SYSRSn

30.16.3.35 CAN_IF3MSK Register (Offset (x8) = 288h, Offset (x16) = 144h) [Reset = FFFFFFFFh]

CAN_IF3MSK is shown in Figure 30-56 and described in Table 30-45.

Return to the Summary Table.

This register provides a window to the acceptance mask for the chosen mailbox.

Figure 30-56. CAN_IF3MSK Register

31	30	29	28	27	26	25	24
MXtd	MDir	RESERVED	Msk				
R-1h	R-1h	R-1h	R-1FFFFFFFh				
23	22	21	20	19	18	17	16
Msk							
R-1FFFFFFFh							
15	14	13	12	11	10	9	8
Msk							
R-1FFFFFFFh							
7	6	5	4	3	2	1	0
Msk							
R-1FFFFFFFh							

Table 30-45. CAN_IF3MSK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MXtd	R	1h	Mask Extended Identifier 0 The extended identifier bit (Xtd) has no effect on the acceptance filtering. 1 The extended identifier bit (Xtd) is used for acceptance filtering. Note: When 11-bit ("standard") identifiers are used for a message object, the identifiers of received data frames are written into bits ID[28:18]. For acceptance filtering, only these bits together with mask bits Msk[28:18] are considered. Reset type: SYSRSn
30	MDir	R	1h	Mask Message Direction 0 The message direction bit (Dir) has no effect on the acceptance filtering. 1 The message direction bit (Dir) is used for acceptance filtering. Reset type: SYSRSn
29	RESERVED	R	1h	Reserved
28-0	Msk	R	1FFFFFFFh	Identifier Mask Identifier Mask 0 The corresponding bit in the identifier of the message object is not used for acceptance filtering (don't care). 1 The corresponding bit in the identifier of the message object is used for acceptance filtering. Identifier Mask Reset type: SYSRSn

30.16.3.36 CAN_IF3ARB Register (Offset (x8) = 290h, Offset (x16) = 148h) [Reset = 0h]

CAN_IF3ARB is shown in Figure 30-57 and described in Table 30-46.

Return to the Summary Table.

The bits of the IF3 Arbitration Register mirrors the arbitration bits of a message object.

Figure 30-57. CAN_IF3ARB Register

31	30	29	28	27	26	25	24
MsgVal	Xtd	Dir	ID				
R-0h	R-0h	R-0h	R-0h				
23	22	21	20	19	18	17	16
ID							
R-0h							
15	14	13	12	11	10	9	8
ID							
R-0h							
7	6	5	4	3	2	1	0
ID							
R-0h							

Table 30-46. CAN_IF3ARB Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MsgVal	R	0h	Message Valid 0 The message object is ignored by the message handler. 1 The message object is to be used by the message handler. The CPU should reset the MsgVal bit of all unused Messages Objects during the initialization before it resets bit Init in the CAN Control Register. Reset type: SYSRSn
30	Xtd	R	0h	Extended Identifier 0 The 11-bit ("standard") Identifier is used for this message object. 1 The 29-bit ("extended") Identifier is used for this message object. Reset type: SYSRSn
29	Dir	R	0h	Message Direction 0 Direction = receive: On TxRqst, a remote frame with the identifier of this message object is transmitted. On reception of a data frame with matching identifier, that message is stored in this message object. 1 Direction = transmit: On TxRqst, the respective message object is transmitted as a data frame. On reception of a remote frame with matching identifier, the TxRqst bit of this message object is set (if RmtEn = one). Reset type: SYSRSn
28-0	ID	R	0h	Message Identifier ID[28:0] 29-bit Identifier ("Extended Frame") ID[28:18] 11-bit Identifier ("Standard Frame") Reset type: SYSRSn

30.16.3.37 CAN_IF3MCTL Register (Offset (x8) = 298h, Offset (x16) = 14Ch) [Reset = 0h]

CAN_IF3MCTL is shown in Figure 30-58 and described in Table 30-47.

Return to the Summary Table.

The bits of the IF3 Message Control Register mirrors the message control bits of a message object.

Figure 30-58. CAN_IF3MCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
EoB	RESERVED			DLC			
R-0h	R-0h			R-0h			

Table 30-47. CAN_IF3MCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	NewDat	R	0h	New Data 0 No new data has been written into the data portion of this message object by the message handler since the last time when this flag was cleared by the CPU. 1 The message handler or the CPU has written new data into the data portion of this message object. Reset type: SYSRSn
14	MsgLst	R	0h	Message Lost (only valid for message objects with direction = receive) 0 No message lost since the last time when this bit was reset by the CPU. 1 The message handler stored a new message into this object when NewDat was still set, so the previous message has been overwritten. Reset type: SYSRSn
13	IntPnd	R	0h	Interrupt Pending 0 This message object is not the source of an interrupt. 1 This message object is the source of an interrupt. The Interrupt Identifier in the Interrupt Register will point to this message object if there is no other interrupt source with higher priority. Reset type: SYSRSn
12	UMask	R	0h	Use Acceptance Mask 0 Mask ignored 1 Use Mask (Msk[28:0], MXtd, and MDir) for acceptance filtering If the UMask bit is set to one, the message object's mask bits have to be programmed during initialization of the message object before MsgVal is set to one. Reset type: SYSRSn
11	TxIE	R	0h	Transmit Interrupt Enable 0 IntPnd will not be triggered after the successful transmission of a frame. 1 IntPnd will be triggered after the successful transmission of a frame. Reset type: SYSRSn

Table 30-47. CAN_IF3MCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	RxIE	R	0h	Receive Interrupt Enable 0 IntPnd will not be triggered after the successful reception of a frame. 1 IntPnd will be triggered after the successful reception of a frame. Reset type: SYSRSn
9	RmtEn	R	0h	Remote Enable 0 At the reception of a remote frame, TxRqst is not changed. 1 At the reception of a remote frame, TxRqst is set. Reset type: SYSRSn
8	TxRqst	R	0h	Transmit Request 0 This message object is not waiting for a transmission. 1 The transmission of this message object is requested and is not yet done. Reset type: SYSRSn
7	EoB	R	0h	End of Block 0 The message object is part of a FIFO Buffer block and is not the last message object of the FIFO Buffer block. 1 The message object is a single message object or the last message object in a FIFO Buffer Block. Note: This bit is used to concatenate multiple message objects to build a FIFO Buffer. For single message objects (not belonging to a FIFO Buffer), this bit must always be set to one. Reset type: SYSRSn
6-4	RESERVED	R	0h	Reserved
3-0	DLC	R	0h	Data length code 0-8 Data frame has 0-8 data bytes. 9-15 Data frame has 8 data bytes. Note: The data length code of a message object must be defined the same as in all the corresponding objects with the same identifier at other nodes. When the message handler stores a data frame, it will write the DLC to the value given by the received message. Reset type: SYSRSn

30.16.3.38 CAN_IF3DATA Register (Offset (x8) = 2A0h, Offset (x16) = 150h) [Reset = 0h]

CAN_IF3DATA is shown in Figure 30-59 and described in Table 30-48.

Return to the Summary Table.

This register provides a window to the data bytes of the CAN message.

Figure 30-59. CAN_IF3DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data_3								Data_2								Data_1								Data_0							
R-0h								R-0h								R-0h								R-0h							

Table 30-48. CAN_IF3DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	Data_3	R	0h	Data Byte 3 Reset type: SYSRSn
23-16	Data_2	R	0h	Data Byte 2 Reset type: SYSRSn
15-8	Data_1	R	0h	Data Byte 1 Reset type: SYSRSn
7-0	Data_0	R	0h	Data Byte 0 Reset type: SYSRSn

30.16.3.39 CAN_IF3DATB Register (Offset (x8) = 2A8h, Offset (x16) = 154h) [Reset = 0h]

CAN_IF3DATB is shown in Figure 30-60 and described in Table 30-49.

Return to the Summary Table.

This register provides a window to the data bytes of the CAN message.

Figure 30-60. CAN_IF3DATB Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data_7								Data_6								Data_5								Data_4							
R-0h								R-0h								R-0h								R-0h							

Table 30-49. CAN_IF3DATB Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	Data_7	R	0h	Data Byte 7 Reset type: SYSRSn
23-16	Data_6	R	0h	Data Byte 6 Reset type: SYSRSn
15-8	Data_5	R	0h	Data Byte 5 Reset type: SYSRSn
7-0	Data_4	R	0h	Data Byte 4 Reset type: SYSRSn

30.16.3.40 CAN_IF3UPD Register (Offset (x8) = 2C0h, Offset (x16) = 160h) [Reset = 0h]

CAN_IF3UPD is shown in Figure 30-61 and described in Table 30-50.

Return to the Summary Table.

The automatic update functionality of the IF3 register set can be configured for each message object. A message object is enabled for automatic IF3 update, if the dedicated IF3UpdEn flag is set. This means that an active NewDat flag of this message object (e.g due to reception of a CAN frame) will trigger an automatic copy of the whole message object to IF3 register set. Note: IF3 Update enable should not be set for transmit objects.

Figure 30-61. CAN_IF3UPD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IF3UpdEn																															
R/W-0h																															

Table 30-50. CAN_IF3UPD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IF3UpdEn	R/W	0h	IF3 Update Enabled (for all message objects) 0 Automatic IF3 update is disabled for this message object. 1 Automatic IF3 update is enabled for this message object. A message object is scheduled to be copied to IF3 register set, if NewDat flag of the message object is active. Reset type: SYSRSn

30.16.4 CAN Registers to Driverlib Functions

Table 30-51. CAN Registers to Driverlib Functions

File	Driverlib Function
CAN_CTL	
can.c	CAN_initModule
can.c	CAN_setBitTiming
can.h	CAN_startModule
can.h	CAN_enableController
can.h	CAN_disableController
can.h	CAN_enableTestMode
can.h	CAN_disableTestMode
can.h	CAN_setInterruptionDebugMode
can.h	CAN_enableDMARequests
can.h	CAN_disableDMARequests
can.h	CAN_disableAutoBusOn
can.h	CAN_enableAutoBusOn
can.h	CAN_enableInterrupt
can.h	CAN_disableInterrupt
can.h	CAN_enableRetry
can.h	CAN_disableRetry
can.h	CAN_isRetryEnabled
CAN_ES	
can.c	CAN_clearInterruptStatus
can.h	CAN_getStatus
CAN_ERRC	
can.h	CAN_getErrorCount

Table 30-51. CAN Registers to Driverlib Functions (continued)

File	Driverlib Function
CAN_BTR	
can.c	CAN_setBitTiming
can.h	CAN_getBitTiming
CAN_INT	
can.h	CAN_getInterruptCause
CAN_TEST	
can.h	CAN_enableTestMode
can.h	CAN_disableTestMode
can.h	CAN_enableMemoryAccessMode
can.h	CAN_disableMemoryAccessMode
CAN_PERR	
-	
CAN_RAM_INIT	
can.h	CAN_initRAM
CAN_GLB_INT_EN	
can.h	CAN_enableGlobalInterrupt
can.h	CAN_disableGlobalInterrupt
CAN_GLB_INT_FLG	
can.h	CAN_getGlobalInterruptStatus
CAN_GLB_INT_CLR	
can.h	CAN_clearGlobalInterruptStatus
CAN_ABOTR	
can.h	CAN_setAutoBusOnTime
CAN_TXRQ_X	
-	
CAN_TXRQ_21	
can.h	CAN_getTxRequests
CAN_NDAT_X	
-	
CAN_NDAT_21	
can.h	CAN_getNewDataFlags
CAN_IPEN_X	
-	
CAN_IPEN_21	
can.h	CAN_getInterruptMessageSource
CAN_MVAL_X	
-	
CAN_MVAL_21	
can.h	CAN_getValidMessageObjects
CAN_IP_MUX21	
can.h	CAN_getInterruptMux
can.h	CAN_setInterruptMux
CAN_IF1CMD	
can.c	CAN_clearInterruptStatus
can.c	CAN_setupMessageObject

Table 30-51. CAN Registers to Driverlib Functions (continued)

File	Driverlib Function
can.c	CAN_sendMessage
can.c	CAN_sendMessage_16bit
can.c	CAN_sendMessage_32bit
can.c	CAN_sendMessage_updateDLC
can.c	CAN_sendRemoteRequestMessage
can.c	CAN_transferMessage
can.c	CAN_clearMessage
can.c	CAN_disableMessageObject
can.c	CAN_disableAllMessageObjects
CAN_IF1MSK	
can.c	CAN_setupMessageObject
CAN_IF1ARB	
can.c	CAN_setupMessageObject
can.c	CAN_clearMessage
can.c	CAN_disableMessageObject
can.c	CAN_disableAllMessageObjects
CAN_IF1MCTL	
can.c	CAN_setupMessageObject
can.c	CAN_sendMessage
can.c	CAN_sendMessage_16bit
can.c	CAN_sendMessage_32bit
can.c	CAN_sendMessage_updateDLC
can.c	CAN_sendRemoteRequestMessage
CAN_IF1DATA	
can.c	CAN_sendMessage
can.c	CAN_sendMessage_16bit
can.c	CAN_sendMessage_32bit
can.c	CAN_sendMessage_updateDLC
CAN_IF1DATB	
-	See IF1DATA
CAN_IF2CMD	
can.c	CAN_readMessage
can.c	CAN_transferMessage
CAN_IF2MSK	
-	
CAN_IF2ARB	
can.c	CAN_readMessageWithID
CAN_IF2MCTL	
can.c	CAN_readMessage
CAN_IF2DATA	
can.c	CAN_readMessage
CAN_IF2DATB	
-	See IF2DATA
CAN_IF3OBS	
-	

Table 30-51. CAN Registers to Driverlib Functions (continued)

File	Driverlib Function
CAN_IF3MSK	
-	
CAN_IF3ARB	
-	
CAN_IF3MCTL	
-	
CAN_IF3DATA	
-	
CAN_IF3DATB	
-	See IF3DATA
CAN_IF3UPD	
-	

Chapter 31

EtherCAT® Slave Controller (ESC)

This chapter describes the implementation and integration of the Ethernet for Control Automation Technology (EtherCAT®) Slave Controller (ESC). The ESC peripheral can be mapped to either CM or CPU1 subsystems (default access mapped to CPU1 subsystem), and thus either of these subsystems can run the ESC slave stack and application software to implement an ESC slave node.

On this device, the CPU1 is the master subsystem. If the intention is to use the CM subsystem as the owner of ESC, then certain ESC subsystem configurations can only be done by the CPU1 during initialization before allocating the ownership of the ESC to the CM subsystem. These details are explained in this chapter.

Information about the EtherCAT standards and working principles can be in found these documents:

- EtherCAT ET1100 Data sheet
- EtherCAT ESC Hardware Data sheet (Section I)
- EtherCAT IP Core for Xilinx™ FPGAs

Refer to Section 31.1.11 for details on EtherCAT IP errata provided from Beckhoff ® Automation.

31.1 Introduction

Ethernet for Control Automation Technology (EtherCAT®) is an Ethernet-based fieldbus system, invented by Beckhoff ® Automation and is standardized in IEC 61158. All the slave nodes connected to the bus interpret, process, and modify the data addressed to them "on the fly", without having to buffer the frame inside the node. This real-time behavior, frame processing and forwarding requirements are implemented by the EtherCAT slave controller hardware. EtherCAT does not require software interaction for data transmission inside the slave. EtherCAT only defines the MAC layer while the higher layer protocols and stack are implemented in software on the microcontrollers connected to the ESC.

The EtherCAT:

- Involves master and slaves setup where slave nodes are physically connected
- Specializes in precise, low jitter synchronization across slave nodes
- Uses IEEE 802.3 Ethernet physical layer and standard Ethernet frames

A list of relevant terms and definitions are listed in Table 31-1.

Table 31-1. Abbreviations

Name	Definition
CM	Connectivity Manager (referring to Cortex®-M4 subsystem on this MCU)
CPU1	Master C28 CPU1 subsystem on this MCU
DIGIO	Digital IO profile
ENI	EtherCAT Network Information
ESC	EtherCAT Slave Controller
ESCSS or SS	Subsystem (specifically referring to EtherCAT Subsystem on this device)
ESI	EtherCAT Slave Information
ET1100	Beckhoff EtherCAT Slave controller
ETG	EtherCAT Technical Group
EtherCAT	Ethernet for Control Automation Technology
FMMU	Fieldbus Memory Management Units
HAL	Hardware Abstraction Layer
MII	Medium Independent Interface
NMIWDRS	NMI Watchdog Reset
PDI	Processor Data Interface
POR	Power-on-Reset
SSC	EtherCAT Slave Stack Code
SII	Slave Information Interface
WDRS	Watchdog Reset
XRS	External Reset

31.1.1 ECAT Related Collateral

Foundational Materials

- C2000 Academy - EtherCAT
- EtherCAT Device Protocol Poster
- EtherCAT Protocol Series (Video)

Getting Started Materials

- EtherCAT User's Guide
- Real-time control meets real-time industrial communications development - part 4

Expert Materials

- *Design Guide: TIDM-02006 Distributed Multi-axis Servo Drive Over Fast Serial Interface (FSI) Reference Design*
- *EtherCAT Based Connected Servo Drive using Fast Current Loop on PMSM Application Report*
- EtherCAT Technology Group - Download Section Landing Page
- Hardware Data Sheet Section I - Technology
- Hardware Data Sheet Section II - Register Description
- PHY Selection Guide

31.1.2 ESC Features

The ESC on this MCU provides the following functionality.

- Up to 2 MII ports to connect to EtherCAT PHYs
- Process data interface (PDI) through 16-bit asynchronous interface (ASYNC16)
- 64-bit distributed clocking.
  - Sync output signals to synchronize device events and latch input signals supporting time stamping for events.
  - Distributed clock features of SYNC0/1 (o/ps) and LATCH0/1 able to synchronize GPIOs and allow inputs from any GPIOs as well as other muxing options for internal device events.
- 8 Field bus Memory Management Units (FMMUs)
  - Supports all native types of RD/, WR/, RDWR, and built-in features of bit- and byte- addressing
- 8 Sync Managers
- I2C EEPROM interface
- Up-to 32 general-purpose inputs and 32 general-purpose outputs
- 2 SYNC and 2 LATCH signals connected to GPIO pads
- 16-KB RAM with parity

31.1.3 ESC Subsystem Integrated Features

In addition to the ESC features, the following are the device-specific features provided by the integration of the ESC to the MCU:

- ESC access allocation to either CM (Cortex®-M4) subsystem or CPU1 subsystem during initialization
- EtherCAT reset request from master can be routed to NMI or general interrupt controller on MCU
- RAM Parity error routed to NMI on MCU
- DMA access to EtherCAT RAM
- Up to 32 GPI (general-purpose inputs) and up to 32 GPO (general-purpose outputs) feature integrated in addition to 16-bit ASYNC PDI interface
- Interface to CLB
- Distributed clock feature of SYNC0 and SYNC1 able to synchronize PWMs, generate interrupt and DMA requests, trigger ECAP capture, or allow external component action through GPIO access
- EtherCAT SYNC0 and SYNC1 pulse can trigger a CLA task
- Distributed clock feature of LATCH0 and LATCH1 allowing inputs from any GPIO or PWM crossbar triggers

31.1.4 F2838x ESC versus Beckhoff ET1100

Table 31-2 details a comparison between the F2838x ESC IP Core implementation and the Beckhoff's ET1100 ESC.

Table 31-2. F2838x ESC versus Beckhoff ET1100

Feature	F2838x ESC	Beckhoff ET1100
Transmission Speed	100 Mbit/s	100 Mbit/s
Number of Ports	2 (MII Only)	4 (MII or EBUS)
FMMUs	8	8
SyncManagers	8	8
RAM (KByte)	16 KB	8 KB
Distributed Clocks	Yes, 64-bit	Yes, 64-bit
EBUS Support	No	Yes
Process Data Interfaces (PDI)	16-bit ASYNC	Digital I/O SPI 8-bit ASYNC/SYNC 16-bit ASYNC/SYNC

31.1.5 EtherCAT IP Block Diagram

Figure 31-1 shows the general functionality of etherCAT IP from Beckhoff and later on this chapter we will see how this is integrated into the MCU.

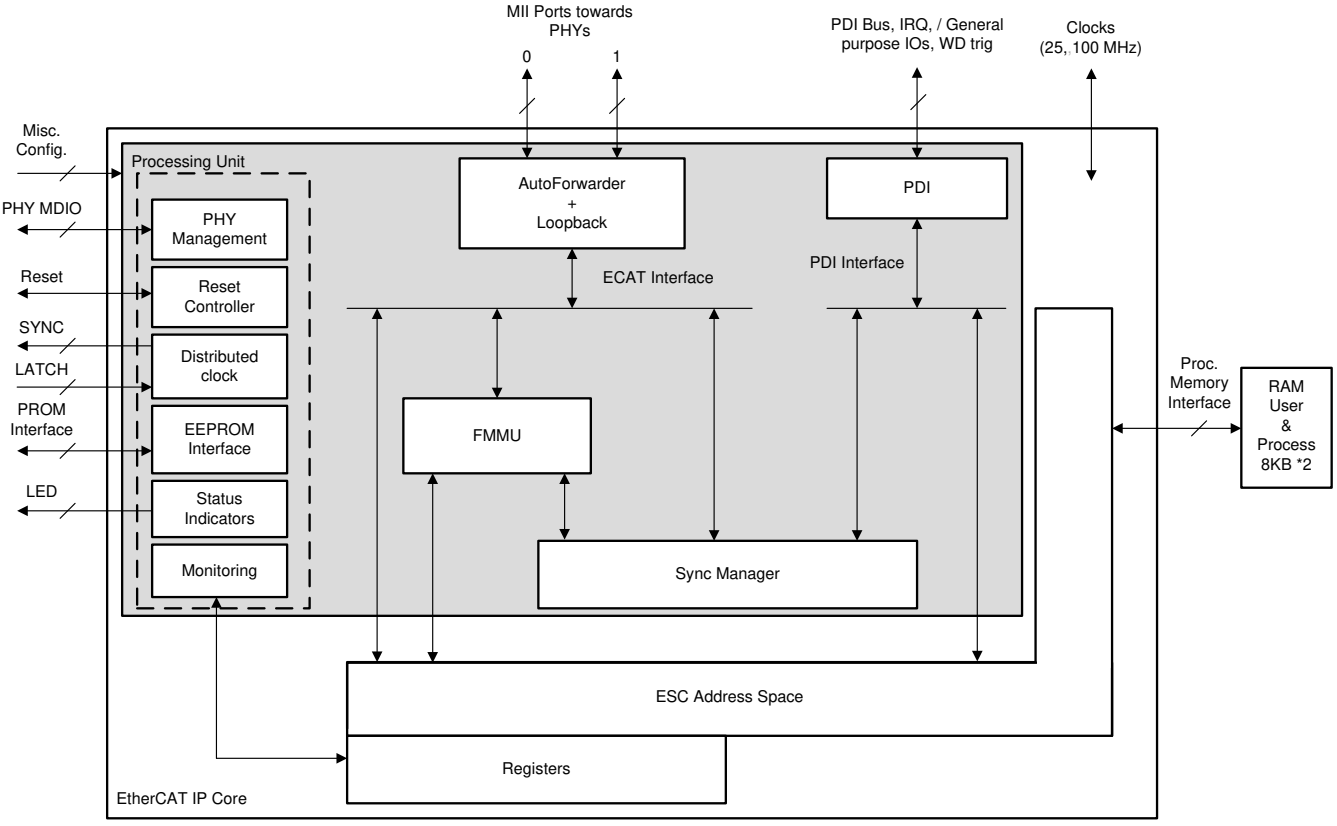


Figure 31-1. EtherCAT IP Block Diagram

31.1.6 ESC Functional Blocks

The following introduces each of the sub-blocks. For more details, refer to the Beckhoff EtherCAT documentation.

31.1.6.1 Interface to EtherCAT Master

The EtherCAT master is a remote entity normally implemented through the device with ethernet switch capability. Beckhoff suggests using either a standard ethernet physical layer interface through MII or RMII protocol, or using a low-voltage differential pair signaling with EBUS protocol. Per Beckhoff specification, up to 4 PHY ports can be supported. In this MCU, implementation of ESC:

- Only MII is supported for Ethernet PHY interface
- ESC supports only 2 PHY ports

31.1.6.2 Process Data Interface

Process data interface (PDI) is a connection from the ESC IP to the microcontroller and slave application. The implementation of the ESC on this MCU supports a 16-bit asynchronous interface (ASYNC16) to the local host only

31.1.6.3 General-Purpose Inputs and Outputs

Since this MCU does not support digital I/O PDI interface, the ESC supports general-purpose I/O that allows 32 GPI and 32 GPO signals. These EtherCAT GPI/GPOs can be programmed to be either driven by (or drive) device I/O pins or be driven by (or drive) the same I/O pins after qualifying them using internal events. Refer to Section 31.2.9 for more details.

31.1.6.4 EtherCAT Processing Unit (EPU)

The EtherCAT Processing Unit (EPU) receives, analyzes, and processes the EtherCAT data stream. The EPU is logically located between port 0 and port 1. The main purpose of the EPU is to enable and coordinate access to the internal registers and the memory space of the ESC, which can be addressed both from the EtherCAT master and from the local application by way of the Processor Data Interface (PDI). Data exchange between the master and slave application is comparable to a dual-ported memory (process memory), enhanced by special functions; for example, for consistency checking (SyncManager) and data mapping (FMMU). The EPU contains the main function blocks of the EtherCAT slaves besides Auto-Forwarding, Loop-back function, and PDI.

- **Auto-Forwarder:**
  - The Auto-forwarder function includes receiving EtherCAT frame, checksum computation and verification, timestamp capture for received frames, and forwarding to Loop-back component.
- **Loop-back Function:**
  - For the two-port implementation on this device, port 3 and port 2 are looped back by design, as shown in Figure 31-2.
  - Port 0 is a special port that is always connected to the master while designing the topology. At egress, port 0 acts as a terminal point for the network connected downstream; hence, any circulating packets get dropped at port 0 in the event of port 0 being looped-back.
  - In the event of a network or link loss downstream to the ESC port 1, the packets end in loopback.

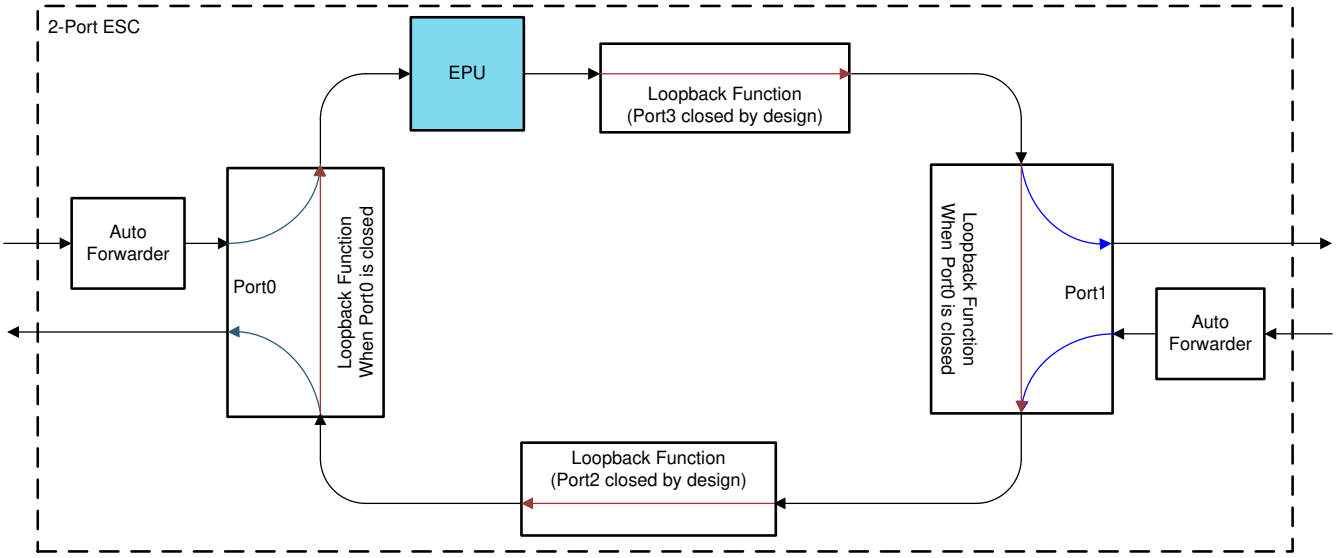


Figure 31-2. Two-port ESC Description

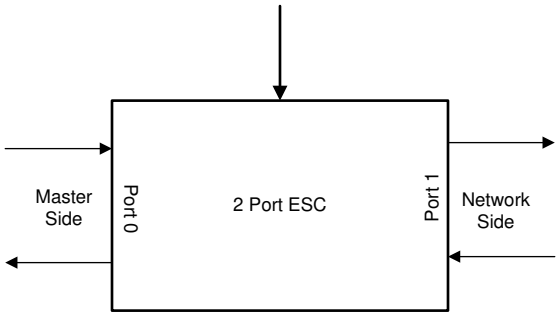


Figure 31-3. Two-port Block Diagram in EtherCAT Topology

31.1.6.5 Fieldbus Memory Management Unit (FMMU)

The fieldbus memory management unit (FMMU) maps the logical addresses of the memory elements in ESC to a physical address that allows the remote EtherCAT master to address registers and process RAM as one memory-map that can be accessed across the EtherCAT network. Mapping can be done up-to bit-wise mapping.

31.1.6.6 Sync Manager

The integrity and security of the data in dual-port memory across the master and local application is maintained by the sync manager. The sync manager allows data to be accessed as buffered or mailbox, where either there is definite read and write space or sequence defined for both of the master entities.

31.1.6.7 Monitoring

The monitoring unit contains error counters and watchdogs. The watchdogs are used for observing communication and returning to a safe state in case of an error. Error counters are used for error detection and analysis.

31.1.6.8 Reset Controller

The reset controller aggregates the internal/external resets as well as asserts the reset to external pin, which can be connected to device pin to reset companion devices on the board.

31.1.6.9 PHY Management

The PHY control is maintained through the MDIO interface which allows the configuration of the PHYs and enabling advanced features of link detection, if present, be enabled.

31.1.6.10 Distributed Clock (DC)

Distributed clocks (DC) allows the tracked time at ESC be synchronized across the EtherCAT network, which in turn makes precisely timed event possible throughout the network. Events could be realized with triggers from ESC outputs which are programmed to create a toggle at certain absolute time tracked by the ESC or through sampling of timestamp of certain system event inputs to ESC which can further be used for synchronizing the time.

- Refer to Section 31.2.10 for more details.

31.1.6.11 EEPROM

Every ESC requires non-volatile memory to store the configuration contents of the ESI (EtherCAT Slave Information) that are accessed by the master normally before enabling the ESC in the network. Size of the memory is configurable with a minimum of 1K bit up to 4M bit are supported depending on the ESC. This non-volatile memory is supported in simplest form by the ESC IP core as serial access EEPROM.

- For ESI EEPROM Layout and mandatory information, refer to the **ESC Hardware Data Sheet Section I - Technology (ethercat_esc_datasheet_sec1_technology)** document provided by Beckhoff/ETG at www.beckhoff.com.
- EEPROM is supported using the I2C interface on this device and is further explained in Section 31.2.8.

31.1.6.12 Status / LEDs

ESC supports the status indication of application activity, link status and link errors which can be utilized for visual status indication through LEDs.

31.1.7 EtherCAT Physical Layer

ESC devices with Ethernet Physical Layer are able to support MII interfaces, RMII interfaces, and EBUS interfaces. Since RMII PHYs include TX FIFOs, the RMII PHYs increase the packet forwarding delay of an ESC device as well as the jitter. RMII as an Ethernet Physical Layer is not supported on this device due to these reasons and therefore only the MII interface is supported.

- On this device, only MII is supported (RMII and EBUS are not supported)
- Refer to the **ESC Application Note - PHY Selection Guide (an_phy_selection_guide)** provided by Beckhoff/ETG at www.beckhoff.com for additional PHY selection details.

Table 31-3 depicts the number of pins needed for MII interface for two ports.

Table 31-3. EtherCAT Physical Layer Signals

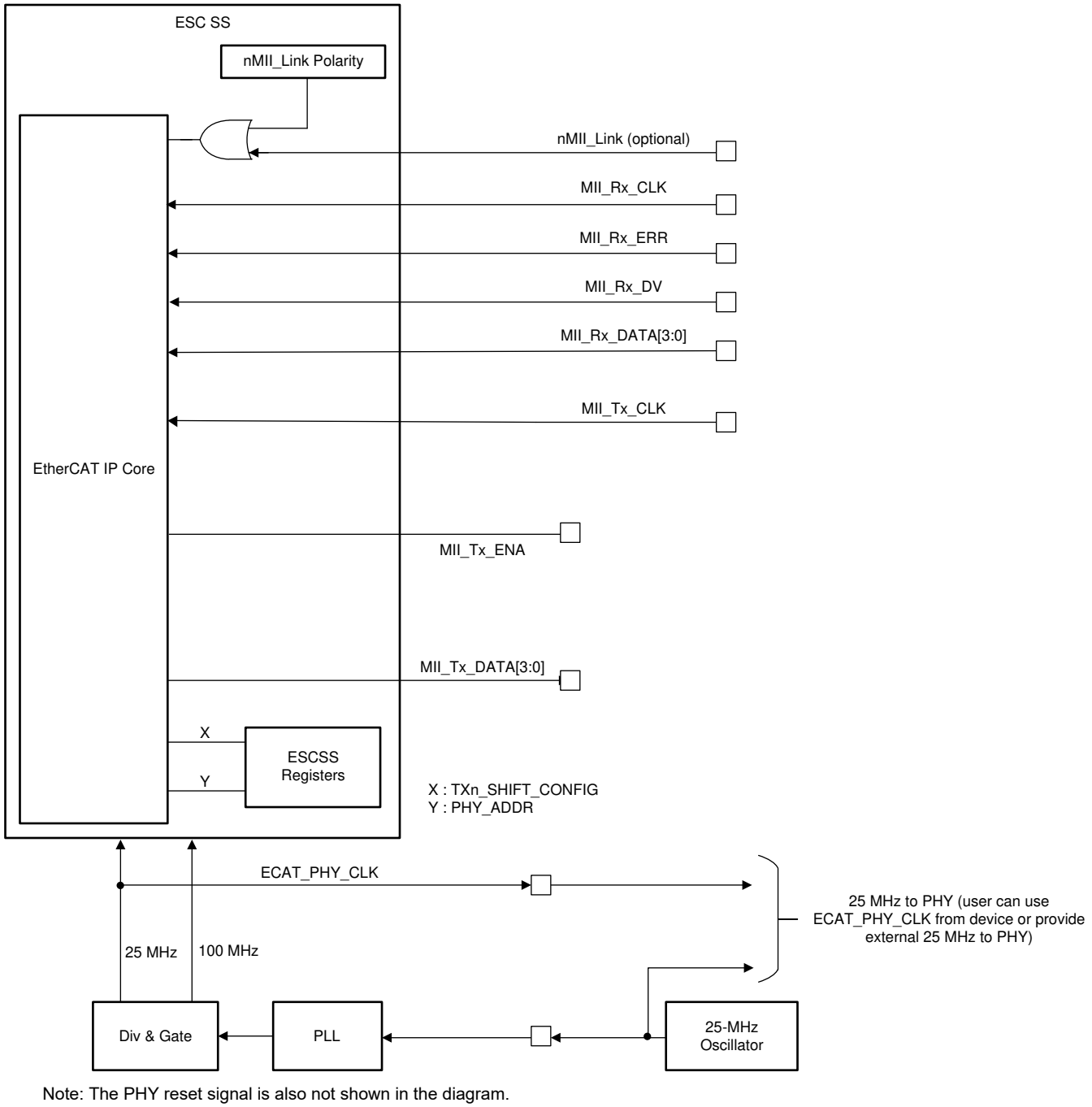
Pin	Number of Pins for 2 Ports	MII	Direction	Description
nMII_Link	2	Yes	IN	Input signal provided by the PHY, if a 100 Mbits/s (full duplex) link is established
RX_CLK	2	Yes	IN	Receive Clock
RX_DV	2	Yes	IN	Receive Data valid
RX_DATA[1:0]	4	Yes	IN	Receive Data
RX_DATA[3:2]	4	Yes	IN	Receive Data
RX_ERR	2	Yes	IN	Receive error
TX_CLK	2	Optional	IN	Transmit Clock
TX_ENA	2	Yes	OUT	Transmit Enable
TX_DATA[1:0]	4	Yes	OUT	Transmit Data
TX_DATA[3:2]	4	Yes	OUT	Transmit Data
MCLK	1	Yes	OUT	MII Interface clock
MDIO	1	Yes	BI-Directional	MII Interface Data
Total Pins:	28 (required) + 2 (optional)			

31.1.7.1 MII Interface

Ethernet IEEE802.3 specified MII interface is supported with possible minor variation in clocking to optimize the clock delay to operate the Tx FIFO. Unlike regular integration of Tx Clock being sourced by the PHY, the ESC implementation allows the common source clock between PHY and ESC be used for Tx logic. The option is selected by manual Tx shift compensation which allows Tx-Data and Tx-En be compensated in steps of 10ns to meet the timing requirements of data sampling at the PHY.

The following signals are used by the ESC to connect to an Ethernet PHY. The MDIO pins are not shown in Figure 31-4, as these pins are covered in the next section.

The MII signals TX_ERR, COL and CRS are not used by the ESC. These are not available on the MCU for EtherCAT.



Note: The PHY reset signal is also not shown in the diagram.

Figure 31-4. ESC PHY Interface Diagram

If an ESC MII interface is not used, LINK_MII has to be tied to the logic value high which indicates no link. RX_CLK, RXD, RX_ER, and especially RX_DV have to be tied to GND and this is taken care of by design internally when the functional IO mux for the RX pins is not configured when the IP is out of reset. The TX outputs can be left unconnected, by not configuring the Functional IO mux for respective EtherCAT functionality, unless they are used for ESC configuration.

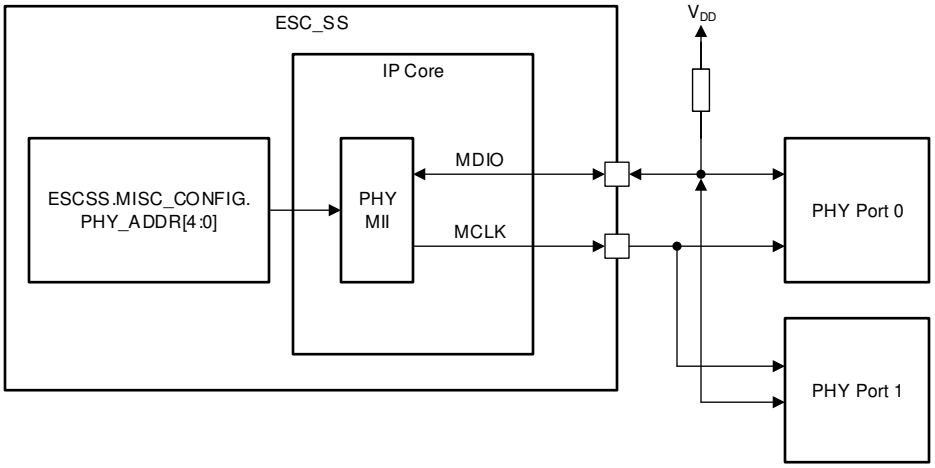
31.1.7.2 PHY Management Interface

Most ESCs with MII/RMII ports use the MII management interface for communication with the Ethernet PHYs. Most ESCs do not use the management interface for link detection or configuration of link modes. For link detection, it is recommended that the ESC use a separate signal (LINK_MII). The MII management interface can be used by the EtherCAT master – or the local MCU. Enhanced MII link detection uses the management interface for restarting auto-negotiation after communication errors occurred.

On this ESC, it is possible to make use of the MII management interface for link detection and PHY configuration.

Note

Note that the EtherCAT link status through the MII management interface option must be enabled only if a Gigabit PHY is to be supported; the EtherCAT link must not be enabled if needing to support both 10/100 and Gigabit PHY. For this reason, the link status through the MII management interface is disabled on this ESC. Instead, use the LINKACT or PHY MII_LINK signals for link status purposes.



Note: The MDIO must have a pull-up resistor (4.7 kOhm recommended) externally. MCLK is driven rail-to-rail, idle value is high.

Figure 31-5. PHY Management Interface Connectivity

31.1.7.2.1 PHY Address Configuration

The ESC addresses the Ethernet PHYs typically using the logical port number plus the PHY address offset. Ideally, the Ethernet PHY addresses must correspond with the logical port number, so PHY addresses 0 and 1 are used.

A PHY address offset of 0 to 31 can be applied which moves the PHY addresses to any consecutive address range. The ESC module expects logical port 0 to have PHY address 0 plus the PHY address offset. The PHY address offset can be selected in register ESCSS_MISC_CONFIG.PHY_ADDR[4:0].

31.1.7.2.2 PHY Reset Signal

The PHY reset signal is generated out of the ESC module. If it is required to release both, the PHY and ESC module synchronous out of reset, this signal can be used. Since there are no pull devices active on the MCU during and after reset, a pull down resistor must be added on this signal on the board level.

In some case, PHYs may be released from reset after releasing the ESC module. To generate a delay, the pin for nPHY_RESET can be used as an I/O and shall be switched later to the alternate output function.

31.1.7.2.3 PHY Clock

The PHY Clock connectivity is shown in Figure 31-4. On this MCU, the user has option to clock the PHY using the ESCSS_PHY_CLK signal if needed otherwise user can chose to provide an external 25 MHz source to the PHY and ESC (both must be clocked from the same source). For more details regarding providing PHY clock, refer to Section 31.2.6.2 and see the EtherCAT data sheet from Beckhoff.

Note

Beckhoff documents strict accuracy (ppm) requirements for the shared 25-MHz ESC and PHY clock. Special care must be taken within a design to make sure the EtherCAT clocking requirements are met. Relying on external components (oscillator and clock buffer) for the 25-MHz source are recommended. Validation of a design during normal operating conditions can be necessary to confirm clocking requirements are adequately met, especially if using the ESCSS_PHY_CLK signal.

31.1.8 EtherCAT Protocol

EtherCAT uses standard IEEE 802.3 Ethernet frames, thus a standard network controller can be used and no special hardware is required on the master side. EtherCAT has a reserved EtherType of 0x88A4 that distinguishes this from other Ethernet frames. Thus, EtherCAT can run in parallel to other Ethernet protocols.

- EtherCAT does not need the IP protocol; however, the EtherCAT can be encapsulated in IP/UDP.
- The ESC processes the frame in hardware; therefore, communication performance is independent from processor power.

An EtherCAT frame is subdivided into the EtherCAT frame header followed by one or more EtherCAT datagrams. At least one EtherCAT datagram has to be in the frame. Only EtherCAT frames with Type 1 in the EtherCAT Header are currently processed by the ESCs. The ESCs also support IEEE802.1Q VLAN Tags, although the VLAN Tag contents are not evaluated by the ESC.

- If the minimum Ethernet frame size requirement is not fulfilled, padding bytes have to be added. Otherwise, the EtherCAT frame is exactly as large as the sum of all EtherCAT datagrams plus EtherCAT frame header.
- For further reading, including details on network time or network propagation delay, refer to the **ESC Hardware Data Sheet Section I - Technology (ethercat_esc_datasheet_sec1_technology)** document provided by Beckhoff/ETG at www.beckhoff.com.

31.1.9 EtherCAT State Machine (ESM)

The EtherCAT State Machine (ESM) is responsible for the coordination of the master and slave applications at start up and during operation. State changes are typically initiated by requests of the master. The state changes are acknowledged by the local application after the associated operations have been executed. Unsolicited state changes of the local application are also possible.

There are four states (see Figure 31-6) an ESC can support, plus one optional state:

- Initialize (Init)
- Pre-Operational
- Safe-Operational
- Operational
- Bootstrap (Optional)

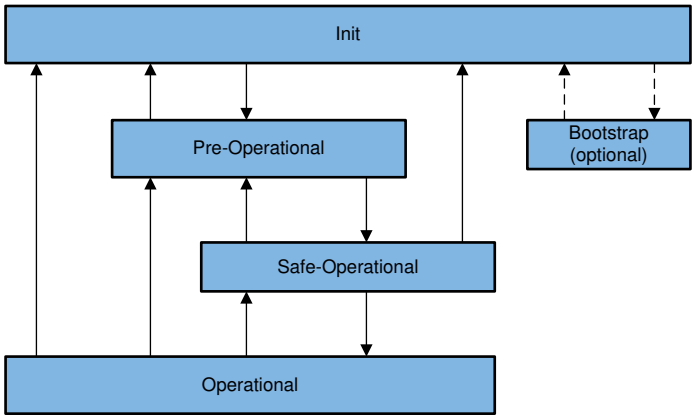


Figure 31-6. EtherCAT State Machine

Note

Not all state changes are possible, for example, the transition from Init to Operational requires the following sequence: Init → Pre-Operational → Safe-Operational → Operational.

31.1.10 More Information on EtherCAT

- For further information on EtherCAT, refer to the EtherCAT specifications, available from the EtherCAT Technology Group (ETG, www.ethercat.org).
- Refer to the IEC standard “Digital data communications for measurement and control – Fieldbus for use in industrial control systems”, IEC 61158 Type 12: EtherCAT, available from the IEC (www.iec.ch).
- Documentation on Beckhoff Automation ESC are available at the Beckhoff website (www.beckhoff.com), for example, data sheets, application notes, and ASIC pinout configuration tools.

31.1.11 Beckhoff® Automation EtherCAT IP Errata

Table 31-4 details the Beckhoff ®Automation errata of the EtherCAT IP integrated onto this device.

Table 31-4. EtherCAT IP Errata

Errata Number	Description
ER#156	WKC increment for reading reserved FMMU registers (+0xD-+0xF)
ER#158	WKC increment for writing 0x0914-0x0917
ER#162	ESC DL Control (0x0101) loop setting Auto-Close (01): if port waits for write access to be opened, the temporary loop bit (0x0100[1]) is not taken into account when a write command to ESC DL Control occurs.
ER#164	EEPROM FSM cannot accept another command within 1680 ns after finishing a previous command
ER#165	EtherCAT reset register 0x0040 cannot be written using VLAN tagged frames, because state is reset due to double ECAT_CLEAR. VLAN tagged frames are rarely used for EtherCAT.
ER#168	Changing SyncSignal cycle times during activation can lead to extremely long cycle times, resulting from (intermediate) violation of the minimum delay between two consecutive pulses. For example, changing Sync1 from a few ns before Sync0 to a few ns after Sync0 results in a missed Sync1 time, which can take up to several 32/64-bit turn-arounds until the next pulse occurs. Changing cycle time from PDI can result in usage of intermediate, inconsistent values, which can result in unwanted cycle times or extremely long delays as above.

31.2 ESC and ESCSS Description

This section details the aspects of the ESC integration in the MCU. On this MCU, the ESC is integrated such that the ESC is accessed by either the CM subsystem or the CPU1 (master) subsystem. The MCU with the ESC functions as an EtherCAT slave device.

Figure 31-7 shows the ESC on this MCU.

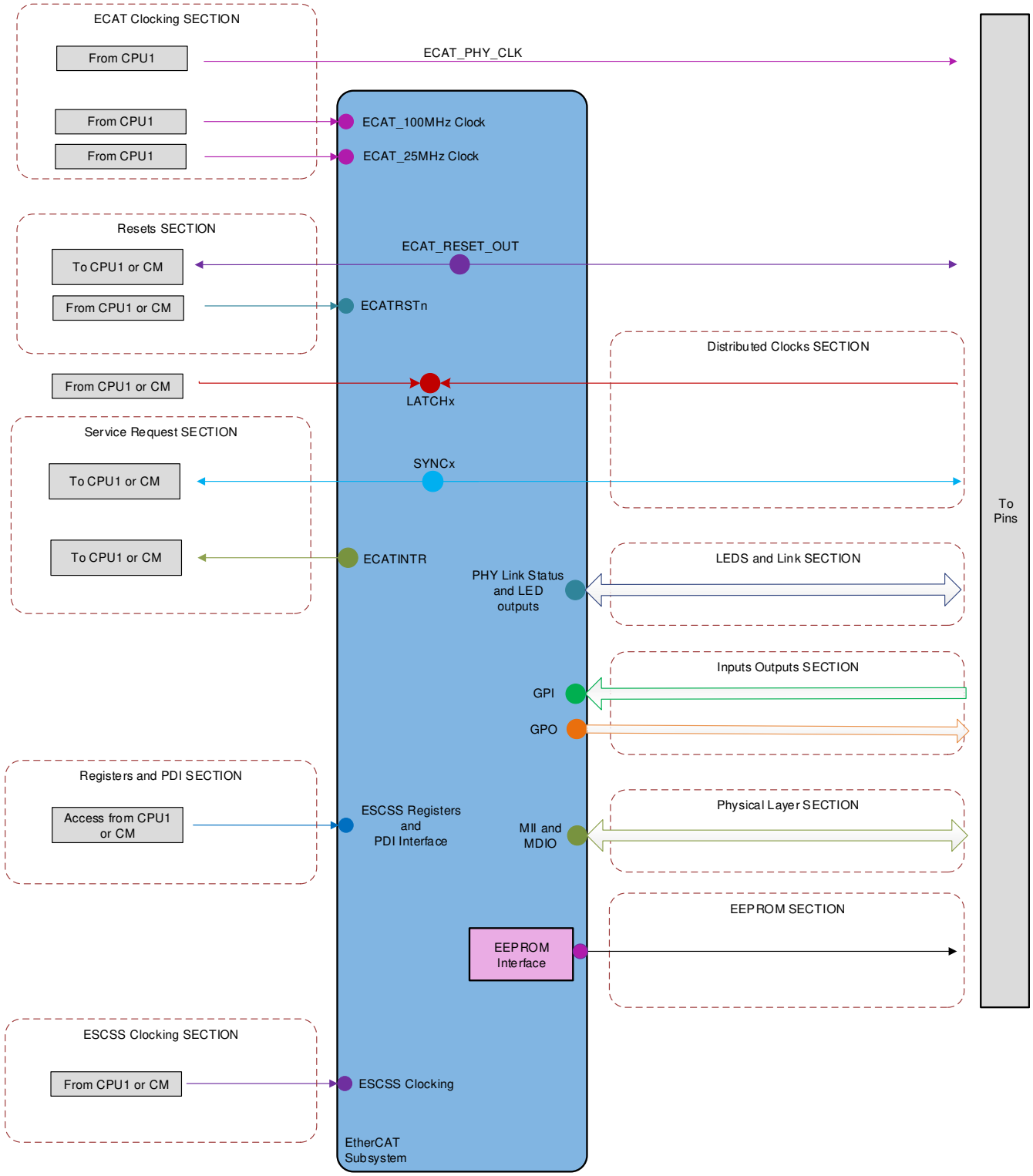


Figure 31-7. ESC Integration on MCU

Table 31-5. ESC Integration Figure Sections

ESC Figure Sections	More Information Links
ECAT/ESCSS Clocking	ESC Clocking
Resets	ESCSS Resets
Service Request	SYNC Signals Interrupts and Interrupt Mapping
Registers and PDI	ESC SubSystem
Distributed Clocks	Distributed Clocks - Sync and Latch
LEDS and Link	LED Controls
Inputs Outputs	General Purpose Inputs and Outputs
Physical Layer	EtherCAT Physical Layer
EEPROM	Slave Node Configuration and EEPROM

There are various actions that the controlling processor (CPU1/CM) needs to perform in response to actions initiated by the EtherCAT master meant for this particular slave. All these interactions take place through one of the following:

- The 16-bit asynchronous interface through which registers and RAM of the EtherCAT IP are read/written.
- An interrupt request going from the ESC to the Local Host (CM or CPU1).
- SYNC0 and SYNC1 pulses generated by the ESC that are used as triggers for initiating further action.
- LATCH0 and LATCH1 that are sampled by the ESC.
- EtherCAT general-purpose inputs and outputs.

31.2.1 ESC RAM Parity and Memory Address Maps

This section details the ESC RAM parity logic, CPU1 ESC memory address map, and CM ESC memory address map.

31.2.1.1 ESC RAM Parity Logic

The ESC RAM has parity logic to ensure the data integrity of the contents. The byte-wide parity is implemented as PDI accesses to RAM could be done byte wide. To ensure that errors in parity logic itself do not mask the memory error, redundant parity generation logic is used. The parity generated by both logics is compared to ensure safe operation of the memory accesses. Use the INITIATE_MEM_INIT bit from the ESCSS register to trigger the memory initialization.

31.2.1.2 CPU1 ESC Memory Address Map

When the ESC is allocated to CPU1, Table 31-6 is the memory map and associated details.

Table 31-6. ESC Address Map on CPU1

Memory Map Region	Memory Region	Accessed Through	Parity Scheme	DMA Access	CLA Access	CPU2 Access	Security	Access/ Bus
DPRAM memory map (16 KB)	0x00050800 - 0x000527FF	PDI interface of ESC	1 parity bit for 8 bits of data	Yes	No	No	No	Async
ESC Registers	0x00050000 - 0x000507FF	PDI Interface of ESC	No	No	No	No	No	Async
ESCSS Registers(1)	0x00057E00 - 0x00057FFF	Direct Access	No	No	No	No	No	VBUS32

(1) When register access protection is enabled, all write accesses are blocked; however, a violation interrupt is not generated in the event of an attempted write access.

31.2.1.3 CM ESC Memory Address Map

When ESC is allocated to CM, Table 31-7 is the memory map and features.

Table 31-7. ESC Address Map on CM

Memory Map Region	Memory Region	Accessed Through	Parity Scheme	µDMA Access	Bus/Protocol	Security
ESC RAM memory map (16 KB)	0x400A1000 - 0x400A4FFF	PDI interface of ESC	1 parity bit for 8 bits of data	Yes	System/PDI	No
ESC Registers	0x400A0000 – 0x400A0FFF	PDI	No	No	System/PDI	No
ESC_SS Registers	0x400AFC00 – 0x400AFFFF	Direct Access	No	No	System/VBUSP	No

31.2.2 Local Host Communication

Figure 31-8 shows how the ESC and ESC subsystem connects to the local host subsystem. The local host subsystem can include the CPU (CPU1 or CM) of the device and the DMA engine as the bus master accessing the ESC. This local host subsystem is acting upon the commands that are sent by the EtherCAT master in the form of EtherCAT datagrams.

The 16-bit asynchronous interface and the interrupt request line are the main channels through which the local host subsystem interacts with the ESC. The interrupt request line can be used to trigger actions based on ESC internal events, exception conditions or time synchronized events. The DMA engine is used to transfer the contents from the process data memory to system RAM on any of these events. The user application needs to select the events that act as interrupts or DMA requests. The mask and clear events for the interrupt causes are configured to be accessed by the CPU/Co-processor that is controlling the ESC.

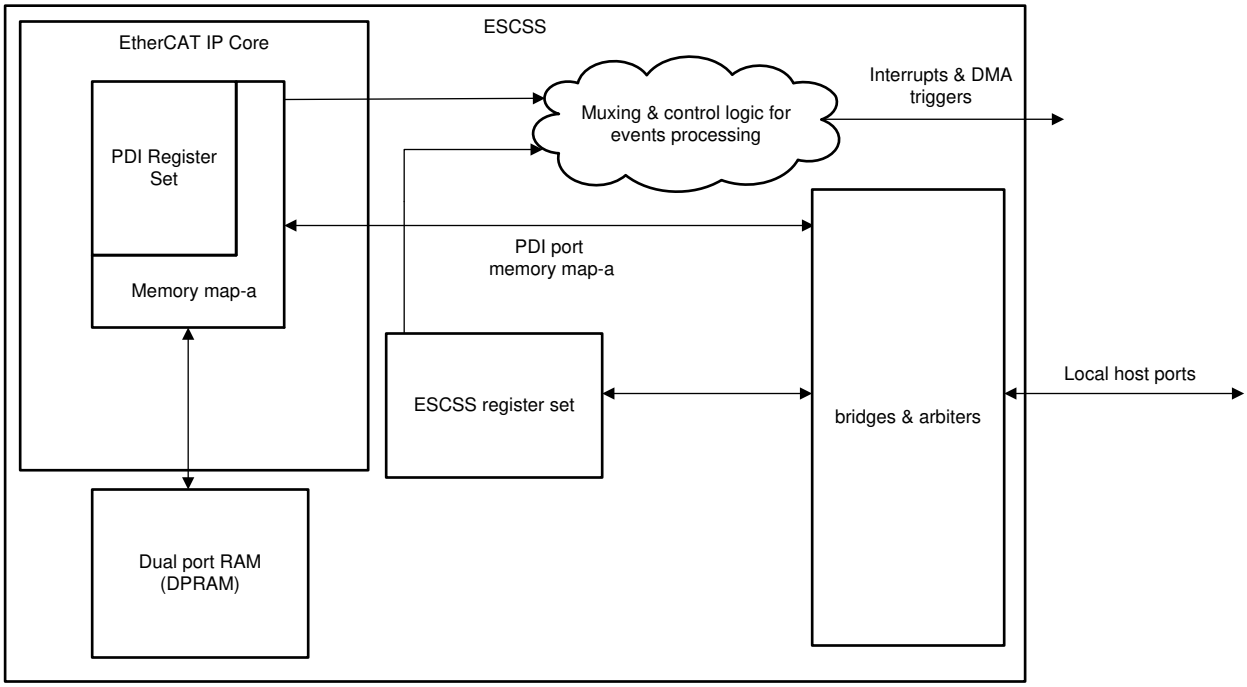


Figure 31-8. Interaction of ESCSS with the CPU Subsystem

31.2.2.1 Byte Accessibility Through PDI

The ESC has a few registers which need byte wide access to affect the functionality. The EtherCAT implementation on this MCU is as follows:

- CPU1 supports 16-bit accesses only
- CM supports byte accesses

31.2.2.2 Software Details for Operation Across Clock Domains

The registers accessible from the CPU are in the system clock domain and are synchronized to the PDI clock before being used within the EtherCAT IP. Given the frequency ratios and different styles of synchronization schemes, the application needs to assume there is a delay in getting the values transferred from one domain to other. Such a delay can vary based on the frequency of system clock and type of synchronizer used in the path.

For example, a system clock of 200 MHz and ESCSS running at 100 MHz requires at least 10 clock-system clock cycles delay before values are affected on the other side. If a write occurs to the PDI and some other action is performed elsewhere (not involving the PDI), then the software must perform a simple read to make sure that the write is complete.

31.2.3 Debug Emulation Mode Operation

There are two aspects of the EtherCAT IP operation that need to be considered from a debug emulation stand point.

- **CPU Halted Condition:**
  - **CPU1:** For any operation from the EtherCAT IP that is based off an interrupt request, it is quite important that this interrupt is marked as a real-time interrupt and the debugger must be halted in RealTime mode. If the CPU is halted without taking the above precaution, then the EtherCAT IP can only be active for those parts where servicing the interrupt is not required (For example, GPIO and SYNC0/1 output triggers can all function unaffected).
  - **CM** Does not apply. CM does not have real-time debug and remains in halt mode until released by the run command, regardless of interrupt request assertions.
- **Debugger Writes/Reads of EtherCAT IP registers/memories Condition:** The EtherCAT IP does not have any mechanism to identify a debug initiated read/write. Debug accesses to the registers or the ESC RAM can affect the state of the EtherCAT IP. This is addressed by the following:
  - **CPU1/CM:** The ENABLE_DEBUG_ACCESS bit must be set in the ESCSS Access Control Register to enable user access to the ESC RAM and EtherCAT registers for the purpose of debug. By default, this is disabled.

31.2.4 ESC SubSystem

The EtherCAT Slave Controller SubSystem (ESCSS) wraps the ESC core with required register configurations and required logic for different functions of the EtherCAT and RAM (ESC RAM). This section covers the ESCSS integration aspects. Figure 31-9 shows the EtherCAT subsystem Integration.

- **ESCSS configuration registers interface** is the port for accessing the critical device level configurations which are a must have for the ESCSS to function.
- **ESCSS register interface** has control and status registers including SYNC, LATCH configurations, interrupt related controls, and GPIO related controls.
- **PDI Async interface** is a 16-bit wide data interface that allows the local application to access the registers internal to the EtherCAT IP Core as well as the dual-port memory (ESC RAM) through the SyncManager and the FMMUs.

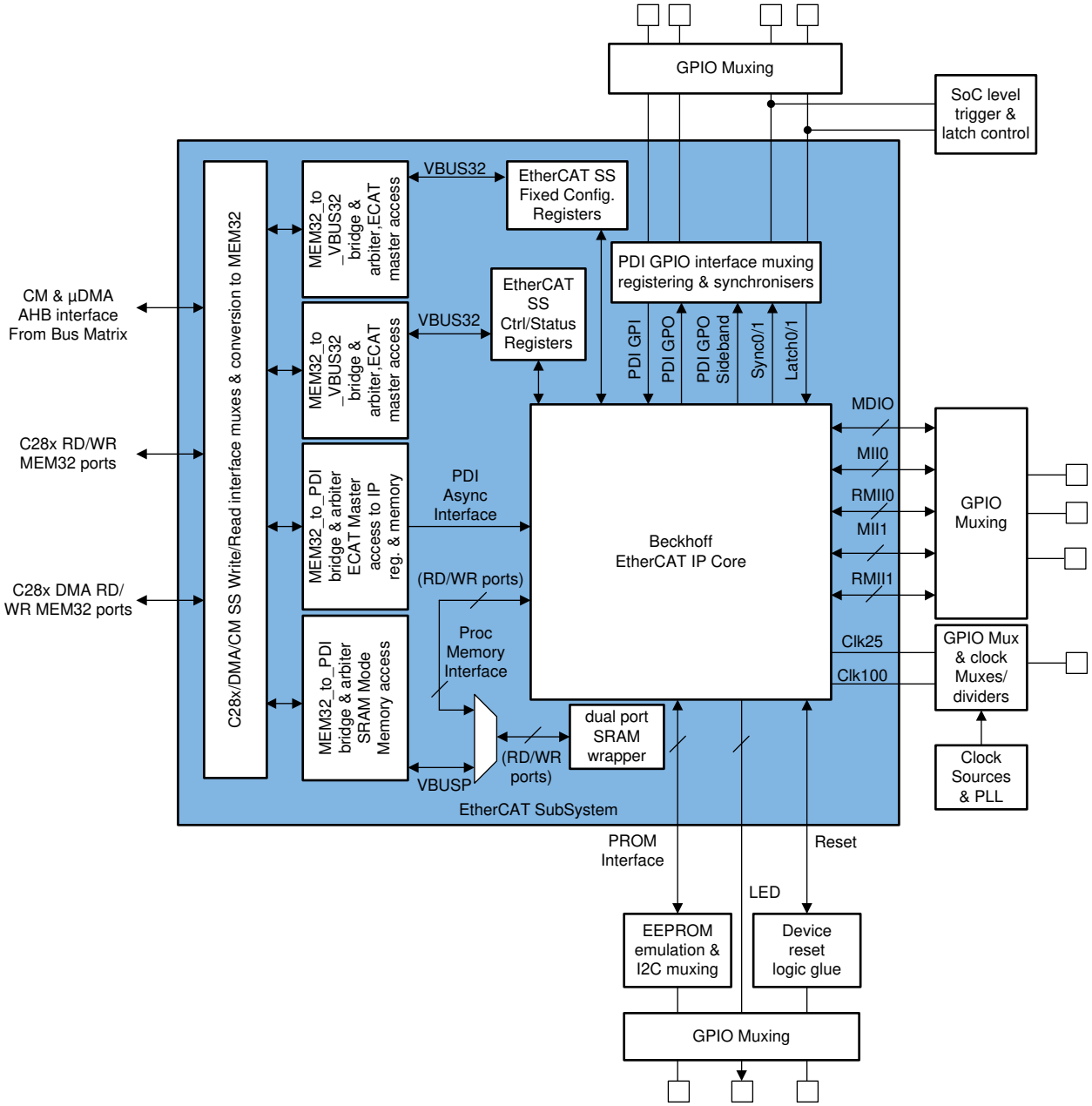


Figure 31-9. ESCSS Wrapper

31.2.4.1 CPU1 Bus Interface

Interface to the CPU1 core is retained at the ESCSS boundary as a native mem32 interface that connects the RD and WR ports independently to all of the slave entities within the ESCSS.

- CPU1 MEM32 RD and WR ports: Configuration Registers, Control/Status Registers, PDI
- CPU1 DMA MEM32 RD and WR ports: PDI (access to ESC RAM only)

31.2.4.2 CM Bus Interface

The CM interface involves the AHB bus that is arbitrated between the CM and µDMA accesses to the PDI .

- CM System bus ports: Configuration Registers, Control/Status Registers, PDI
- µDMA ports: PDI (access to ESC RAM only)

31.2.5 Interrupts and Interrupt Mapping

The ESC has one interrupt line that can be connected to the local host (CPU1 or CM) that is called PDI IRQ from the ESC. Besides this, there are other interrupt causes generated within the ESCSS. These are aggregated to a total of 4 interrupt lines that are connected to the local host core.

Table 31-8 summarizes these exceptions and their mapping. There are 4 interrupt lines provided for ESCSS which are: ECATSS_Intr, RESET_OUT_Intr, SYNC0_Intr and SYNC1_Intr. Each of these causes have an independent set of mask/clear/set controls and raw/masked interrupt status flags. This allows independent cause servicing on the exception event with flexibility to mask or service the desired set of causes. All these interrupts are mapped onto NVIC on CM and PIE on CPU1. Refer to the *System Control and Interrupts* chapter for details on the interrupt numbers.

In the event that the same exception is available to the multiple bus master cores of the local host CPU, the application software is expected to make sure that clearing of the RIS is a software synchronized event between those masters and there is no stale exceptions pending for one master while the other clears the RIS. In other words, there is no separate exception cause (RIS/MIS) copy per master but are common interrupt registers for the local host across masters.

Table 31-8. Service Request Generation Map

Source	Description	Master				
		CPU1	CLA	CPU1 DMA	CM	µDMA
EtherCAT IRQ	AL Event Request of the ESC	ECATSS_ intr	ECATSS_ intr	Not Available	ECATSS_ intr	Not Available
PDI Interface Timeout Error	PDI Interface WatchDog timeout error	ECATSS_ intr	ECATSS_ intr	Not Available	ECATSS_ intr	Not Available
µDMA Done(1)	This event indicates to CM or CPU1 that the µDMA transfer by earlier event is over	ECATSS_ intr	Not Available	Not Available	ECATSS_ intr	Not Available
EtherCAT RESET_ OUT Event	RESET_OUT can be programmed as interrupt to the CPU to either complete the reset sequence with required pre-steps if an or acknowledge the reset request in some other way. Given the high priority nature of the signal independent interrupt line is allocated	RESET_ OUT_Intr	RESET_ OUT_Intr	Not Available	RESET_ OUT_Intr	Not Available
SYNC0 Event	Precise time event 2, can be used to start routine SYNC1_Intron cores or data transfer using DMA. Given the precise time and priority of these interrupts a separate interrupt line is dedicated.	SYNC0_ Intr	SYNC0_ Intr	SYNC_ DMAReq	SYNC0_ Intr	SYNC_ DMAReq
SYNC1 Event	Precise time event 1, can be used to start routine on cores or data transfer using DMA. Given the precise time and priority of these interrupts a separate interrupt line is dedicated.	SYNC1_ Intr	SYNC1_ Intr	SYNC_ DMAReq	SYNC1_ Intr	SYNC_ DMAReq

(1) Only µDMA has the DMA_DONE qualifier. This is not applicable to the CPU1 DMA. The DMA_DONE cause is specific to µDMA integration in the systems and is routed to the CPU that is the master of the EtherCAT. While µDMA configuration is limited by the CM core, in cases where µDMA request is linked to time precise SYNC triggers, the EtherCAT master must be able to clear the DMA_DONE RIS (RAW Interrupt Status) to re-arm the µDMA request generation in ESCSS. If unused, this interrupt cause must be kept masked by the EtherCAT master CPU.

31.2.6 Power, Clocks, and Resets

This section details the ESCSS power requirements, clocking, and resets. The EtherCAT IP has 2 clock ports and one reset output. Since the PLLs, dividers, and gating is implemented as part of system control, this setup is always handled by CPU1 during the initialization sequence.

31.2.6.1 Power

The ESC module is inside the main power domain like other peripherals and there are no special considerations about the power-up or power-down sequences that need to be taken. Refer to the *C28x System Control and Interrupts* chapter for different power modes.

31.2.6.2 Clocking

Two functional clock inputs are defined for the ESC: CLK25 (25 MHz) and CLK100 (100 MHz).

EtherCAT functional clock inputs CLK25 and CLK100 are sourced by the MCU clocking module either using SYSPLL or AUXPLL. At the SoC level, you have multiple options to choose from regarding what inputs are used for the SYSPLL or AUXPLL.

- Due to the 25 ppm requirement for EtherCAT, an external oscillator of 25 MHz is suggested to be used as the main clock source.
- A less accurate clock than 25 ppm limits the ability of the ESC to act as the network reference clock. For practical reasons, clock accuracy must be the same or better than the Ethernet clock source that is 50 ppm.

Note

- CLK25 and CLK100 are used for the ESC core operation. These two clocks and the clocks to the PHY **must have a common source**, which establishes a deterministic phase relationship between PHY clocks and ESC clocks.
- In MII mode, 25 MHz RX clock is sourced by the PHY while the TX clock from PHY is optional (which effectively saves a pin). The phase differential between the PHY TX clock and CLK25 can be compensated to the TX data and TX EN using the manual compensation mode in increments of 10 ns.

Figure 31-10 shows the clocks connectivity. Refer to the *System Control and Interrupts* chapter for additional details of the PLL source selections and clock dividers.

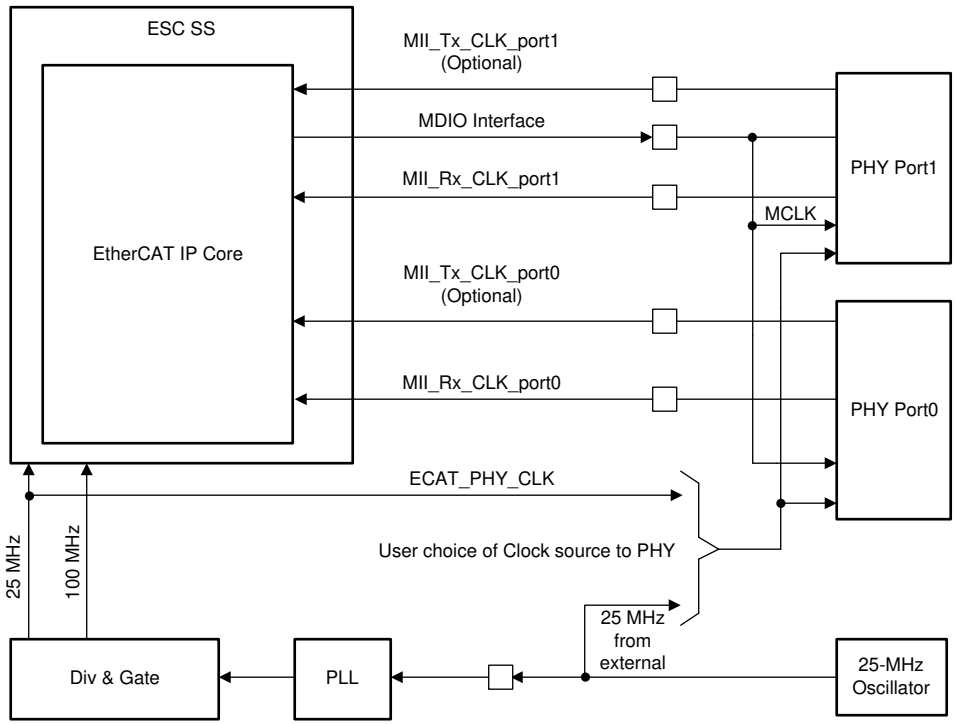


Figure 31-10. Clocking of ESC

In the case of a low-accuracy clock (up to 100 ppm) being used, a clock accuracy of 25 ppm cannot be feasible, then the following restrictions apply:

- RX FIFO size cannot be reduced lower than 7 (default for 100 ppm), which in turn affects the latency of the transfer as the transfer starts after half of the FIFO full threshold, thus resulting in a RX Delay.
- This ECS cannot be used as the first slave from the master in the network, as typically the first slave from the master is treated as the reference clock for the distributed clocks functionality.
- The number of iterations required for synchronizing clocks (specially the drift computation) increases.

31.2.6.3 Resets

This section describes the ESCSS sources of reset. There are other conditional reset sources that can impact ESCSS and those include all the device level reset sources including: device pin reset “XRSN” or software initiated device level reset “SYSRSn”.

Note

Whenever the device comes out of reset, software must execute the following sequence to make sure that PHY can see the full stretching of the reset:

1. Configure the EtherCAT GPIOs
2. Put the ESCSS into soft reset
3. Bring the ESCSS out of soft reset

Refer to Section 31.3 more information on proper software procedure.

31.2.6.3.1 Chip-Level Reset

Chip-level resets (such as POR, XRS, WDRS, or NMIWDRS) are a master reset that resets the entire device including the ESC.

31.2.6.3.2 EtherCAT Soft Resets

ESCSS can be reset by three soft reset mechanisms, including:

- System control level software programmable (SOFTPRES23) that is reset only by chip-level and EtherCAT resets. Default keeps the ESC in reset that is released by the local application after setting the ESC ready for operation.
- Reset by a particular command sequence from the remote EtherCAT master.
- Reset by a particular command sequence from the local host application. (CM core only)

31.2.6.3.3 Reset Out (RESET_OUT)

Multiple reset sources of ESCSS are combined to make the reset vector that drives the EtherCAT and companion devices (PHYs) on this MCU. The RESET_OUT is asserted only when configured to be a reset. If configured as an interrupt or NMI, the local host completes the reset.

By default, RESET_OUT performs no action and must be setup to cause the reset to the EtherCAT core and PHY. The input events to RESET_OUT include the remote EtherCAT master, local host application, and EtherCAT soft reset.

When the intent is to perform a reset to the ESCSS and the device, the following options are available to reset the device after software has given a reset to the ESCSS:

- Use the system control SIMRESET register to generate a device reset.
- Use the watchdog or NMIWD to generate a device reset.

31.2.7 LED Controls

There are four LED outputs from the ESC that can be routed out of GPIO to indicate different status of the ESC. Table 31-9 is the recommendation for applications on the usage, especially for the pin sensitive systems that can

support this and still meet the intent. All the LED functions are routed to the peripheral pin mux and the user must configure the LEDs as per the available pins.

Table 31-9. Status LED Options and Priority

LED	Priority Order	Function	Recommendation	Usage Priority
RUN	5	Shows the status of ESC state machine with different blinking patterns.	In the case where color LEDs are not supported, then this needs to be supported.	Must
ERR(1)	4	Indicates the Error in ESC operation.	In the case where color LEDs are not supported, then at least the Error status needs to be reported.	Must
STATE	2	Combination of the RUN and ERR LED.	Not supported on this device.	-
LINKACT0 (ESC_LED_LINK0_ACTIVE)	3	Link Active Status for link towards the master.	Link Status on master side (Port0) is important to know, if the ESC and network downstream is part of network or not. The prior ESC will loopback, if this node fails. The status on this LED eases the debug.	Desired
LINKACT1 (ESC_LED_LINK1_ACTIVE)	1	Link Active status for link towards Network side.	The Link status on the network side (Port 1) is critical from continuity through the ESC point; hence, this is kept highest priority status reporting.	Must

(1) Driven by software and not the ESC on this device.

The PHY MII_LINK (ESC_PHYx_LINKSTATUS) indication is an indication from PHY and does not indicate if the established link meets the characteristics including auto-negotiation as required by EtherCAT. Hence, it may not be a true status indication; however, it is critical to shorten reaction time in the event of link loss that is crucial for redundancy operation. Using LINKACT LEDs (LED_LINKx_ACTIVE) for the status output and MII_LINK (ESC_PHYx_LINKSTATUS) for the status input is the best usage; however, they may not be practical to sacrifice 4 pins. The tradeoff of whether MII_LINK is used as a link status LED or LINKACT is used depends on link loss reaction time requirements and available number of GPIOs for the system.

The PHY MII_LINK (ESC_PHYx_LINKSTATUS) signal is an active-low signal input to the C2000 ESC. It is possible to use the GPxINV register to invert the signal's polarity before the signal enters the ESC, if the system hardware cannot provide the desired polarity for this signal.

The LINKACT LEDs (LED_LINKx_ACTIVE) is an output signal of the C2000 ESC intended to control an LED for Link indication.

Table 31-10. LINKACT and PHY MII_LINK States

Link State	LINKACTx (LED_LINKx_ACTIVE) State	PHY MII_LINK (ESC_PHYx_LINKSTATUS) State
No Link	0 (LOW)	1 (HIGH)
Link WITHOUT Activity	1 (HIGH)	0 (LOW)
Link WITH Activity	Toggling / Blinking	0 (LOW)

Note

The EtherCAT link status through the MII management interface option must be enabled if only a Gigabit PHY is to be supported and must not be enabled if needing to support both 10/100 and Gigabit PHY. For this reason, the link status through MII management interface is disabled on this ESC. Instead, use the LINKACT or PHY MII_LINK signals for link status purposes.

31.2.8 Slave Node Configuration and EEPROM

The ESC requires non-volatile memory (for example, an EEPROM with I2C interface), which is referred to as the Slave Information Interface (SII) memory, to store the EtherCAT Slave Information (ESI) data.

The ESI data contains information about the slave device identification, supported features, and other network accessible properties.

Note

The ESI XML file is generated from Beckhoff's Slave Stack Code (SSC) tool.

- **EtherCAT Slave**
  - Uses the SII upon power-on or reset to load configuration data into the EtherCAT slave configuration registers.
- **EtherCAT Master**
  - During development, the master is used to program the device SII memory with the ESI data.
  - When the ESI file isn't provided to the master, the master reads the device SII memory upon boot-up to determine the slave device properties such as the process data and their mapping options, the supported mailbox protocols including optional features, as well as the supported modes of synchronization.

31.2.9 General-Purpose Inputs and Outputs

The ESC subsystem implementation on this MCU supports 32 general-purpose inputs (GPI) and 32 general-purpose outputs (GPO) in addition to the PDI.

These GPI/GPOs provide an advantage for embedded applications over discrete components because the GPI/GPOs can selectively drive or sense GPIOs in a time controlled manner. For example, it could be a sensor/actuator control, status read, LED status update, and so on. The following section describes the integration and usage of this feature.

31.2.9.1 General-Purpose Inputs

General-purpose inputs are connected to the device GPIO pin mux through an input buffer with an option to directly connect the asynchronous input flowing directly into the ECS or through a ESCSS register pipeline, which can hold the value stable while the value is captured within the ESC. The state of the GPI can be captured based on the following events:

- **Start-of-Frame (SOF):** Contents can be ready for capture in the frame if requested by the master
- **SYNC0/1:** Contents captured on a precise time tick
- **LATCH0/1:** Contents synchronized with LATCHIN read (allows simultaneous input and timestamp capture)

The ESCSS_GPIN_DAT register, containing the contents of the GPIN data, is available for reading from host CPU to allow debug access. Additionally, ESCSS_GPIN_DAT allows CPU writes for being used GPIN override purposes such as in place of using GPIOs.

GPIs are divided into 4 sets: GPI0:7, GPI8:15, GPI16:23, and GPI24:31, for clocking the capture trigger, which means the same capture trigger has to be used for the IOs within a set. Thus, either a bus can be formed out of these or individual IOs which need to be aggregated under common trigger can be combined in one set. This allows limited freedom of trigger selection for inputs. Selection of which Inputs (ESCSS_GPIN_SEL) and outputs (ESCSS_GPOUT_SEL) can be connected to GPIO is possible at each single IO level.

Figure 31-11 shows the integration of the GPI feature.

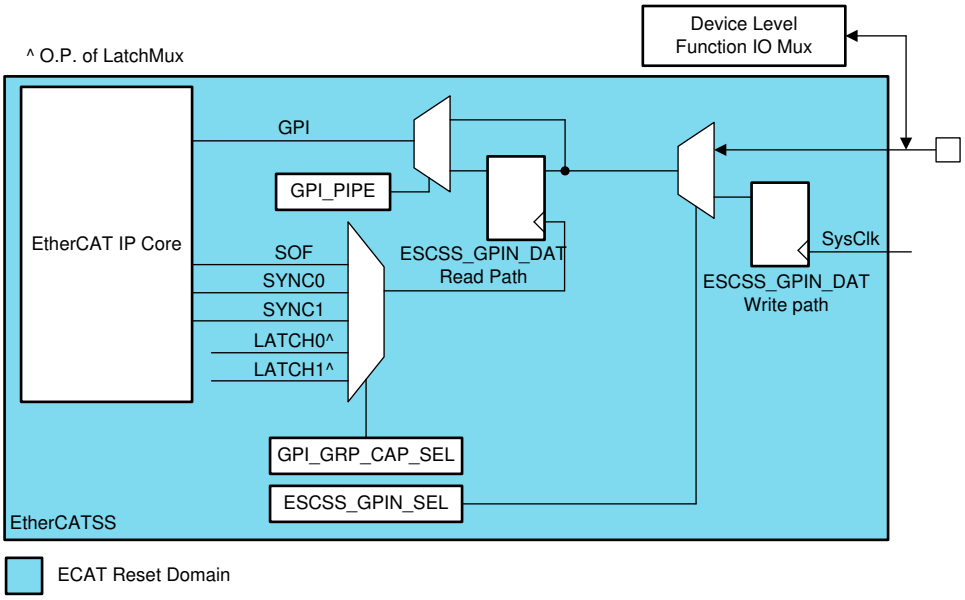


Figure 31-11. ESCSS General-Purpose Inputs Integration

Note

For software: The write data gets reflected on the ESCSS_GPIN_DAT read path only when one of the configured events, like SOF, SYNC0/1, and so on, is seen as configured after a few cycles of that event getting generated. Additionally, the copy of ESCSS_GPIN_DAT register readable by the CPU is provided without synchronization; therefore, multiple reads must be performed to confirm a stable value before using the value.

31.2.9.2 General-Purpose Output

General-purpose outputs are connected to the device GPIO pin mux through an output buffer with an option to directly connect from the ESC IP core registers to the external world or through a ESCSS register pipeline, which can hold the output value stable. The state of the GPO can be set based on the following events:

- **End-of-Frame:**Contents can be updated by the master from the last received frame
- **SYNC0/1:** Contents can be updated based on a precise time tick
- **Watchdog Trigger:** Contents can be updated based on the last successful access to process data memory

The ESCSS_GPOUT_DAT register, containing the contents of the GPO data, is available for reading from host CPU to allow debug access or to read in place of GPIOs.

GPOs are divided into 4 sets: GPI0:7, GPI8:15, GPI16:23, and GPI24:31, for clocking the output trigger, which means the same output trigger has to be used for the IOs within a set. Thus, either a bus can be formed out of these or individual IOs which need to be aggregated under common trigger can be combined in one set. This allows limited freedom of trigger selection for inputs. Selection of which inputs (ESCSS_GPIN_SEL) and outputs (ESCSS_GPOUT_SEL) can be connected to GPIO is possible at each single IO level.

Figure 31-12 shows the integration of the GPO feature.

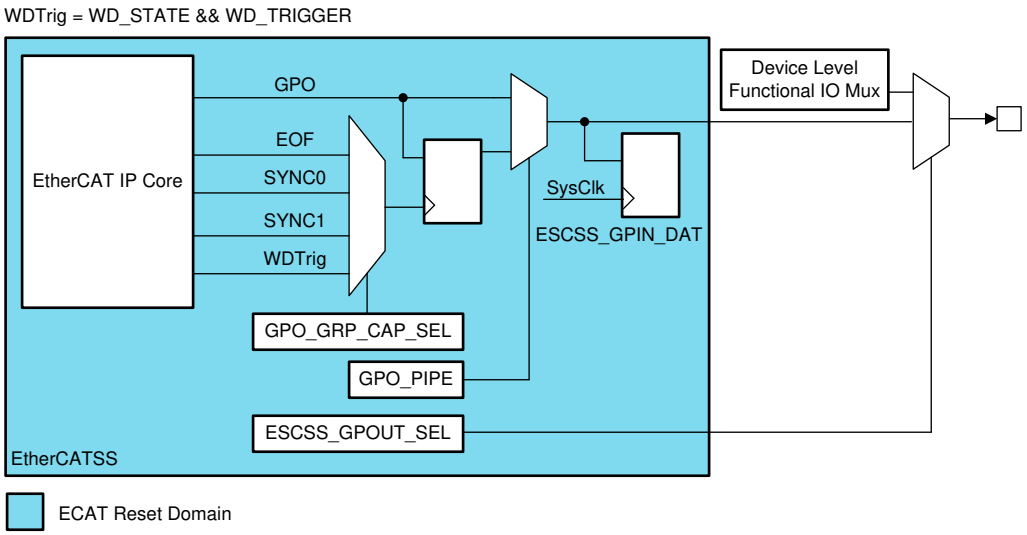


Figure 31-12. ESCSS General-Purpose Output Integration

Note

In Figure 31-12, the synchronization between CLK100 or CLK25 with respect to SysClk is not shown and is implicit by design. The local host can adjust for the synchronization delays as required while processing the data in either direction.

31.2.10 Distributed Clocks – Sync and Latch

Distributed clocks (DC) is a differentiating feature of the EtherCAT network. Distributed clocks allows all the slave nodes in the EtherCAT network to be bound in a tight margin of time to synchronize the events and the EtherCAT master command.

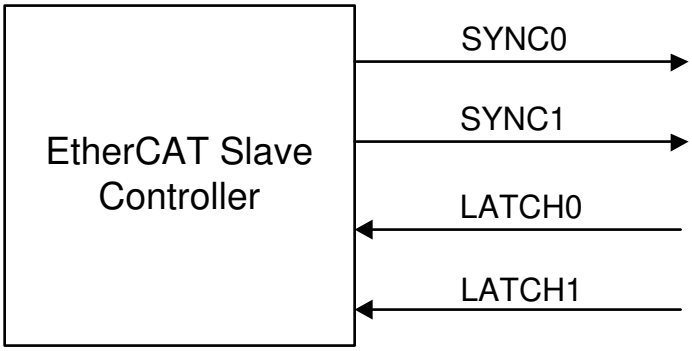


Figure 31-13. ESC SYNC and LATCH

This feature has sub-features as listed below. To enable the utility of these features at the application level, the related signals of DC are integrated tightly with the C2000 control loop logic, as well as, an external component that can be a possible implementation to trigger/latch other components on the board. This section explains the integration of these signals and related nuances of usage.

Distributed clock features:

- Clock Synchronization
- Sync Signal Generation
- Latch Signal event capture

Apart from usage for control loop synchronizations, these features are also used to synchronize the system and network time. These include the following:

- **System Time PDI Controlled:** Synchronize system time between two different EtherCAT networks.
- **Communication Timing:** To synchronize slave communication with sync signals, output or input events, and so on.

For further reading, including details on network time or network propagation delay, refer to the **ESC Hardware Data Sheet Section I - Technology (ethercat_esc_datasheet_sec1_technology)** document provided by Beckhoff/ETG at www.beckhoff.com.

31.2.10.1 Clock Synchronization

DC clock synchronization is the ability and procedure of the ESC to maintain the copy of the reference clock based on local clock (internal 64-bit time base) and other adjustment as derived by the procedure. The reference clock is the most accurate clock in the system and is typically held by one of the slaves (Topologically first one after master is preferred). This time-base has higher accuracy requirement and it needs to periodically synchronize with an absolute time source like GPS or any other maintained time-base as in IEEE1588 network.

The synchronization order is as follows:

1. The master maintains its own master clock that is either the absolute time synchronized or controls the reference clock.
2. The master queries the ESC time-base based on topology information and timestamps to determine the drift of the local clock for each ESC with regards to the reference clock.
3. The master programs the clock drift adjustments in each respective ESC.
4. With further periodic queries of the local clock, the master determines the drift of the local clock and keeps adjusting the clock so as to maintain the copies of the reference clocks on each ESCs in a tight limit.

While the ESC requires two clocks of 25 MHz and 100 MHz, the 100-MHz clock is used for the internal time-base and supports the best accuracy possible. Refer to Section 31.2.6.2 for more details on clocking.

31.2.10.2 SYNC Signals

Sync signals are precise time controlled signals and are able to trigger time synchronized actions of the master device. CPU core and DMA engine triggers are configured independently for every end point. The following sections describe ESC signal integration and various control triggers in the device.

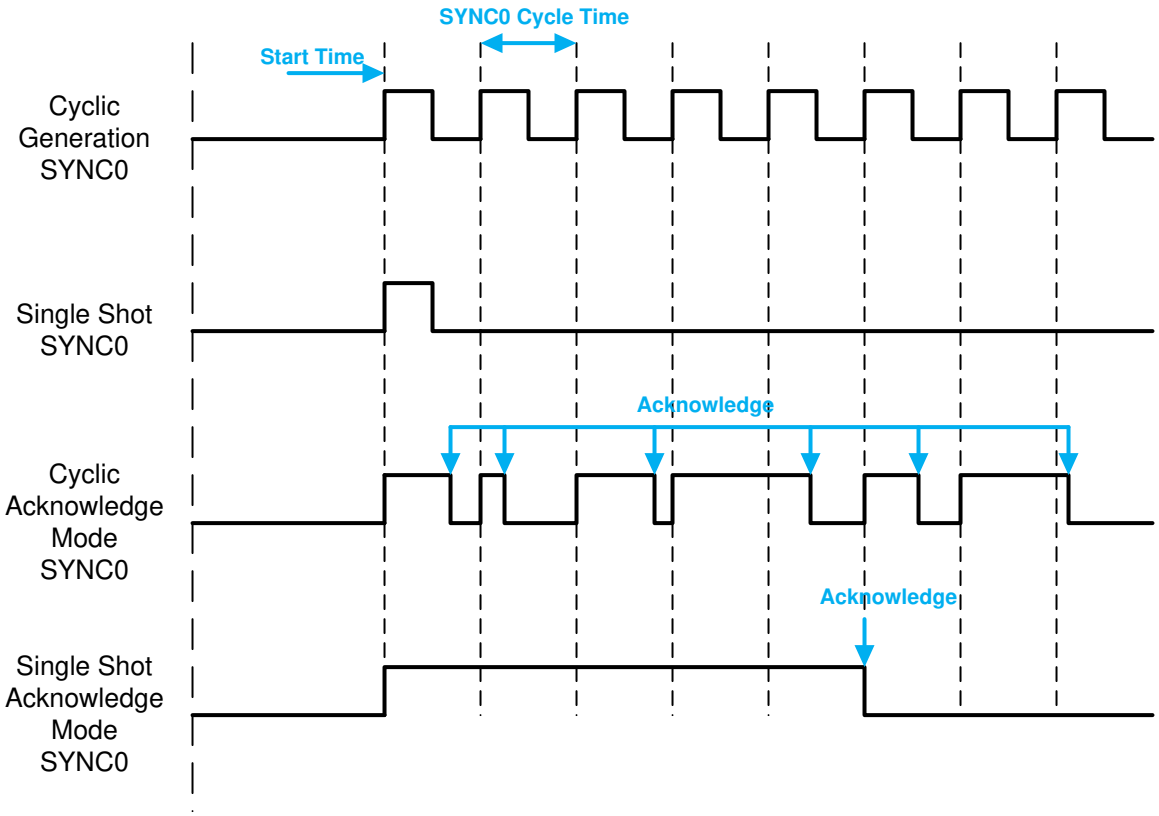


Figure 31-14. SYNC0 Signal Modes

- **SYNC0:** SYNC0 is the primary trigger and can be generated in either a cyclic (periodic event generation like rise edge at every x period) or one-shot mode. Furthermore, these modes can be either with or without acknowledge, such that when enabled with acknowledge mode, the next event is not generated until the acknowledgment is received from the controlling master. If the acknowledgment is delayed, the event is skipped and the next periodic event is generated. Remember that the acknowledgment can be part of the interrupt servicing from the PDI and such a delay in triggering the next event is acceptable since the servicing routine can accordingly take action for the period elapsed. These modes are shown in Figure 31-14.
- **SYNC1:** The SYNC1 generation follows the SYNC0 generation with a programmable delay and depending upon the delay time defined, SYNC1 can or cannot generate the pulse since the predefined delay from the SYNC0 event is newly measured only after the SYNC1 event (except for the start).

Note

- The SYNC0/1 when used in a system clock domain is stretched to at least 3 clock-wide pulse.
- When ESC goes through a reset, the SYNC outputs triggering external device events need to be kept at a safe value. At reset, the output values are 0 and this can be an active state when the reset occurs; therefore, the corresponding care of usage from the remote device must be taken.

31.2.10.2.1 Seeking Host Intervention

SYNC can be used to initiate the action from the local host or related DMA engine that can respond with control action or data transfers from the ESC IP.

These actions can be selectively initiated in terms of:

- Interrupts to CPU cores (CPU1, CLA, or CM)
- DMA requests to DMA cores (CPU1 DMA and CM µDMA)

Figure 31-15 depicts the SYNC connection diagram to the CPU subsystems.

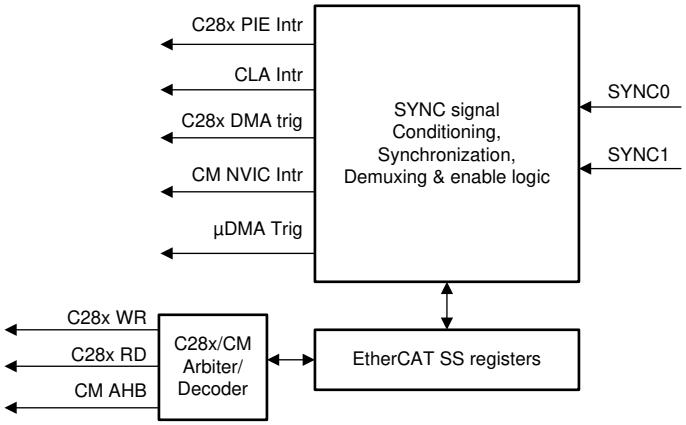


Figure 31-15. SYNC Integration for the HOST Intervention

Table 31-11 shows the Host Intervention selections, control and information that helps applications to route and handle the SYNC signals for respective Host intervention.

Table 31-11. ESC SYNC Integration Map

Destination	Source	Enable	Mask	Clear	Source Clock	Destination Clock	Destination Signaling
C28x PIE Interrupt	SYNC0	ESCSS_SYNC0_CONFIG[0]	ESCSS_INTR_MASK[0]	ESCSS_INTR_CLR[0]	ECAT.100MHz	C28x.SysClk	Pulse
C28x PIE Interrupt	SYNC1	ESCSS_SYNC1_CONFIG[0]	ESCSS_INTR_MASK[1]	ESCSS_INTR_CLR[1]	ECAT.100MHz	C28x.SysClk	Pulse
CLA Interrupt	SYNC0	ESCSS_SYNC0_CONFIG[1]	NA	NA	ECAT.100MHz	C28x.SysClk	Pulse
CLA Interrupt	SYNC1	ESCSS_SYNC1_CONFIG[1]	NA	NA	ECAT.100MHz	C28x.SysClk	Pulse
C28x DMA Trigger	SYNC0	ESCSS_SYNC0_CONFIG[2]	NA	NA	ECAT.100MHz	C28x.SysClk	Pulse
C28x DMA Trigger	SYNC1	ESCSS_SYNC1_CONFIG[2]	NA	NA	ECAT.100MHz	C28x.SysClk	Pulse
CMNVIC Interrupt	SYNC0	ESCSS_SYNC0_CONFIG[3]	ESCSS_INTR_MASK[0]	ESCSS_INTR_CLR[0]	ECAT.100MHz	CM.SysClk	Pulse/Level
CM NVIC Interrupt	SYNC1	ESCSS_SYNC1_CONFIG[3]	ESCSS_INTR_MASK[1]	ESCSS_INTR_CLR[1]	ECAT.100MHz	CM.SysClk	Pulse/Level
µDMA Trigger	SYNC0	ESCSS_SYNC0_CONFIG[4]	NA	NA	ECAT.100MHz	CM.SysClk	Pulse/Level
µDMA Trigger	SYNC1	ESCSS_SYNC1_CONFIG[4]	NA	NA	ECAT.100MHz	CM.SysClk	Pulse/Level

The difference between Enable and Mask is that Enable allows the conditioned and synchronized interrupt to be routed to the raw interrupt/trigger cause register, while Mask is a software control to allow raising an interrupt or not. Disabling the SYNC0 and SYNC1 on a respective trigger can loose any events that happen until SYNC0 and SYNC1 are enabled again.

Regarding the CLA:

- The CLA does not have access to EtherCAT, so the CLA does not have the MASK and Clear controls; however, the trigger starts the action on the CLA processing which, upon completion, must be acknowledged by CPU1 to clear the cause.

Regarding the DMA:

- The DMA triggers do not have Mask and Clear capabilities, as typically there is no feedback mechanism from the DMA to clear the trigger cause.
- The expectation is that DMA Done is routed to ESCSS and raises an interrupt to the CPU for acknowledgment. CPU1 upon clearing the trigger cause, acknowledges the SYNC through PDI.
- Figure 31-16 is the symbolic view of DMA request source select. If a given source event triggers multiple events across different masters, then the software has to make sure there is a status exchange before the cause is cleared.

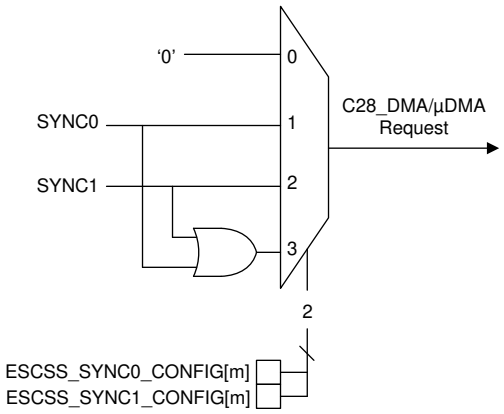


Figure 31-16. SYNC Event Muxing for Different Host DMA Triggers

31.2.10.3 LATCH Signals

Latch inputs to the ESC can be used to capture the time-stamp or register the GPI inputs. Two latch inputs are available. Latch enables external events:

- To take a system time snapshot of the EtherCAT network, measure time, or schedule further operations.
- To capture the status of the GPI inputs either as the status of connected external components or as part of control flow.

Both of the Latch inputs can be independently configured to operate on either the rising or falling edge of the signal. The LATCH events can be individually assigned either for the PDI control or EtherCAT master control. Additionally, one shot or continuous mode is supported.

- In **one shot mode**, the next capture of the timestamp is done on a qualifying LATCH event only after the previous event is acknowledged.
- In **continuous mode**, timestamps are successively captured on a qualifying event, regardless of a preceding event being read or not.

Figure 31-17 depicts the different sources for the LATCH0/1 so as to fulfill the application requirements. These are explained for possible use.

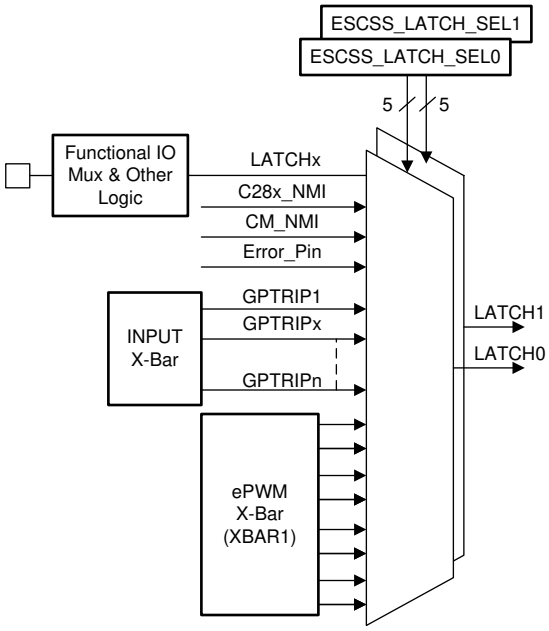


Figure 31-17. ESC Latch Input Integration

The details of connections and mux select to these muxes is shown in Table 31-12 and selection is possible individually for each of the LATCH0 or LATCH1 signal.

Table 31-12. ESC LATCH0/1 Trigger Table

ESCSS_LATCH_SELx	Signal Hookup
0	ESCSS_LATCH0 (From Pin)
1	ESCSS_LATCH1 (From Pin)
2	CPU1NMI
3	CMNMI
4	ERRORSTS
5	GPTRIPOUT0
6	GPTRIPOUT1
7	GPTRIPOUT2
8	GPTRIPOUT3
9	GPTRIPOUT4
10	GPTRIPOUT5
11	GPTRIPOUT6
12	GPTRIPOUT7
13	GPTRIPOUT8
14	GPTRIPOUT9
15	GPTRIPOUT10
16	GPTRIPOUT11
17	GPTRIPOUT12
18	GPTRIPOUT13
19	GPTRIPOUT14
20	GPTRIPOUT15
21	PWMXBAROUT0
22	PWMXBAROUT1
23	PWMXBAROUT2
24	PWMXBAROUT3
25	PWMXBAROUT4
26	PWMXBAROUT5
27	PWMXBAROUT6
28	PWMXBAROUT7
29	Reserved
30	Reserved
31	Reserved

31.2.10.3.1 Timestamping

Timestamping allows the local application or remote EtherCAT master to measure the time events and plan the subsequent controls.

Timestamping: Device Internal Events

- Enable timestamping and logging of internal device events coming from control logic
- The same outputs of the EPWM crossbar that connect to the 8 EPWMs can be used as 8 input signals (PWMXBAROUT0-PWMXBAROUT7) for time stamping
- Note that there is no independent option for Latch muxing as far as EPWM cross-bar inputs are concerned; unless one of the TRIP outputs are not used in control loop and such a crossbar output is dedicated for LATCH0/1 toggle

Timestamping: External Events

- Enable external board components apart from the MCU to trigger timestamp capture
- For example, timestamp a periodic event or pulse/edge event from a senor when the sensor data is read or accessed
- Connecting the LATCH0/1 controls through the input crossbar allows timestamp capture based on any selected GPIO toggle (such as GPTRIP15 or GPTRIP16)
  - Two GPTRIPs are provided for independent GPIO choice for LATCH0 and LATCH1
  - GPTRIP1, 2, and 3 allow the same functionality; however, those can be utilized for EPWM trip zone functions where pins can or cannot need to match. In the case of a trip zone based on GPTRIP is to be latched, one of these inputs can be used.

Timestamping: Device Exception Events

- Log and timestamp exception events within the device (example: NMI exceptions) and periodically collect information to find out systemic issues in the system
- Can be used by local application and the remote master device to diagnose or debug the system
- On CPU1, ERAD can be used to analyze particular accesses or data patterns/counts on the CPU bus. Refer to the *Embedded Real-time Analysis and Diagnostic (ERAD)* chapter for further information

31.2.10.4 Device Control and Synchronization

The primary advantage of DC functionality is when used in conjunction with device control functions. The precisely timed pulse/edge nature of the SYNC allows the ESC to synchronize the internal resources like PWMs, ECAPs (for capture as well as PWM) with remote system components. The following are the connections which are available for applications to program for allowing this synchronization.

31.2.10.4.1 Synchronization of PWM

The PWM Sync-chain on the MCU can be triggered either by the device external inputs from the GPIO via input cross-bar or the PWM internal events when the chain is programmed appropriately. The SYNC0/1 inputs in this implementation also act as the external sync inputs to the PWM sync-chain. The integration is shown in Figure 31-18.

The additional select register added to the trigger crossbar register allows independent programming of the EXTSYNCx from either GPTRIP or SYNC0/1.

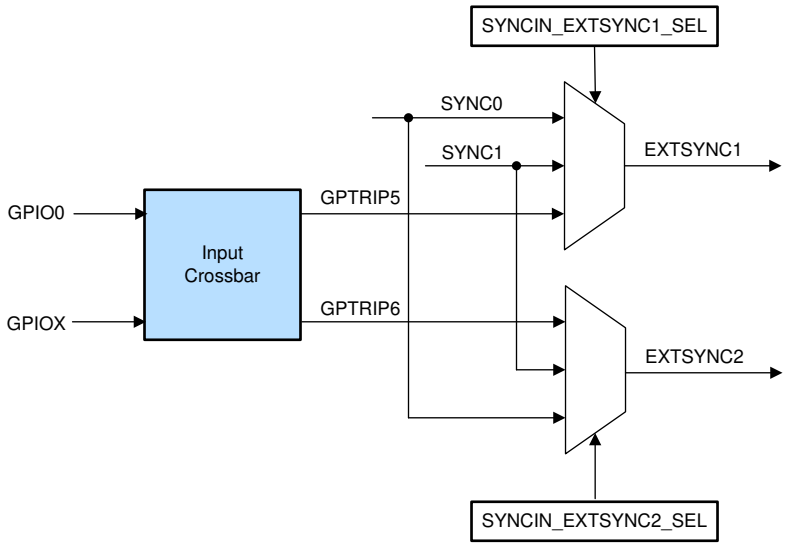


Figure 31-18. SYNC Integration for Control Functions - PWM SYNC

31.2.10.4.2 ECAP SYNC Inputs

ECAP supports accurate time capture of events that can be relatively checked against the series of events in vicinity and used for the control loop action. Additionally, ECAP has inbuilt capability of limited PWM action that can be triggered based on the selected input. Hence, the SYNC0/1 are also connected to ECAP input mux. The exact select value/vector for SYNC0/1 is defined in the *Enhanced Capture (eCAP)* chapter.

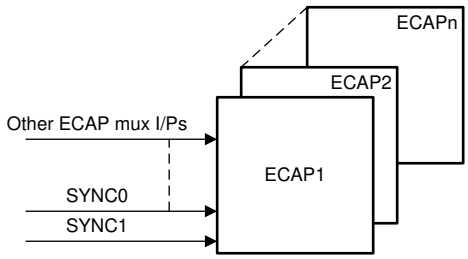


Figure 31-19. SYNC Integration for Control Functions – ECAP

31.2.10.4.3 SYNC Signal Conditioning and Rerouting

SYNC0/SYNC1 generation modes can generate various waveform patterns to create different trigger conditions. The SYNC0/1 signals are routed to the CLB, which can make conditional and delay based waveforms. Using the CLB, these signals can be processed for additional uses based on the application.

Additionally, this integration also allows routing SYNC0/1 signals with or without processing to other destinations within the device that are not explicitly connected. Details of the CLB input mux selects are explained in the *Configurable Logic Block (CLB)* chapter.

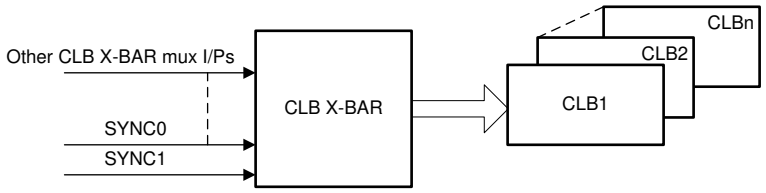


Figure 31-20. SYNC Integration for Signal Conditioning – CLB

31.3 Software Initialization Sequence and Allocating Ownership

This section details the software initialization sequence when configuring CPU1 or CM as ESC owner. The CM sequence includes details on allocating ownership of the ESC peripheral.

Table 31-13. CPU1 Software Initialization Sequence

Step	Action
1	General device initialization (configure clock, enable PLL, enable peripheral clocks except EtherCAT)
2	Configure Aux Clock for EtherCAT (if using Aux clock as source)
3	Configure GPIOs for EtherCAT (set pin configurations, set GPIO qualification mode, set pad configuration)
4	Initialize interrupts and register ISR handlers
5	Set EtherCAT clock source and divider. Then configure if EtherCAT PHY is clocked from device or external PHY clock.
6	Configure the EEPROM size
7	Bring ESC out of reset using system control register
8	Perform EtherCAT memory initialization and wait until memory initialization is complete
9	(Optional) Enable debug access to the EtherCAT registers
10	(Optional) Check that EEPROM loaded successfully
11	EtherCAT subsystem configurations for interrupt masking, SYNCx connections, and so on(1)

(1) Applications must make sure that ESC outputs are in a safe state until the EEPROM is loaded and that SYNC and LATCH are configured only after the EEPROM is loaded.

Table 31-14. CM Software Initialization Sequence

Step	Core	Action
1	CPU1	General device initialization (configure clock, enable PLL, enable peripheral clocks except EtherCAT)
2	CPU1	Configure GPIOs for EtherCAT (set pin configurations, set GPIO qualification mode, set pad configuration)
3	CPU1	Allocate ESC to CM
4	CPU1	Configure CM clocks and release CM from reset to wait mode
5	CPU1	Configure Aux Clock for EtherCAT (if using Aux clock as source)
6	CPU1	Set EtherCAT clock source and divider. Then configure if EtherCAT PHY is clocked from device or external PHY clock.
7	CPU1	Set CM boot mode and boot CM to start application
8	CPU1	General Device Initialization
9	CM	Initialize interrupts and register ISR handlers
10	CM	Configure EtherCAT EEPROM size
11	CM	Bring ESC out of reset using system control register
12	CM	Perform EtherCAT memory initialization and wait until memory initialization is complete
13	CM	(Optional) Enable debug access to the EtherCAT registers
14	CM	(Optional) Check that EEPROM loaded successfully
15	CM	EtherCAT subsystem configurations for interrupt masking, SYNCx connections, and so on(1)

(1) Applications must make sure that ESC outputs are in a safe state until the EEPROM is loaded and that SYNC and LATCH are configured only after the EEPROM is loaded.

31.4 ESC Configuration Constants

The following is EtherCAT IP core configurations that are static by design.

Table 31-15. ESC Configuration Constants Table

Name	Value
ESC Type	0x91
Revision	0x0
Build	0x0
FMMU Supported	0x8
SyncManagers	0x8
RAM Size	0x10
Port Descriptor	0xF
ESC Features Supported	0x1CC
Product ID	0x0
Vendor ID	0x0

Table 31-16. ESC IP Register Constants Table

EtherCAT IP Register	Offset Address	Value(1)
PDI 8/16Bit asynchronous Microcontroller configuration	0x0150	0x00
Sync/Latch PDI configuration	0x0151	0x66
Pulse Length of SyncSignals	0x0982/0x0983	0x000A

(1) These values will not be changed or overwritten by EEPROM contents.

31.5 EtherCAT IP Registers

The EtherCAT IP (IP core) register details are included as part of Beckhoff documentation.

Table 31-17. EtherCAT IP Register Documentation

Document	Beckhoff ESC Type	Link
EtherCAT Slave Controller Hardware Data Sheet - Section II - Register Description (ethercat_esc_datasheet_sec2_registers)	EtherCAT IP Core	www.beckhoff.com

31.5.1 ECAT Base Address Table (C28)

Table 31-18. ECAT Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
EscssRegs	ESCSS_REGS	ESC_SS_BASE	0x0005_7E00	YES	-	-	-	YES
EscssConfigRegs	ESCSS_CONFIG_REGS	ESC_SS_CONFIG_BASE	0x0005_7F00	YES	-	-	-	YES

31.5.2 ESCSS_REGS Registers

Table 31-19 lists the memory-mapped registers for the ESCSS_REGS registers. All register offset addresses not listed in Table 31-19 should be considered as reserved locations and the register contents should not be modified.

Table 31-19. ESCSS_REGS Registers

Offset (x8)	Offset (x16)	Acronym	Register Name	Write Protection	Section
0h	0h	ESCSS_IPREVNUM	IP Revision Number		Go
4h	2h	ESCSS_INTR_RIS	EtherCATSS Interrupt Raw Status		Go
8h	4h	ESCSS_INTR_MASK	EtherCATSS Interrupt Mask		Go
Ch	6h	ESCSS_INTR_MIS	EtherCATSS Masked Interrupt Status		Go
10h	8h	ESCSS_INTR_CLR	EtherCATSS Interrupt Clear		Go
14h	Ah	ESCSS_INTR_SET	EtherCATSS Interrupt Set to emulate		Go
18h	Ch	ESCSS_LATCH_SEL	Select for Latch0/1 inputs and LATCHIN input		Go
1Ch	Eh	ESCSS_ACCESS_CTRL	PDI interface access control config.		Go
20h	10h	ESCSS_GPIN_DAT	GPIN data capture for debug & override		Go
24h	12h	ESCSS_GPIN_PIPE	GPIN pipeline select		Go
28h	14h	ESCSS_GPIN_GRP_CAP_SEL	GPIN pipe group capture trigger		Go
2Ch	16h	ESCSS_GPOUT_DAT	GPOUT data capture for debug & override		Go
30h	18h	ESCSS_GPOUT_PIPE	GPOUT pipeline select		Go
34h	1Ah	ESCSS_GPOUT_GRP_CAP_SEL	GPOUT pipe group capture trigger		Go
38h	1Ch	ESCSS_MEM_TEST	Memory Test Control		Go
3Ch	1Eh	ESCSS_RESET_DEST_CONFIG	ResetOut impact or destination config	LOCK	Go
40h	20h	ESCSS_SYNC0_CONFIG	SYNC0 Configuration for various triggers	LOCK	Go
44h	22h	ESCSS_SYNC1_CONFIG	SYNC1 Configuration for various triggers	LOCK	Go

Complex bit access types are encoded to fit into small table cells. Table 31-20 shows the codes that are used for access types in this section.

Table 31-20. ESCSS_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 31-20. ESCSS_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

31.5.2.1 ESCSS_IPREVNUM Register (Offset (x8) = 0h, Offset (x16) = 0h) [Reset = 0h]

ESCSS_IPREVNUM is shown in Figure 31-21 and described in Table 31-21.

Return to the Summary Table.

IP Revision number showing the Major & Minor IP versions 4 bit each

Figure 31-21. ESCSS_IPREVNUM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								IP_REV_MAJOR				IP_REV_MINOR			
R-0-0h								R-0h				R-0h			

Table 31-21. ESCSS_IPREVNUM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R-0	0h	Reserved
7-4	IP_REV_MAJOR	R	0h	Major IP Type increment is hardcoded reset value which increments to signify major change in IP behavior in terms of data/control flow or new feature addition. Reset type: ECAT.IPRSn
3-0	IP_REV_MINOR	R	0h	Reset value for this register is hardcoded and increments with minor changes to the IP those will not increment IP Type, but the bug fixes and changes impact behavior or software control than previous silicon version. Reset type: ECAT.IPRSn

31.5.2.2 ESCSS_INTR_RIS Register (Offset (x8) = 4h, Offset (x16) = 2h) [Reset = 0h]

ESCSS_INTR_RIS is shown in Figure 31-22 and described in Table 31-22.

Return to the Summary Table.

Registers the Raw Interrupt status of different interrupt triggers regardless of mask.

Figure 31-22. ESCSS_INTR_RIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		MASTER_RESET_RIS	TIMEOUT_ERR_RIS	DMA_DONE_RIS	IRQ_RIS	SYNC1_RIS	SYNC0_RIS
R-0-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 31-22. ESCSS_INTR_RIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5	MASTER_RESET_RIS	R	0h	Indicates Raw Status of the EtherCAT Master Reset event , until cleared by ESCSS_INTR_CLR 0: EtherCAT Master Reset Event did not happen since last IP reset or last clear of this bit and ECAT Master reset programmed to be Interrupt to local host. 1: EtherCAT Master Reset Event has occured. This status gets updated regardless of interrupt mask, it is registered on the EtherCATSS clock domain, sticky and remains active till cleared using ESCSS_INTR_CLR. If simultaneous clear & incoming event on same clock edge the incoming event registering has priority and clear does not have effect. Information of this event is lost if ECAT Master event is programmed to be reset IP. Reset type: ECAT.IPRSn
4	TIMEOUT_ERR_RIS	R	0h	Indicates Raw Status of the past event on PDI access timeout Error, until cleared by ESCSS_INTR_CLR 0: PDI Access Timeout Error Event did not happen since reset or last clear of this bit. 1: PDI Access Timeout Error Event has triggered the interrupt/DMA trigger. This status gets updated regardless of interrupt mask as long as Timeout is enabled, it is registered on the EtherCATSS clock domain, sticky and remains active till cleared using ESCSS_INTR_CLR. If simultaneous clear & incoming event on same clock edge the incoming event registering has priority and clear does not have effect. Reset type: ECAT.IPRSn

Table 31-22. ESCSS_INTR_RIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	DMA_DONE_RIS	R	0h	Indicates Raw Status of the past event on DMA Done, until cleared by ESCSS_INTR_CLR 0: DMA Done Event did not happen since reset or last clear of this bit. 1: DMA Done Event has triggered the interrupt/DMA trigger. This status gets updated regardless of interrupt mask, it is registered on the EtherCATSS clock domain, sticky and remains active till cleared using ESCSS_INTR_CLR. If simultaneous clear & incoming event on same clock edge the incoming event registering has priority and clear does not have effect. Reset type: ECAT.IPRSn
2	IRQ_RIS	R	0h	Indicates Raw Status of the past event on EtherCATSS IRQ, until cleared by ESCSS_INTR_CLR 0: EtherCATSS IRQ Event did not happen since reset or last clear of this bit. 1: EtherCATSS IRQ Event has triggered the interrupt/DMA trigger. This status gets updated regardless of interrupt mask, it is registered on the EtherCATSS clock domain, sticky and remains active till cleared using ESCSS_INTR_CLR. If simultaneous clear & incoming event on same clock edge the incoming event registering has priority and clear does not have effect. Reset type: ECAT.IPRSn
1	SYNC1_RIS	R	0h	Indicates Raw Status of the past event on SYNC1, until cleared by ESCSS_INTR_CLR 0: SYNC1 Event did not happen since reset or last clear of this bit. 1: SYNC1 Event has triggered the interrupt/DMA trigger. This status gets updated regardless of interrupt mask, it is registered on the EtherCATSS clock domain, sticky and remains active till cleared using ESCSS_INTR_CLR. If simultaneous clear & incoming event on same clock edge the incoming event registering has priority and clear does not have effect. Reset type: ECAT.IPRSn
0	SYNC0_RIS	R	0h	Indicates Raw Status of the past event on SYNC0, until cleared by ESCSS_INTR_CLR 0: SYNC0 Event did not happen since reset or last clear of this bit. 1: SYNC0 Event has triggered the interrupt/DMA trigger. This status gets updated regardless of interrupt mask, it is registered on the EtherCATSS clock domain, sticky and remains active till cleared using ESCSS_INTR_CLR. If simultaneous clear & incoming event on same clock edge the incoming event registering has priority and clear does not have effect. Reset type: ECAT.IPRSn

31.5.2.3 ESCSS_INTR_MASK Register (Offset (x8) = 8h, Offset (x16) = 4h) [Reset = 0h]

ESCSS_INTR_MASK is shown in Figure 31-23 and described in Table 31-23.

Return to the Summary Table.

Allows to mask individual interrupt cause impacting the interrupt

Figure 31-23. ESCSS_INTR_MASK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		MASTER_RESET_MASK	TIMEOUT_ERR_MASK	DMA_DONE_MASK	IRQ_MASK	SYNC1_MASK	SYNC0_MASK
R-0-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 31-23. ESCSS_INTR_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5	MASTER_RESET_MASK	R/W	0h	Masks EtherCAT Master reset event against any effect on interrupts or other CPU Interrupts. 1: EtherCAT Master Reset affects the interrupt/DMA trigger. 0: EtherCAT Master Reset masked and does not affect Interrupt. Reset type: ECAT.IPRSn
4	TIMEOUT_ERR_MASK	R/W	0h	Masks PDI access timeout Error to have effect on interrupts or other CPU Interrupts. 1: PDI Access Timeout Errors affects the interrupt/DMA trigger. 0: PDI Access Timeout Errors masked and does not affect Interrupt. Reset type: ECAT.IPRSn
3	DMA_DONE_MASK	R/W	0h	Masks DMA Done status update to have effect on interrupts or other CPU Interrupts. 1: DMA Done affects the interrupt/DMA trigger. 0: DMA Done masked and does not affect Interrupt. Raw DMA Done status is updated regardless of this setting. Reset type: ECAT.IPRSn
2	IRQ_MASK	R/W	0h	Masks EtherCATSS IRQ to have effect on interrupts or other CPU/DMA triggers. 1: EtherCATSS IRQ affects the interrupt/DMA trigger. 0: EtherCATSS IRQ masked and does not affect Interrupt/DMA trigger. Raw EtherCATSS IRQ status is updated regardless of this setting. Reset type: ECAT.IPRSn
1	SYNC1_MASK	R/W	0h	Masks SYNC1 to have effect on interrupts or other CPU/DMA triggers as programmed in HOST_TRIG_MAP registers. 1: SYNC1 affects the interrupt/DMA trigger. 0: SYNC1 masked and does not affect Interrupt/DMA trigger. Raw SYNC1 status is updated regardless of this setting. Reset type: ECAT.IPRSn

Table 31-23. ESCSS_INTR_MASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	SYNC0_MASK	R/W	0h	Masks SYNC0 to have effect on interrupts or other CPU/DMA triggers as programmed in HOST_TRIG_MAP registers. 1: SYNC0 affects the interrupt/DMA trigger. 0: SYNC0 masked and does not affect Interrupt/DMA trigger. Raw SYNC0 status is updated regardless of this setting. Reset type: ECAT.IPRSn

31.5.2.4 ESCSS_INTR_MIS Register (Offset (x8) = Ch, Offset (x16) = 6h) [Reset = 0h]

ESCSS_INTR_MIS is shown in Figure 31-24 and described in Table 31-24.

Return to the Summary Table.

Registers the Msked Interrupt status of different interrupt triggers. This is AND of RIS & MASK of respective fields

Figure 31-24. ESCSS_INTR_MIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		MASTER_RESET_MIS	TIMEOUT_ERR_MIS	DMA_DONE_MIS	IRQ_MIS	SYNC1_MIS	SYNC0_MIS
R-0-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 31-24. ESCSS_INTR_MIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5	MASTER_RESET_MIS	R	0h	Indicates Masked Interrupt status of the past event on EtherCAT Master Reset, until RIS cleared by ESCSS_INTR_CLR or by Reset. 0: No pending EtherCAT Master Reset interrupt, if configured and unmasked. 1: EtherCAT Master Reset Event has triggered the interrupt/DMA trigger and it's pending. This is MASK qualified RIS such that application can select to service or suppress the interrupt cause behind this. Upon reset or upon RIS clear through ESCSS_INTR_CLR this field is cleared. Reset type: ECAT.IPRSn
4	TIMEOUT_ERR_MIS	R	0h	Indicates Masked Interrupt status of the past event on PDI Access Timeout Error, until RIS cleared by ESCSS_INTR_CLR 0: No pending PDI Access Timeout Error interrupt. 1: PDI Access Timeout Error Event has triggered the interrupt/DMA trigger and it's pending. This is MASK qualified RIS such that application can select to service or suppress the interrupt cause behind this. Upon reset or upon RIS clear through ESCSS_INTR_CLR this field is cleared. Reset type: ECAT.IPRSn
3	DMA_DONE_MIS	R	0h	Indicates Masked Interrupt status of the past event on DMA Done, until RIS is cleared by ESCSS_INTR_CLR 0: No pending DMA Done interrupt. 1: DMA Done Event has triggered the interrupt/DMA trigger and it's pending. This is MASK qualified RIS such that application can select to service or suppress the interrupt cause behind this. Upon reset or upon RIS clear through ESCSS_INTR_CLR this field is cleared. Reset type: ECAT.IPRSn

Table 31-24. ESCSS_INTR_MIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	IRQ_MIS	R	0h	Indicates Masked Interrupt status of the past event on EtherCATSS IRQ, until RIS is cleared by ESCSS_INTR_CLR 0: No pending EtherCATSS IRQ interrupt. 1: EtherCATSS IRQ Event has triggered the interrupt and it's pending. This is MASK qualified RIS such that application can select to service or suppress the interrupt cause behind this. Upon reset or upon RIS clear through ESCSS_INTR_CLR this field is cleared. Reset type: ECAT.IPRSn
1	SYNC1_MIS	R	0h	Indicates Masked Interrupt status of the past event on SYNC0, until RIS is cleared by ESCSS_INTR_CLR 0: No pending SYNC1 interrupt. 1: SYNC1 Event has triggered the interrupt/DMA trigger and it's pending. This is MASK qualified RIS such that application can select to service or suppress the interrupt cause behind this. Upon reset or upon RIS clear through ESCSS_INTR_CLR this field is cleared. Reset type: ECAT.IPRSn
0	SYNC0_MIS	R	0h	Indicates Masked Interrupt status of the past event on SYNC0, until RIS is cleared by ESCSS_INTR_CLR 0: No pending SYNC0 interrupt. 1: SYNC0 Event has triggered the interrupt/DMA trigger and it's pending. This is MASK qualified RIS such that application can select to service or suppress the interrupt cause behind this. Upon reset or upon RIS clear through ESCSS_INTR_CLR this field is cleared. Reset type: ECAT.IPRSn

31.5.2.5 ESCSS_INTR_CLR Register (Offset (x8) = 10h, Offset (x16) = 8h) [Reset = 0h]

ESCSS_INTR_CLR is shown in Figure 31-25 and described in Table 31-25.

Return to the Summary Table.

Individual Interrupt cause clear register

Figure 31-25. ESCSS_INTR_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED		MASTER_RESET_CLR	TIMEOUT_ERR_CLR	DMA_DONE_CLR	IRQ_CLR	SYNC1_CLR	SYNC0_CLR
R-0-0h		R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h

Table 31-25. ESCSS_INTR_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R-0	0h	Reserved
5	MASTER_RESET_CLR	R-0/W1C	0h	Clears EtherCAT Master Reset raw Status (Common Mask & Clear among all hosts, only one host expected accessing). Write 1 clears the Raw status of the EtherCAT Master reset, read always returns 0. Reset type: ECAT.IPRSn
4	TIMEOUT_ERR_CLR	R-0/W1C	0h	Clears PDI access timeout Error raw Status (Common Mask & Clear among all hosts, only one host expected accessing). Write 1 clears the Raw status of the PDI Access Timeout Error, read always returns 0. Reset type: ECAT.IPRSn
3	DMA_DONE_CLR	R-0/W1C	0h	Clears DMA Done raw Status (Common Mask & Clear among all hosts, only one host expected accessing). Write 1 clears the Raw status of the DMA Done, read always returns 0. Reset type: ECAT.IPRSn
2	IRQ_CLR	R-0/W1C	0h	Clears EtherCATSS IRQ raw Status (Common Mask & Clear among all hosts, only one host expected accessing). Write 1 clears the Raw status of the EtherCATSS IRQ, read always returns 0. Reset type: ECAT.IPRSn
1	SYNC1_CLR	R-0/W1C	0h	Clears SYNC1 raw Status (Common Mask & Clear among all hosts, only one host expected accessing). Write 1 clears the Raw status of the SYNC1, read always returns 0. Reset type: ECAT.IPRSn
0	SYNC0_CLR	R-0/W1C	0h	Clears SYNC0 raw Status (Common Mask & Clear among all hosts, only one host expected accessing). Write 1 clears the Raw status of the SYNC0, read always returns 0. Reset type: ECAT.IPRSn

31.5.2.6 ESCSS_INTR_SET Register (Offset (x8) = 14h, Offset (x16) = Ah) [Reset = 0h]

ESCSS_INTR_SET is shown in Figure 31-26 and described in Table 31-26.

Return to the Summary Table.

Individual Interrupt cause set register to emulate the interrupt cause

Figure 31-26. ESCSS_INTR_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
WRITE_KEY							
R-0/W-0h							
7	6	5	4	3	2	1	0
RESERVED		MASTER_RESET_SET	TIMEOUT_ERR_SET	DMA_DONE_SET	IRQ_SET	SYNC1_SET	SYNC0_SET
R-0-0h		R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 31-26. ESCSS_INTR_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	WRITE_KEY	R-0/W	0h	The key value should be 0xa5 for the writes to Set bits to take effect. Writes of other values will be ignored. Reset type: ECAT.IPRSn
7-6	RESERVED	R-0	0h	Reserved
5	MASTER_RESET_SET	R-0/W1S	0h	Sets EtherCAT Master Reset raw Status (Common Mask & Clear among all hosts, only one host expected accessing). Write 1 Sets the Raw status of the PDI Access Timeout Error, read always returns 0. Note this emulation can only assert interrupt to CPU but it can not reset the EtherCAT IP. Reset type: ECAT.IPRSn
4	TIMEOUT_ERR_SET	R-0/W1S	0h	Sets PDI access timeout Error raw Status (Common Mask & Clear among all hosts, only one host expected accessing). Write 1 Sets the Raw status of the PDI Access Timeout Error, read always returns 0. Reset type: ECAT.IPRSn
3	DMA_DONE_SET	R-0/W1S	0h	Sets DMA Done raw Status (Common Mask & Clear among all hosts, only one host expected accessing). Write 1 Sets the Raw status of the DMA Done, read always returns 0. Reset type: ECAT.IPRSn
2	IRQ_SET	R-0/W1S	0h	Sets EtherCATSS IRQ raw Status (Common Mask & Clear among all hosts, only one host expected accessing). Write 1 sets the Raw status of the EtherCATSS IRQ, read always returns 0. Reset type: ECAT.IPRSn
1	SYNC1_SET	R-0/W1S	0h	Sets SYNC1 raw Status (Common Mask & Clear among all hosts, only one host expected accessing). Write 1 Sets the Raw status of the SYNC1, read always returns 0. Reset type: ECAT.IPRSn

Table 31-26. ESCSS_INTR_SET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	SYNC0_SET	R-0/W1S	0h	Sets SYNC0 raw Status (Common Mask & Clear among all hosts, only one host expected accessing). Write 1 sets the Raw status of the SYNC0, read always returns 0. Reset type: ECAT.IPRSn

31.5.2.7 ESCSS_LATCH_SEL Register (Offset (x8) = 18h, Offset (x16) = Ch) [Reset = 0h]

ESCSS_LATCH_SEL is shown in Figure 31-27 and described in Table 31-27.

Return to the Summary Table.

Select for LATCH0/1 input Triggers as well as LATCHIN used for registering the GPIs.

Figure 31-27. ESCSS_LATCH_SEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							RESERVED
R-0-0h							R/W-0h
15	14	13	12	11	10	9	8
RESERVED			LATCH1_SELECT				
R-0-0h			R/W-0h				
7	6	5	4	3	2	1	0
RESERVED			LATCH0_SELECT				
R-0-0h			R/W-0h				

Table 31-27. ESCSS_LATCH_SEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R-0	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15-13	RESERVED	R-0	0h	Reserved
12-8	LATCH1_SELECT	R/W	0h	Mux Select for LATCH1 input to ECATSS. Refer device specification for details of mux select options. Reset type: ECAT.IPRSn
7-5	RESERVED	R-0	0h	Reserved
4-0	LATCH0_SELECT	R/W	0h	Mux Select for LATCH0 input to ECATSS. Refer device specification for details of mux select options. Reset type: ECAT.IPRSn

31.5.2.8 ESCSS_ACCESS_CTRL Register (Offset (x8) = 1Ch, Offset (x16) = Eh) [Reset = 400h]

ESCSS_ACCESS_CTRL is shown in Figure 31-28 and described in Table 31-28.

Return to the Summary Table.

Wait state control for EtherCAT access

Figure 31-28. ESCSS_ACCESS_CTRL Register

31	30	29	28	27	26	25	24
RESERVED				TIMEOUT_COUNT			
R-0-0h				R/W-0h			
23	22	21	20	19	18	17	16
TIMEOUT_COUNT							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED					ENABLE_PARALLEL_PORT_ACCESS	ENABLE_DEBUG_ACCESS	RESERVED
R-0-0h					R/W-1h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
EN_TIMEOUT	WAIT_STATES						
R/W-0h	R/W-0h						

Table 31-28. ESCSS_ACCESS_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R-0	0h	Reserved
27-16	TIMEOUT_COUNT	R/W	0h	This is the cycle count in SYSCLK cycles after which an access on the EtherCAT Async interface will be aborted and CPU will be given back a READY. Reset type: ECAT.IPRSn
15-11	RESERVED	R-0	0h	Reserved
10	ENABLE_PARALLEL_PORT_ACCESS	R/W	1h	Enabled memory accesses through the parallel port interface. 0: Memory accesses using parallel port are not allowed to go through. 1: Memory accesses using parallel port are allowed through the Bridge. Reset type: ECAT.IPRSn
9	ENABLE_DEBUG_ACCESS	R/W	0h	Enabled debug accesses through the PDI interface. 0: Debug accesses are not allowed to go through. 1: Debug accesses are allowed through the Bridge. Bridge logic will ensure that access will not hang. Reset type: ECAT.IPRSn
8	RESERVED	R/W	0h	Reserved
7	EN_TIMEOUT	R/W	0h	Enables the Timeout features which counts programmed number of Sys clocks before the Local host aborts the transaction. 0: Timeout feature is not enabled on PDI interface. 1: The timeout counter starts counting upon BUSY is asserted by EtherCAT IP. Reset type: ECAT.IPRSn
6-0	WAIT_STATES	R/W	0h	This is the predefined minimum number of wait-states which the VBUS bridge will put out accesses on the 16-bit Async interface. Reset type: ECAT.IPRSn

31.5.2.9 ESCSS_GPIN_DAT Register (Offset (x8) = 20h, Offset (x16) = 10h) [Reset = 0h]

ESCSS_GPIN_DAT is shown in Figure 31-29 and described in Table 31-29.

Return to the Summary Table.

GPI data status for debug & overridwe

Figure 31-29. ESCSS_GPIN_DAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIN_DAT																															
R/W-0h																															

Table 31-29. ESCSS_GPIN_DAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPIN_DAT	R/W	0h	Local GPIN data register connects to GPIN pipelined register for debug & override purposes. Note: The copy of this register readable by the CPU is provided without synchronization, therefore multiple reads should be performed to confirm a stable value before using it. Reset type: ECAT.IPRSn

31.5.2.10 ESCSS_GPIN_PIPE Register (Offset (x8) = 24h, Offset (x16) = 12h) [Reset = 0h]

ESCSS_GPIN_PIPE is shown in Figure 31-30 and described in Table 31-30.

Return to the Summary Table.

Register to select raw PAD input or pipelined input be presented to ESC.

Figure 31-30. ESCSS_GPIN_PIPE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPI_PIPE																															
R/W-0h																															

Table 31-30. ESCSS_GPIN_PIPE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPI_PIPE	R/W	0h	Enables the connection of GPIN to EtherCATSS through pipelined register as against the direct from IO. 0: The connection is directly from the IO pad 1: Connection is through the pipelined register which is captured on programmed event. Reset type: ECAT.IPRSn

31.5.2.11 ESCSS_GPIN_GRP_CAP_SEL Register (Offset (x8) = 28h, Offset (x16) = 14h) [Reset = 0h]

ESCSS_GPIN_GRP_CAP_SEL is shown in Figure 31-31 and described in Table 31-31.

Return to the Summary Table.

Register to configure trigger select for the group of 8 IOs together.

Figure 31-31. ESCSS_GPIN_GRP_CAP_SEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED	GPI_GRP_CAP_SEL3			RESERVED	GPI_GRP_CAP_SEL2		
R-0-0h	R/W-0h			R-0-0h	R/W-0h		
7	6	5	4	3	2	1	0
RESERVED	GPI_GRP_CAP_SEL1			RESERVED	GPI_GRP_CAP_SEL0		
R-0-0h	R/W-0h			R-0-0h	R/W-0h		

Table 31-31. ESCSS_GPIN_GRP_CAP_SEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R-0	0h	Reserved
14-12	GPI_GRP_CAP_SEL3	R/W	0h	Selects the trigger to capture the IO input in pipeline register for GPI31-24 0: Start Of Frame as capture trigger 1-3: Reserved, selects default value of 0. 4: SYNC0 as capture trigger 5: SYNC1 as capture trigger 6: LATCH0 as capture trigger 7: LATCH1 as capture trigger Reset type: ECAT.IPRSn
11	RESERVED	R-0	0h	Reserved
10-8	GPI_GRP_CAP_SEL2	R/W	0h	Selects the trigger to capture the IO input in pipeline register for GPI23-16 0: Start Of Frame as capture trigger 1-3: Reserved, selects default value of 0. 4: SYNC0 as capture trigger 5: SYNC1 as capture trigger 6: LATCH0 as capture trigger 7: LATCH1 as capture trigger Reset type: ECAT.IPRSn
7	RESERVED	R-0	0h	Reserved
6-4	GPI_GRP_CAP_SEL1	R/W	0h	Selects the trigger to capture the IO input in pipeline register for GPI15-8 0: Start Of Frame as capture trigger 1-3: Reserved, selects default value of 0. 4: SYNC0 as capture trigger 5: SYNC1 as capture trigger 6: LATCH0 as capture trigger 7: LATCH1 as capture trigger Reset type: ECAT.IPRSn
3	RESERVED	R-0	0h	Reserved

Table 31-31. ESCSS_GPIN_GRP_CAP_SEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	GPI_GRP_CAP_SEL0	R/W	0h	Selects the trigger to capture the IO input in pipeline register for GPI7-0 0: Start Of Frame as capture trigger 1-3: Reserved, selects default value of 0. 4: SYNC0 as capture trigger 5: SYNC1 as capture trigger 6: LATCH0 as capture trigger 7: LATCH1 as capture trigger Reset type: ECAT.IPRSn

31.5.2.12 ESCSS_GPOUT_DAT Register (Offset (x8) = 2Ch, Offset (x16) = 16h) [Reset = 0h]

ESCSS_GPOUT_DAT is shown in Figure 31-32 and described in Table 31-32.

Return to the Summary Table.

GPO data capture for debug & overridwe

Figure 31-32. ESCSS_GPOUT_DAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPOUT_DAT																															
R-0h																															

Table 31-32. ESCSS_GPOUT_DAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPOUT_DAT	R	0h	Local GPOUT data register which is synchronised version on SysClk, each bit is represents GPO IO Read is allowed for CPU to process (IO extender or so if required). Reset type: ECAT.IPRSn

31.5.2.13 ESCSS_GPOUT_PIPE Register (Offset (x8) = 30h, Offset (x16) = 18h) [Reset = 0h]

ESCSS_GPOUT_PIPE is shown in Figure 31-33 and described in Table 31-33.

Return to the Summary Table.

Register to select pipeline of ESC output against direct route to IO pad on per IO based.

Figure 31-33. ESCSS_GPOUT_PIPE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPO_PIPE																															
R/W-0h																															

Table 31-33. ESCSS_GPOUT_PIPE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPO_PIPE	R/W	0h	Enables the connection of EtherCATSS GPO output to the IO pad through pipelined register as against the direct connection. 0: The connection is directly to the IO pad 1: Connection is through the pipelined register which captures EtherCATSS o/p on programmed event. Reset type: ECAT.IPRSn

31.5.2.14 ESCSS_GPOUT_GRP_CAP_SEL Register (Offset (x8) = 34h, Offset (x16) = 1Ah) [Reset = 0h]

ESCSS_GPOUT_GRP_CAP_SEL is shown in Figure 31-34 and described in Table 31-34.

Return to the Summary Table.

Register to configure trigger select for pipelined register in group of 8 IOs together.

Figure 31-34. ESCSS_GPOUT_GRP_CAP_SEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED		GPO_GRP_CAP_SEL3		RESERVED		GPO_GRP_CAP_SEL2	
R-0-0h		R/W-0h		R-0-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		GPO_GRP_CAP_SEL1		RESERVED		GPO_GRP_CAP_SEL0	
R-0-0h		R/W-0h		R-0-0h		R/W-0h	

Table 31-34. ESCSS_GPOUT_GRP_CAP_SEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R-0	0h	Reserved
13-12	GPO_GRP_CAP_SEL3	R/W	0h	Selects the trigger to capture the EtherCATSS output in pipeline register for GPO31-24 0: End Of Frame as capture trigger 1: SYNC0 as capture trigger 2: SYNC1 as capture trigger 3: WDTrig as capture trigger Reset type: ECAT.IPRSn
11-10	RESERVED	R-0	0h	Reserved
9-8	GPO_GRP_CAP_SEL2	R/W	0h	Selects the trigger to capture the EtherCATSS output in pipeline register for GPO23-16 0: End Of Frame as capture trigger 1: SYNC0 as capture trigger 2: SYNC1 as capture trigger 3: WDTrig as capture trigger Reset type: ECAT.IPRSn
7-6	RESERVED	R-0	0h	Reserved
5-4	GPO_GRP_CAP_SEL1	R/W	0h	Selects the trigger to capture the EtherCATSS output in pipeline register for GPO15-8 0: End Of Frame as capture trigger 1: SYNC0 as capture trigger 2: SYNC1 as capture trigger 3: WDTrig as capture trigger Reset type: ECAT.IPRSn
3-2	RESERVED	R-0	0h	Reserved
1-0	GPO_GRP_CAP_SEL0	R/W	0h	Selects the trigger to capture the EtherCATSS output in pipeline register for GPO7-0 0: End Of Frame as capture trigger 1: SYNC0 as capture trigger 2: SYNC1 as capture trigger 3: WDTrig as capture trigger Reset type: ECAT.IPRSn

31.5.2.15 ESCSS_MEM_TEST Register (Offset (x8) = 38h, Offset (x16) = 1Ch) [Reset = 0h]

ESCSS_MEM_TEST is shown in Figure 31-35 and described in Table 31-35.

Return to the Summary Table.

This register controls access to memory test mode

Figure 31-35. ESCSS_MEM_TEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0-0h							
7	6	5	4	3	2	1	0
RESERVED						MEM_INIT_DONE	INITIATE_MEM_INIT
R-0-0h						R-0h	R-0/W1S-0h

Table 31-35. ESCSS_MEM_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R-0	0h	Reserved
1	MEM_INIT_DONE	R	0h	Read-only status bit indicating memory initialisation completion. Gets self cleared with INITIATE_MEM_INIT is written '1'. Reset type: ECAT.IPRSn
0	INITIATE_MEM_INIT	R-0/W1S	0h	Memory Initialisation Trigger When set 1 the memory wrapper starts initialisation of DPRAM including parity programming. The bit gets Autocleared after memory initialisation starts. Write of 0 has no effect. Reset type: ECAT.IPRSn

31.5.2.16 ESCSS_RESET_DEST_CONFIG Register (Offset (x8) = 3Ch, Offset (x16) = 1Eh) [Reset = 0h]

ESCSS_RESET_DEST_CONFIG is shown in Figure 31-36 and described in Table 31-36.

Return to the Summary Table.

EtherCAT RESET_OUT configuration

Figure 31-36. ESCSS_RESET_DEST_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
WRITE_KEY							
R-0/W-0h							
7	6	5	4	3	2	1	0
DEVICE_RESET_EN	RESERVED				CPU_INT_EN	CPU_NMI_EN	CPU_RESET_EN
R/W-0h	R-0-0h				R/W-0h	R/W-0h	R/W-0h

Table 31-36. ESCSS_RESET_DEST_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	WRITE_KEY	R-0/W	0h	The key value should be 0xa5 for the writes to this register to take effect. Writes of other values will be ignored. Reset type: ECAT.IPRSn
7	DEVICE_RESET_EN	R/W	0h	Enable the EtherCAT RESET_OUT which is combination of IP Reset out and Pin reset to drive the Device Reset (XRSn) 0: EtherCAT RESET_OUT only drives the EtherCATSS & companion component reset to PHY 1: EtherCAT RESET_OUT drives EtherCATSS, external PHY reset and device by connecting this net on to device XRSn User's note: The connection from IP Resetout to EtherCAT RESET_OUT has no relation with this selection. Reset type: ECAT.XRSn
6-3	RESERVED	R-0	0h	Reserved
2	CPU_INT_EN	R/W	0h	Enable for resetout to drive the interrupt to CPU which it belongs to. 0: IP Resetout does not drive the CPU interrupt. 1: IP Resetout drives interrupt to the CPU master to which EtherCATSS belongs. The Host completes the reset through System control Soft reset after completing required context save or tasks if any. Reset type: ECAT.IPRSn
1	CPU_NMI_EN	R/W	0h	Enable for resetout to drive the CPU NMI 0: IP Resetout does not drive the CPU NMI. 1: IP Resetout drives CPU NMI to which it belongs. NMI handler is expected to complete the required taks or context save if any and then reset the EtherCAT through the system control soft reset. Reset type: ECAT.IPRSn

Table 31-36. ESCSS_RESET_DEST_CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	CPU_RESET_EN	R/W	0h	Enables EtherCAT Reset to drive the IP & PHY reset 0: EtherCAT Reset does not drive reset connection. 1: EtherCAT Reset drives EtherCAT IP and PHY Reset EtherCAT Reset Combines Master Reset, PDI sequence Reset, RESET_IN, System control soft Reset This selection is to drive the Master & PDI Reset to this combination. When this bit is set 0, application shall configure NMI/Interrupt to eventually complete the reset through system control soft reset. Reset type: ECAT.XRSn

31.5.2.17 ESCSS_SYNC0_CONFIG Register (Offset (x8) = 40h, Offset (x16) = 20h) [Reset = 0h]

ESCSS_SYNC0_CONFIG is shown in Figure 31-37 and described in Table 31-37.

Return to the Summary Table.

SYNC0 Triggers enable for Host events like Interrupts, DMA triggers across all masters & GPIO

Figure 31-37. ESCSS_SYNC0_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
WRITE_KEY							
R-0/W-0h							
7	6	5	4	3	2	1	0
RESERVED			uDMA_TRIG_EN	CM4_NVIC_EN	C28x_DMA_EN	CLA_INT_EN	C28x_PIE_EN
R-0-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 31-37. ESCSS_SYNC0_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	WRITE_KEY	R-0/W	0h	The key value should be 0xa5 for the writes to this register to take effect. Writes of other values will be ignored. Reset type: ECAT.IPRSn
7-5	RESERVED	R-0	0h	Reserved
4	uDMA_TRIG_EN	R/W	0h	Makes the connection from SYNC0 output to uDMA Trigger. 0: SYNC0 does not contribute to uDMA trigger. 1: SYNC0 toggle Triggers the uDMA Transfer. Reset type: ECAT.IPRSn
3	CM4_NVIC_EN	R/W	0h	Makes the connection from SYNC0 output to CM4 NVIC Interrupt. 0: SYNC0 does not contribute to CM4 NVIC regardless of mask. 1: SYNC0 follows the PIE interrupt behavior as controlled by RIS,MASK,CLR register pairs. Reset type: ECAT.IPRSn
2	C28x_DMA_EN	R/W	0h	Makes the connection from SYNC0 output to C28x DMA Trigger. 0: SYNC0 does not contribute to C28x DMA trigger. 1: SYNC0 toggle Triggers the C28x DMA Transfer. Reset type: ECAT.IPRSn
1	CLA_INT_EN	R/W	0h	Makes the connection from SYNC0 output to CLA Interrupt. 0: SYNC0 does not contribute to CLA Interrupt regardless of mask. 1: SYNC0 follows the CLA interrupt behavior as controlled by RIS,MASK,CLR register pairs. Reset type: ECAT.IPRSn
0	C28x_PIE_EN	R/W	0h	Makes the connection from SYNC0 output to C28x PIE Interrupt. 0: SYNC0 does not contribute to C28x PIE regardless of mask. 1: SYNC0 follows the PIE interrupt behavior as controlled by RIS,MASK,CLR register pairs. Reset type: ECAT.IPRSn

31.5.2.18 ESCSS_SYNC1_CONFIG Register (Offset (x8) = 44h, Offset (x16) = 22h) [Reset = 0h]

ESCSS_SYNC1_CONFIG is shown in Figure 31-38 and described in Table 31-38.

Return to the Summary Table.

SYNC1 Triggers enable for Host events like Interrupts, DMA triggers across all masters & GPIO

Figure 31-38. ESCSS_SYNC1_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
WRITE_KEY							
R-0/W-0h							
7	6	5	4	3	2	1	0
RESERVED			uDMA_TRIG_EN	CM4_NVIC_EN	C28x_DMA_EN	CLA_INT_EN	C28x_PIE_EN
R-0-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 31-38. ESCSS_SYNC1_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	WRITE_KEY	R-0/W	0h	The key value should be 0xa5 for the writes to this register to take effect. Writes of other values will be ignored. Reset type: ECAT.IPRSn
7-5	RESERVED	R-0	0h	Reserved
4	uDMA_TRIG_EN	R/W	0h	Makes the connection from SYNC1 output to uDMA Trigger. 0: SYNC1 does not contribute to uDMA trigger. 1: SYNC1 toggle Triggers the uDMA Transfer. Reset type: ECAT.IPRSn
3	CM4_NVIC_EN	R/W	0h	Makes the connection from SYNC1 output to CM4 NVIC Interrupt. 0: SYNC1 does not contribute to CM4 NVIC regardless of mask. 1: SYNC1 follows the PIE interrupt behavior as controlled by RIS,MASK,CLR register pairs. Reset type: ECAT.IPRSn
2	C28x_DMA_EN	R/W	0h	Makes the connection from SYNC1 output to C28x DMA Trigger. 0: SYNC1 does not contribute to C28x DMA trigger. 1: SYNC1 toggle Triggers the C28x DMA Transfer. Reset type: ECAT.IPRSn
1	CLA_INT_EN	R/W	0h	Makes the connection from SYNC1 output to CLA Interrupt. 0: SYNC1 does not contribute to CLA Interrupt regardless of mask. 1: SYNC1 follows the CLA interrupt behavior as controlled by RIS,MASK,CLR register pairs. Reset type: ECAT.IPRSn
0	C28x_PIE_EN	R/W	0h	Makes the connection from SYNC1 output to C28x PIE Interrupt. 0: SYNC1 does not contribute to C28x PIE regardless of mask. 1: SYNC1 follows the PIE interrupt behavior as controlled by RIS,MASK,CLR register pairs. Reset type: ECAT.IPRSn

31.5.3 ESCSS_CONFIG_REGS Registers

Table 31-39 lists the memory-mapped registers for the ESCSS_CONFIG_REGS registers. All register offset addresses not listed in Table 31-39 should be considered as reserved locations and the register contents should not be modified.

Table 31-39. ESCSS_CONFIG_REGS Registers

Offset (x8)	Offset (x16)	Acronym	Register Name	Write Protection	Section
0h	0h	ESCSS_CONFIG_LOCK	EtherCATSS Configuration Lock		Go
4h	2h	ESCSS_MISC_IO_CONFIG	RESET_IN, EEPROM IO connections select	LOCK	Go
8h	4h	ESCSS_PHY_IO_CONFIG	Control Register of ESCSS		Go
Ch	6h	ESCSS_SYNC_IO_CONFIG	SYNC Signals IO configurations	LOCK	Go
10h	8h	ESCSS_LATCH_IO_CONFIG	LATCH inputs IO pad select	LOCK	Go
14h	Ah	ESCSS_GPIN_SEL	GPIN Select between IO PAD & tieoff	LOCK	Go
18h	Ch	ESCSS_GPIN_IOPAD_SEL	GPIN IO pad Select	LOCK	Go
1Ch	Eh	ESCSS_GPOUT_SEL	GPOUT IO pad connect select	LOCK	Go
20h	10h	ESCSS_GPOUT_IOPAD_SEL	GPOUT IO pad select	LOCK	Go
24h	12h	ESCSS_LED_CONFIG	Selection of LED o/p connect to IO pad	LOCK	Go
28h	14h	ESCSS_MISC_CONFIG	Miscelleneous Configuration	LOCK	Go

Complex bit access types are encoded to fit into small table cells. Table 31-40 shows the codes that are used for access types in this section.

Table 31-40. ESCSS_CONFIG_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

31.5.3.1 ESCSS_CONFIG_LOCK Register (Offset (x8) = 0h, Offset (x16) = 0h) [Reset = 0h]

ESCSS_CONFIG_LOCK is shown in Figure 31-39 and described in Table 31-41.

Return to the Summary Table.

Lock bit for EtherCAT configuration registers

Figure 31-39. ESCSS_CONFIG_LOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
WRITE_KEY							
R-0/W-0h							
7	6	5	4	3	2	1	0
RESERVED			IO_CONFIG_ENABLE	RESERVED			LOCK_ENABLE
R-0-0h			R/W-0h	R-0-0h			R/WSonce-0h

Table 31-41. ESCSS_CONFIG_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	WRITE_KEY	R-0/W	0h	The key value should be 0xa5 for the writes to bit 0 take effect. Writes of other values will be ignored. Reset type: ECAT.XRSn
7-5	RESERVED	R-0	0h	Reserved
4	IO_CONFIG_ENABLE	R/W	0h	This bit enables the IO configurations allowing the EtherCAT ports to take effect. Till this bit is written EtherCAT ports are not connected to the IO pad. Enable takes effect when this bit is set to 1. Changing IO selections or IO configurations after this bit is set can have unpredictable IO behavior on the device IOs. Reset type: ECAT.XRSn
3-1	RESERVED	R-0	0h	Reserved
0	LOCK_ENABLE	R/WSonce	0h	This bit enables locking the contents of all the EtherCAT configuration registers. The lock takes effect when this bit is set to 1. This bit can be set only once after ecatXRSN and gets reset after the next ecatXRSN. Reset type: ECAT.XRSn

31.5.3.2 ESCSS_MISC_IO_CONFIG Register (Offset (x8) = 4h, Offset (x16) = 2h) [Reset = 2h]

ESCSS_MISC_IO_CONFIG is shown in Figure 31-40 and described in Table 31-42.

Return to the Summary Table.

Condiguration of RESET_IN, EEPROM I2C connections

Figure 31-40. ESCSS_MISC_IO_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
WRITE_KEY							
R-0/W-0h							
7	6	5	4	3	2	1	0
RESERVED						EEPROM_I2C_IO_EN	RESETIN_GPIO_EN
R-0-0h						R/W-1h	R/W-0h

Table 31-42. ESCSS_MISC_IO_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	WRITE_KEY	R-0/W	0h	The key value should be 0xa5 for the writes to this register to take effect. Writes of other values will be ignored. Reset type: ECAT.XRSn
7-2	RESERVED	R-0	0h	Reserved
1	EEPROM_I2C_IO_EN	R/W	1h	Enables connecting EtherCAT I2C connections to IOPAD for EEPROM control 0: EEPROM I2C Connections are not connected to IOPAD. 1: EEPROM I2C connections are driving the IOPAD connections. Reset type: ECAT.XRSn
0	RESETIN_GPIO_EN	R/W	0h	Acts as enabled to receive the Reset input from GPIO pad. 0: RESET_IN GPIO pad is not enabled, only SW & PMM resets affect EtherCAT reset 1: RESET_IN GPIO pad input is connected in reset input cone. Reset type: ECAT.XRSn

31.5.3.3 ESCSS_PHY_IO_CONFIG Register (Offset (x8) = 8h, Offset (x16) = 4h) [Reset = 44h]

ESCSS_PHY_IO_CONFIG is shown in Figure 31-41 and described in Table 31-43.

Return to the Summary Table.

PHY Type, clock source type select

Figure 31-41. ESCSS_PHY_IO_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
WRITE_KEY							
R-0/W-0h							
7	6	5	4	3	2	1	0
RESERVED	TX_CLK_AUTO_COMP	PHY_INTF_IOPAD_SEL		PHY_PORT_CNT		RESERVED	
R/W-0h	R/W-1h	R/W-0h		R/W-1h		R/W-0h	

Table 31-43. ESCSS_PHY_IO_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	WRITE_KEY	R-0/W	0h	The key value should be 0xa5 for the writes to bit 0 take effect. Writes of other values will be ignored. Reset type: ECAT.XRSn
7	RESERVED	R/W	0h	Reserved
6	TX_CLK_AUTO_COMP	R/W	1h	This setting is used to allocate the IO pad for TX_CLK for doing the Auto compensation for the samlping of TXEN & TXDATA. 0 : Manual Compensation using CLK_IN no TX_CLK Pad, IP input is tied to '0'. 1: Auto Compensation based on sampling of TX_CLK. Pad is allocated. Reset type: ECAT.XRSn
5-4	PHY_INTF_IOPAD_SEL	R/W	0h	Selects the PHY position for IO PAD set selection. Groupings can be targeted to suit the device edge or other IO function tradeoffs. While 4 options possible in there could be less than 4 options supported in device, writing mux combination that is not supported defaults to IO pad sets of option"00". Refer to device specifications for IOPAD details for each mapping. Reset type: ECAT.XRSn
3-2	PHY_PORT_CNT	R/W	1h	Indicates the number of PHY ports selected for operation inaddition to Port0 which is default 00-One port operation (Port0) 01-Two port operation(Port0,Port1) 10-Three port operation(Port0,Port1,Port2) : Reserved 11-Four port operation(Port0,Port1,Port2,Port3): Reserved Programming reserved configuration causes selection of Reset value. Reset type: ECAT.XRSn
1-0	RESERVED	R/W	0h	Reserved

31.5.3.4 ESCSS_SYNC_IO_CONFIG Register (Offset (x8) = Ch, Offset (x16) = 6h) [Reset = 88h]

ESCSS_SYNC_IO_CONFIG is shown in Figure 31-42 and described in Table 31-44.

Return to the Summary Table.

SYNC0/1 IO configurations including enable & Pad Select

Figure 31-42. ESCSS_SYNC_IO_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
WRITE_KEY							
R-0/W-0h							
7	6	5	4	3	2	1	0
SYNC1_GPIO_EN	RESERVED	SYNC1_IOPAD_SEL		SYNC0_GPIO_EN	RESERVED	SYNC0_IOPAD_SEL	
R/W-1h	R-0-0h	R/W-0h		R/W-1h	R-0-0h	R/W-0h	

Table 31-44. ESCSS_SYNC_IO_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	WRITE_KEY	R-0/W	0h	The key value should be 0xa5 for the writes to bit 0 take effect. Writes of other values will be ignored. Reset type: ECAT.XRSn
7	SYNC1_GPIO_EN	R/W	1h	Enables the direct mux between Sync1 output of EtherCAT & other GPIO functions. Reset type: ECAT.XRSn
6	RESERVED	R-0	0h	Reserved
5-4	SYNC1_IOPAD_SEL	R/W	0h	Selects the SYNC1 position for IO PAD selection. IOPADposition can be targeted to suit the device edge or other IO function tradeoffs. While 4 options possible in there could be less than 4 options supported in device, writing mux combination that is not supported defaults to IO pad sets of option"00". Refer to device specifications for IOPAD details for each mapping. Reset type: ECAT.XRSn
3	SYNC0_GPIO_EN	R/W	1h	Enables the direct mux between Sync0 output of EtherCAT & other GPIO functions. Reset type: ECAT.XRSn
2	RESERVED	R-0	0h	Reserved
1-0	SYNC0_IOPAD_SEL	R/W	0h	Selects the SYNC0 position for IO PAD selection. IOPADposition can be targeted to suit the device edge or other IO function tradeoffs. While 4 options possible in there could be less than 4 options supported in device, writing mux combination that is not supported defaults to IO pad sets of option"00". Refer to device specifications for IOPAD details for each mapping. Reset type: ECAT.XRSn

31.5.3.5 ESCSS_LATCH_IO_CONFIG Register (Offset (x8) = 10h, Offset (x16) = 8h) [Reset = 88h]

ESCSS_LATCH_IO_CONFIG is shown in Figure 31-43 and described in Table 31-45.

Return to the Summary Table.

LATCH0/1 IO configurations including enable & Pad Select

Figure 31-43. ESCSS_LATCH_IO_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
WRITE_KEY							
R-0/W-0h							
7	6	5	4	3	2	1	0
LATCH1_GPIO_EN	RESERVED	LATCH1_IOPAD_SEL		LATCH0_GPIO_EN	RESERVED	LATCH0_IOPAD_SEL	
R/W-1h	R-0-0h	R/W-0h		R/W-1h	R-0-0h	R/W-0h	

Table 31-45. ESCSS_LATCH_IO_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-8	WRITE_KEY	R-0/W	0h	The key value should be 0xa5 for the writes to bit 0 take effect. Writes of other values will be ignored. Reset type: ECAT.XRSn
7	LATCH1_GPIO_EN	R/W	1h	Enables the direct mux between LATCH1 input from IOPAD & other GPIO functions to the EtherCATSS input Reset type: ECAT.XRSn
6	RESERVED	R-0	0h	Reserved
5-4	LATCH1_IOPAD_SEL	R/W	0h	Selects the LATCH1 position for IO PAD selection. IOPADposition can be targeted to suit the device edge or other IO function tradeoffs. While 4 options possible in there could be less than 4 options supported in device, writing mux combination that is not supported defaults to IO pad sets of option"00". Refer to device specifications for IOPAD details for each mapping. Reset type: ECAT.XRSn
3	LATCH0_GPIO_EN	R/W	1h	Enables the direct mux between LATCH0 input from IOPAD & other GPIO functions to the EtherCATSS input Reset type: ECAT.XRSn
2	RESERVED	R-0	0h	Reserved
1-0	LATCH0_IOPAD_SEL	R/W	0h	Selects the LATCH0 position for IO PAD selection. IOPADposition can be targeted to suit the device edge or other IO function tradeoffs. While 4 options possible in there could be less than 4 options supported in device, writing mux combination that is not supported defaults to IO pad sets of option"00". Refer to device specifications for IOPAD details for each mapping. Reset type: ECAT.XRSn

31.5.3.6 ESCSS_GPIN_SEL Register (Offset (x8) = 14h, Offset (x16) = Ah) [Reset = 0h]

ESCSS_GPIN_SEL is shown in Figure 31-44 and described in Table 31-46.

Return to the Summary Table.

Register to configure each GPI input is connected to IO-pad or not.

Figure 31-44. ESCSS_GPIN_SEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIN_SEL																															
R/W-0h																															

Table 31-46. ESCSS_GPIN_SEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPIN_SEL	R/W	0h	Allows bit-wise selection of the GPIN be connected from GPIO PAD. Once those are not driven by GPIO, will be driven from register writable from local Host. 0: No connection to GPIO PAD, but connects to ESCSS_GPIN_DAT. 1: Mux Select the GPIN from the dedicated IO PAD. This acts as Mux select for input from GPIO over tieoff. Reset type: ECAT.XRSn

31.5.3.7 ESCSS_GPIN_IOPAD_SEL Register (Offset (x8) = 18h, Offset (x16) = Ch) [Reset = 0h]

ESCSS_GPIN_IOPAD_SEL is shown in Figure 31-45 and described in Table 31-47.

Return to the Summary Table.

Register to configure each GPI input is connected to IO-pad or not.

Figure 31-45. ESCSS_GPIN_IOPAD_SEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIN_IOPAD_SEL																															
R/W-0h																															

Table 31-47. ESCSS_GPIN_IOPAD_SEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPIN_IOPAD_SEL	R/W	0h	This is to allow the EtherCAT GPIN input be taken from either of the two IO-pads allocated. The details of which IOs are allocated will be listed in device spec. In case only one IOPAD is allocated, the corresponding bit of the GPIN does not have the effect. Reset type: ECAT.XRSn

31.5.3.8 ESCSS_GPOUT_SEL Register (Offset (x8) = 1Ch, Offset (x16) = Eh) [Reset = 0h]

ESCSS_GPOUT_SEL is shown in Figure 31-46 and described in Table 31-48.

Return to the Summary Table.

Register to configure each GPO to be connected to IO-pad or not.

Figure 31-46. ESCSS_GPOUT_SEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPOUT_SEL																															
R/W-0h																															

Table 31-48. ESCSS_GPOUT_SEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPOUT_SEL	R/W	0h	Allows bit-wise selection for GPOUT connection to IO PAD. hence acts as direct mux select between GPO & other non-EtherCAT functions. 0: GPO is not connected to dedicated IO instead non-EtherCAT function is connected. 1: Connect the GPOUT to the dedicated IO pad through output buffer. Reset type: ECAT.XRSn

31.5.3.9 ESCSS_GPOUT_IOPAD_SEL Register (Offset (x8) = 20h, Offset (x16) = 10h) [Reset = 0h]

ESCSS_GPOUT_IOPAD_SEL is shown in Figure 31-47 and described in Table 31-49.

Return to the Summary Table.

Register to configure each GPO to be connected to IO-pad or not.

Figure 31-47. ESCSS_GPOUT_IOPAD_SEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPOUT_IOPAD_SEL																															
R/W-0h																															

Table 31-49. ESCSS_GPOUT_IOPAD_SEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPOUT_IOPAD_SEL	R/W	0h	This is to allow the EtherCAT GPOUT output be driven to either of the two IO-pads allocated. The details of which IOs are allocated will be listed in device spec. In case only one IOPAD is allocated, the corresponding bit of the GPOUT does not have the effect. Reset type: ECAT.XRSn

31.5.3.10 ESCSS_LED_CONFIG Register (Offset (x8) = 24h, Offset (x16) = 12h) [Reset = 0h]

ESCSS_LED_CONFIG is shown in Figure 31-48 and described in Table 31-50.

Return to the Summary Table.

Register to select of LED o/p is connected to IO-PAD

Figure 31-48. ESCSS_LED_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RUN_IOPAD_SEL		ERR_IOPAD_SEL		STATE_IOPAD_SEL		LINKACT1_IOPAD_SEL	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
LINKACT0_IOPAD_SEL		RESERVED	RUN	ERR	STATE	LINKACT1	LINKACT0
R/W-0h		R-0-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 31-50. ESCSS_LED_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R-0	0h	Reserved
15-14	RUN_IOPAD_SEL	R/W	0h	Selects the RUN LED position for IO PAD selection. IOPADposition can be targeted to suit the device edge or other IO function tradeoffs. While 4 options possible in there could be less than 4 options supported in device, writing mux combination that is not supported defaults to IO pad sets of option"00". Refer to device specifications for IOPAD details for each mapping. Reset type: ECAT.XRSn
13-12	ERR_IOPAD_SEL	R/W	0h	Selects the ERROR LED position for IO PAD selection. IOPADposition can be targeted to suit the device edge or other IO function tradeoffs. While 4 options possible in there could be less than 4 options supported in device, writing mux combination that is not supported defaults to IO pad sets of option"00". Refer to device specifications for IOPAD details for each mapping. Reset type: ECAT.XRSn
11-10	STATE_IOPAD_SEL	R/W	0h	Selects the STATE LED position for IO PAD selection. IOPADposition can be targeted to suit the device edge or other IO function tradeoffs. While 4 options possible in there could be less than 4 options supported in device, writing mux combination that is not supported defaults to IO pad sets of option"00". Refer to device specifications for IOPAD details for each mapping. Reset type: ECAT.XRSn
9-8	LINKACT1_IOPAD_SEL	R/W	0h	Selects the LINKACT1 LED position for IO PAD selection. IOPADposition can be targeted to suit the device edge or other IO function tradeoffs. While 4 options possible in there could be less than 4 options supported in device, writing mux combination that is not supported defaults to IO pad sets of option"00". Refer to device specifications for IOPAD details for each mapping. Reset type: ECAT.XRSn

Table 31-50. ESCSS_LED_CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	LINKACT0_IOPAD_SEL	R/W	0h	Selects the LINKACT0 LED position for IO PAD selection. IOPADposition can be targeted to suit the device edge or other IO function tradeoffs. While 4 options possible in there could be less than 4 options supported in device, writing mux combination that is not supported defaults to IO pad sets of option"00". Refer to device specifications for IOPAD details for each mapping. Reset type: ECAT.XRSn
5	RESERVED	R-0	0h	Reserved
4	RUN	R/W	0h	Acts as Mux select to enable RUN LED function directly onto output pad. 0: The non-EtherCAT function is selected on the IO 1: RUN LED is selected to be output on the IO This selection assumes both buffer input and buffer enable connection as required. Reset type: ECAT.XRSn
3	ERR	R/W	0h	Acts as Mux select to enable ERR LED function directly onto output pad. 0: The non-EtherCAT function is selected on the IO 1: ERR LED is selected to be output on the IO This selection assumes both buffer input and buffer enable connection as required. Reset type: ECAT.XRSn
2	STATE	R/W	0h	Acts as Mux select to enable STATE LED function directly onto output pad. 0: The non-EtherCAT function is selected on the IO 1: STATE LED is selected to be output on the IO This selection assumes both buffer input and buffer enable connection as required. Reset type: ECAT.XRSn
1	LINKACT1	R/W	0h	Acts as Mux select to enable LINKACT1 function directly onto output pad. 0: The non-EtherCAT function is selected on the IO 1: LINKACT1 is selected to be output on the IO This selection assumes both buffer input and buffer enable connection as required. Reset type: ECAT.XRSn
0	LINKACT0	R/W	0h	Acts as Mux select to enable LINKACT0 function directly onto output pad. 0: The non-EtherCAT function is selected on the IO 1: LINKACT0 is selected to be output on the IO This selection assumes both buffer input and buffer enable connection as required. Reset type: ECAT.XRSn

31.5.3.11 ESCSS_MISC_CONFIG Register (Offset (x8) = 28h, Offset (x16) = 14h) [Reset = 0h]

ESCSS_MISC_CONFIG is shown in Figure 31-49 and described in Table 31-51.

Return to the Summary Table.

Configuration info for the MII interface containing TX_SHIFT compensation values, PHY Address offset, EEPROM SIZE etc.

Figure 31-49. ESCSS_MISC_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0-0h							
15	14	13	12	11	10	9	8
RESERVED					PHY_ADDR		
R-0-0h					R/W-0h		
7	6	5	4	3	2	1	0
PHY_ADDR		PDI_EMULATION	EEPROM_SIZE	TX1_SHIFT_CONFIG		TX0_SHIFT_CONFIG	
R/W-0h		R/W-0h	R/W-0h	R/W-0h		R/W-0h	

Table 31-51. ESCSS_MISC_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R-0	0h	Reserved
10-6	PHY_ADDR	R/W	0h	These bits will be hooked up to the PHY_OFFSET[4:0] input of the EtherCAT IP. Reset type: ECAT.XRSn
5	PDI_EMULATION	R/W	0h	This bit will be hooked up to the PDI_EMULATION input of the EtherCAT IP. Reset type: ECAT.XRSn
4	EEPROM_SIZE	R/W	0h	This bit will be hooked up to the EEPROM_SIZE input of the EtherCAT IP . This is set to 0 for EEPROMs of size 16K bits or lower. This is set to 1 for EEPROMs of size above 16K bits. Reset type: ECAT.XRSn
3-2	TX1_SHIFT_CONFIG	R/W	0h	Two bit TX_SHIFT configuration in terms of 10ns counts for port0. This is the shift added to TX_ENA & TX_DATA to match delay of PHY TX_CLK w.r.t. device internal clock. Reset type: ECAT.XRSn
1-0	TX0_SHIFT_CONFIG	R/W	0h	Two bit TX_SHIFT configuration in terms of 10ns counts for port0. This is the shift added to TX_ENA & TX_DATA to match delay of PHY TX_CLK w.r.t. device internal clock. Reset type: ECAT.XRSn

31.5.4 ESC_SS Registers to Driverlib Functions

Table 31-52. ESC_SS Registers to Driverlib Functions

File	Driverlib Function
IPREVNUM	
escss.h	ESCSS_readIPMinorRevNumber
escss.h	ESCSS_readIPMajorRevNumber
escss.h	ESCSS_readIPRevNumber
INTR_RIS	

Table 31-52. ESC_SS Registers to Driverlib Functions (continued)

File	Driverlib Function
escss.h	ESCSS_getRawInterruptStatus
escss.h	ESCSS_readRawInterruptStatus
INTR_MASK	
escss.h	ESCSS_setMaskedInterruptStatus
escss.h	ESCSS_resetMaskedInterruptStatus
INTR_MIS	
escss.h	ESCSS_getMaskedInterruptStatus
INTR_CLR	
escss.h	ESCSS_clearRawInterruptStatus
INTR_SET	
escss.c	ESCSS_setRawInterruptStatus
LATCH_SEL	
escss.h	ESCSS_selectLatch0Mux
escss.h	ESCSS_selectLatch1Mux
ACCESS_CTRL	
escss.h	ESCSS_configure16BitAsyncAccessWaitState
escss.h	ESCSS_enablePDIAccessTimeOut
escss.h	ESCSS_disablePDIAccessTimeOut
escss.h	ESCSS_enableDebugAccess
escss.h	ESCSS_disableDebugAccess
GPIN_DAT	
escss.h	ESCSS_readGPINData
escss.h	ESCSS_setGPINData
escss.h	ESCSS_resetGPINData
GPIN_PIPE	
escss.h	ESCSS_enableGPIPipelinedRegCaptureOnEvent
escss.h	ESCSS_disableGPIPipelinedRegCaptureOnEvent
GPIN_GRP_CAP_SEL	
escss.c	ESCSS_setGPINGroupCaptureTriggerSelect
GPOUT_DAT	
escss.h	ESCSS_readGPOUTData
GPOUT_PIPE	
escss.h	ESCSS_enableGPOUTPipelinedRegCaptureOnEvent
escss.h	ESCSS_disableGPOUTPipelinedRegCaptureOnEvent
GPOUT_GRP_CAP_SEL	
escss.c	ESCSS_setGPOUTGroupCaptureTriggerSelect
MEM_TEST	
escss.h	ESCSS_initMemory
escss.h	ESCSS_getMemoryInitDoneStatusNonBlocking
escss.h	ESCSS_getMemoryInitDoneStatusBlocking
RESET_DEST_CONFIG	
escss.c	ESCSS_enableCPUReset
escss.c	ESCSS_disableCPUReset
escss.c	ESCSS_enableResetToNMI
escss.c	ESCSS_disableResetToNMI

Table 31-52. ESC_SS Registers to Driverlib Functions (continued)

File	Driverlib Function
escss.c	ESCSS_enableResetToInterrupt
escss.c	ESCSS_disableResetToInterrupt
SYNC0_CONFIG	
escss.c	ESCSS_configureSync0Connections
SYNC1_CONFIG	
escss.c	ESCSS_configureSync1Connections
CONFIG_LOCK	
escss.c	ESCSS_enableConfigurationLock
escss.c	ESCSS_enableIOConnectionLock
escss.c	ESCSS_disableIOConnectionLock
escss.h	ESCSS_isConfigurationLockEnabled
MISC_IO_CONFIG	
escss.c	ESCSS_enableResetInputFromGpioPad
escss.c	ESCSS_disableResetInputFromGpioPad
escss.c	ESCSS_enableESCEEPROMI2CIoPadConnection
escss.c	ESCSS_disableESCEEPROMI2CIoPadConnection
PHY_IO_CONFIG	
escss.c	ESCSS_configurePortCount
escss.c	ESCSS_enableAutoCompensationTxClkIOPad
escss.c	ESCSS_disableAutoCompensationTxClkIOPad
SYNC_IO_CONFIG	
escss.c	ESCSS_enableSync0GpioMuxConnection
escss.c	ESCSS_disableSync0GpioMuxConnection
escss.c	ESCSS_enableSync1GpioMuxConnection
escss.c	ESCSS_disableSync1GpioMuxConnection
LATCH_IO_CONFIG	
escss.c	ESCSS_enableLatch0GpioMuxConnection
escss.c	ESCSS_disableLatch0GpioMuxConnection
escss.c	ESCSS_enableLatch1GpioMuxConnection
escss.c	ESCSS_disableLatch1GpioMuxConnection
GPIN_SEL	
escss.h	ESCSS_enableGPIN
escss.h	ESCSS_disableGPIN
GPIN_IOPAD_SEL	
-	
GPOUT_SEL	
escss.h	ESCSS_enableGPOUT
escss.h	ESCSS_disableGPOUT
GPOUT_IOPAD_SEL	
-	
LED_CONFIG	
escss.h	ESCSS_enableLEDOptions
escss.h	ESCSS_disableLEDOptions
MISC_CONFIG	
escss.c	ESCSS_configureEEPROMSize

Table 31-52. ESC_SS Registers to Driverlib Functions (continued)

File	Driverlib Function
escss.h	ESCSS_configureTX0ShiftForTxEnaAndTxData
escss.h	ESCSS_configureTX1ShiftForTxEnaAndTxData
escss.h	ESCSS_enablePDIEmulation
escss.h	ESCSS_disablePDIEmulation
escss.h	ESCSS_configurePhyAddressOffset

This page intentionally left blank.

Chapter 32

Fast Serial Interface (FSI)

This chapter contains a general description of the Fast Serial Interface (FSI) module. The FSI is a serial peripheral capable of reliable high-speed communication across isolation barriers.

32.1 Introduction

The Fast Serial Interface (FSI) module is a serial communication peripheral capable of reliable high-speed communication across isolation devices. Galvanic isolation devices are used in situations where two different electronic circuits, which do not have common power and ground connections, must exchange information. Though isolation devices facilitate these signal communications, they can also introduce a large delay on the signal lines and add skew between the signals. The FSI is designed specifically to ensure reliable high-speed communication for system scenarios that involve communication across isolation barriers without adding components.

The FSI consists of independent transmitter (FSITX) and receiver (FSIRX) cores. The FSITX and FSIRX cores are configured and operated independently.

For additional information on the FSI module, refer to *Fast Serial Interface (FSI) Skew Compensation*.

32.1.1 FSI Related Collateral

Foundational Materials

- C2000 Academy - FSI

Getting Started Materials

- *Fast Serial Interface (FSI) Skew Compensation Application Report*
- Fast serial interface (FSI) adapter board evaluation module
- *Using the Fast Serial Interface (FSI) With Multiple Devices in an Application Application Report*

Expert Materials

- *Design Guide: TIDM-02006 Distributed Multi-axis Servo Drive Over Fast Serial Interface (FSI) Reference Design*
- *The Essential Guide for Developing With C2000 Real-Time Microcontrollers Application Report*
  - Refer to the See sections 'Distributed Real-Time Control Across an Isolation Boundary' and 'Solving Event Synchronization Across Multiple Controllers in Decentralized Control Systems'. section

32.1.2 FSI Features

The FSI module includes the following features:

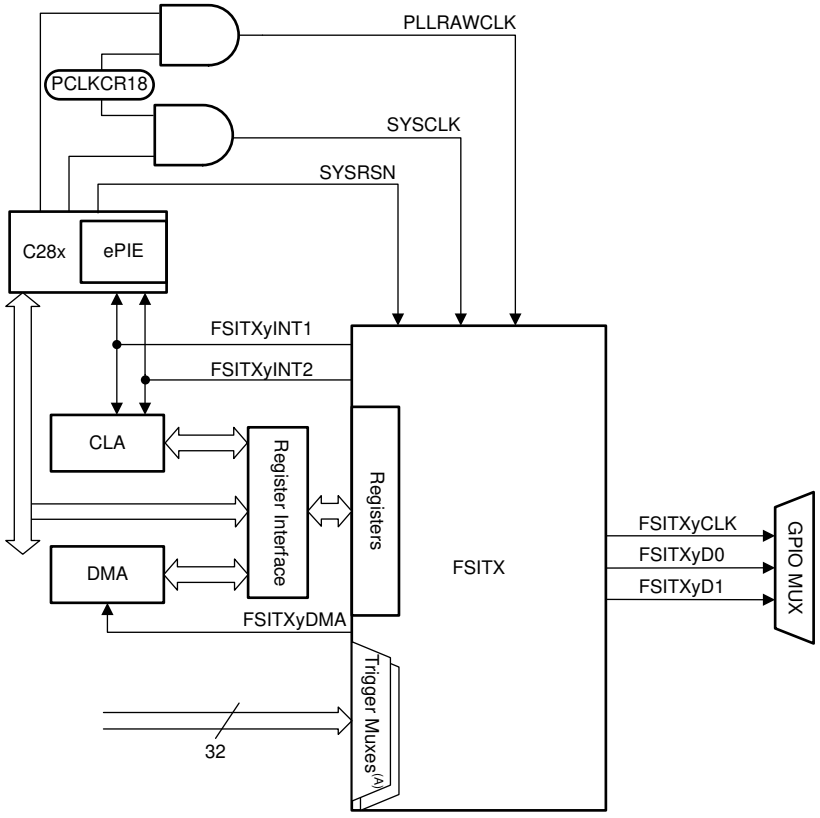
- Independent transmitter and receiver cores
- Source-synchronous transmission
- Double Data Rate (DDR)
- One or two data lines
- Programmable data length
- Skew adjustment block to compensate for board and system delay mismatches
- Frame error detection
- Programmable frame tagging for message filtering
- Hardware ping to detect line breaks during communication (ping watchdog)
- Two interrupts per FSI core
- Externally triggered frame generation
- Hardware- or software-calculated CRC
- Embedded ECC computation module
- Register write protection
- FSI-SPI compatibility mode (limited features available)
- Tag match notifications

32.2 System-level Integration

This section describes the device-level integration of the FSI module. Some of the features can require additional configuration of modules that are not within the scope of this chapter, the details can be found elsewhere in this TRM.

32.2.1 CPU Interface

The following diagrams show the CPU interface of each FSI module.



A. The signals connected to the trigger muxes are described in Section 32.2.6.

Figure 32-1. FSI Transmitter (FSITX) CPU Interface

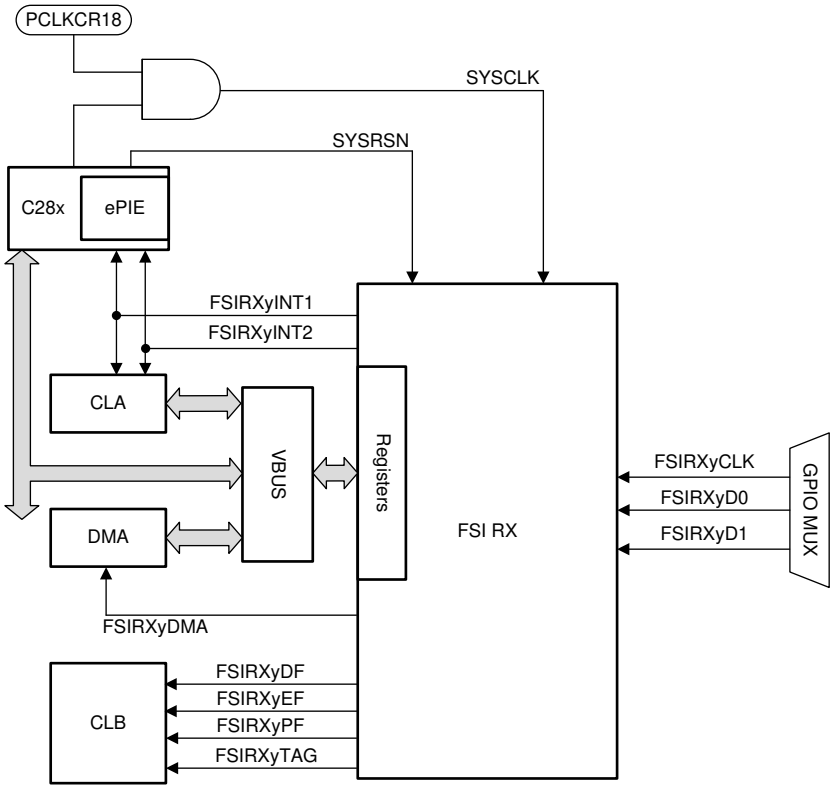


Figure 32-2. FSI Receiver (FSIRX) CPU Interface with CLB

32.2.2 Signal Description

FSI is a point-to-point communication protocol. Hence, an FSI transmitter core communicates directly to a single FSI receiver core. Similarly, an FSI receiver core receives data from a single FSI transmitter core.

Each FSI core has three signals: one clock and two data signals. Data is always transmitted or received with the most-significant bit of each frame field being first. If multi-lane transmissions are not used, the TXD1 and RXD1 signals can be left unconnected and their GPIOs repurposed for other application needs. Table 32-1 and Table 32-2 describe the various signals that can be selected by the PADCONFIG register to be brought out to device pins.

CAUTION
The maximum RXCLK rate is SYSCLK/2 and must not exceed this limit.

Table 32-1. FSI Receiver Core Signals

Signal Name	Direction	Description	Inactive Level(1)
RXCLK	Input	This is the receive clock input signal for the FSI receive module. This must be connected to TXCLK of the transmitting FSI module.	Logic High
RXD0	Input	This is the primary data input line for reception. This must be connected to the TXD0 of the transmitting FSI module.	Logic High
RXD1	Input	This is an additional data input line for reception. This signal must be connected to the TXD1 of the transmitting FSI module, if multi-lane transmission is used.	Logic High

(1) Inactive level refers to the state of the pin while the module is not actively receiving data.

Table 32-2. FSI Transmitter Core Signals

Signal Name	Direction	Description	Inactive Level(1)
TXCLK	Output	This is the transmit clock and is driven by the FSI transmit module. During a transmission, four clock edges are transmitted before the start of frame phase (preamble) and four clock edges follow the last bit of the frame (postamble). Data is transmitted on both edges of the clock. In FSI-SPI compatibility mode, the preamble and the post frame clock edges are not transmitted. Data is transmitted only on one edge of the clock. Data transmits on rising edge and received on falling edge of the clock.	Logic High
TXD0	Output	This is the primary data output line for transmission and is driven by the FSI transmit module. When the FSI is configured for multi-lane transmission, TXD0 contains all the even numbered bits of the data and CRC bytes. Other frame fields such as frame type, start-of-frame, tag, and end-of-frame are transmitted in full.	Logic High
TXD1	Output	This is an additional data output line for transmission, if the FSI is configured for multi-lane transmission. This signal is driven by the FSI transmit module. During transmission, the data bits are split between TXD0 and TXD1. TXD1 contains all the odd numbered bits of the data and CRC bytes. This applies only to the data words and the CRC bytes. Other data frame related information like Frame Type, Start-of-Frame, Tag and End-of-frame, the state of this line are identical to TXD0.	Logic High

(1) Inactive level refers to the state of the pin while the module is not actively transmitting, or held in reset.

32.2.2.1 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

Some IO functionality is defined by GPIO register settings independent of this peripheral. For input signals, the GPIO input qualification must be set to asynchronous mode by setting the appropriate GPxQSELn register bits to 0x3. The internal pullups can be configured in the GPyPUD register. See the *General Purpose Input-Output (GPIO)* chapter for more details on the GPIO mux and settings.

32.2.3 FSI Interrupts

Each FSI module contains multiple interrupt sources that can be assigned to two different interrupt vectors: INT1 and INT2. Each interrupt source has an associated status flag, force, and clear bits in the EVT_STS, EVT_FRC, and the EVT_CLR registers, respectively.

Each interrupt can be assigned to either interrupt vector, INT1 and INT2, to allow for two priority levels. Alternately, the interrupt source can be prevented from generating any interrupt, though the status flag can still be set and monitored by software. The transmitter events are assigned to either interrupt vector in the TX_INT_CTRL register. The receiver events are assigned an interrupt vector using RX_INT1_CTRL and RX_INT2_CTRL registers. If an interrupt is not required, make sure the bit is not set in the respective INT_CTRL register.

32.2.3.1 Transmitter Interrupts

The transmitter can generate the following interrupts:

- **Frame Done (FRAME_DONE):** This event indicates that FSI has completed transmitting a frame.
- **Buffer Underrun (BUF_UNDERRUN):** This event indicates that the transmit buffer has experienced underrun. Buffer underrun occurs when the transmitter tries to read data from a location which has not yet be written to by the CLA, CPU, or DMA.
- **Buffer Overrun (BUF_OVERRUN):** The buffer overrun interrupt is generated when the buffer has experienced overrun. Buffer overrun can occur if a piece of data is overwritten before the data has been transmitted.
- **Ping Frame Triggered (PING_TRIGGERED):** The ping frame triggered interrupt is generated when the ping frame has been triggered. This bit is set when the ping counter has timed out or an external ping trigger event has occurred.

32.2.3.2 Receiver Interrupts

The receiver core is capable of generating interrupts from many different events:

- **Ping Watchdog Timeout (PING_WD_TO):** This event indicates that the ping watchdog timer has timed out. The receiver has not received a valid frame within the time period specified in the RX_PING_WD_REF register.
- **Frame Watchdog Timeout (FRAME_WD_TO):** This event indicates that the frame watchdog timer has timed out. The conditions of this timeout are set using the RX_FRAME_WD_CTRL register. As soon as the start of frame phase is detected, the frame watchdog counter starts counting from 0. The end of frame phase must complete by the time the watchdog counter reaches the reference value. If this does not happen, the watchdog times out and this event is generated. If this event occurs, the receiver must undergo a soft reset and subsequent resynchronization to resume proper operation.
- **CRC Error (CRC_ERR):** This error indicates that a CRC error has occurred. A CRC error is generated when the received CRC and the computed CRC do not match.
- **Frame Type Error (TYPE_ERR):** This error indicates that an invalid frame type has been received. If this error occurs, the receiver must undergo a soft reset and subsequent resynchronization to resume proper operation.
- **End-of-Frame Error (EOF_ERR):** This error indicates that an invalid end-of-frame bit pattern has been received. If this error occurs, the receiver must undergo a soft reset and subsequent resynchronization to resume proper operation.
- **Receive Buffer Overrun (BUF_OVERRUN):** This event indicates that an overrun condition has occurred in the receive buffer.
- **Receive Buffer Underrun (BUF_UNDERRUN):** This event indicates that an underrun condition has occurred in the receive buffer. This condition occurs when software reads an empty buffer.
- **Frame Done (FRAME_DONE):** This event indicates that a valid frame has been received without error.
- **Error Frame Received (ERR_FRAME):** This event indicates that an error frame has been received.
- **Ping Frame Received (PING_FRAME):** This event indicates that a ping frame has been received.
- **Frame Overrun (FRAME_OVERRUN):** This event indicates that a new frame has been received while the FRAME_DONE flag was still set.
- **Data Frame Received (DATA_FRAME):** This event indicates that a data frame has been received.
- **Ping Tag Matched (PING_TAG_MATCH):** This event indicates that a ping frame with a matching tag has been received.
- **Data Tag Matched (DATA_TAG_MATCH):** This event indicates that a data frame with a matching tag has been received.
- **Error Tag Matched (ERROR_TAG_MATCH):** This event indicates that an error frame with a matching tag has been received.

32.2.3.3 Configuring Interrupts

To configure interrupts on the FSI, the application must select the interrupt vector for each desired event using the TX_INT_CTRL register for the transmitter, and RX_INT1_CTRL and RX_INT2_CTRL registers for the receiver. There is no module-level interrupt enable bit to configure.

Note

If an event is registered for both interrupt vectors, both interrupts fire. There are no hardware checks for overlapping interrupt vector assignments.

32.2.3.4 Handling Interrupts

Inside the interrupt service routine (ISR), the user must clear the event flag using the EVT_CLR register and then acknowledge the CPU interrupt.

If the one event occurs multiple times before the corresponding bit is cleared by software, no new interrupt is generated.

If multiple events occur simultaneously, or very close in time, it is possible to handle multiple conditions within a single interrupt. Each flag is independently set by hardware and must be cleared by application software. If multiple different events occur, the ISR can handle each in whatever order is deemed necessary by the application. It is not advisable to clear the full interrupt status register in every ISR. This can cause the application to miss events that can be detrimental to the application. A sample sequence for handling interrupts on the receiver follows; the transmitter routine is similar.

- On receiving an interrupt, copy the current state of the receive event and error status flag register (RX_EVT_STS) into a local snapshot variable.
- Read all of the bits from the snapshot to determine the events that require action.
- Perform the necessary actions for each of the events seen in the snapshot.
- Write to the receive event and error clear register (RX_EVT_CLR) with the snapshot to clear only those interrupts that were set at the beginning of the ISR.
- Repeat this sequence for every generated ISR.

There is a chance that another event occurred during the just-handled ISR since only the snapshot of events was handled and then cleared; an event flag can still be set at the end of the ISR. As soon as the ISR completes, a new interrupt is generated and this flag is still set and can be handled accordingly.

Software accesses tied to multiple events and handled within the same ISR can cause race conditions that cause the software to not function as desired. For example, it is recommended to use different interrupt lines if the user wants to enable events for both ping and data frames. If both events are handled within the same interrupt line, the software can only respond to one of the events if both events occur close in time.

32.2.4 CLA Task Triggering

In addition to generating interrupt vectors to the PIE, both interrupts lines for each module TX_INT1, TX_INT2, RX_INT1, and RX_INT2 can be assigned to trigger CLA tasks. Refer to the Configuration options table for the list of all sources capable of CLA task triggering. The configuration and use of CLA tasks are described in Section 8.2.4. The CLA has access to the entire FSI register map. This allows the CLA to manage the FSI independently from the CPU, freeing it up for other tasks.

32.2.5 DMA Interface

Both the transmitter and receiver are capable of using the DMA for automatic data transfers. The DMA trigger is independent from the interrupt signals. DMA events are only triggered on the completion of a data frame.

The transmitter DMA trigger is enabled by setting TX_DMA_CTRL.DMA_EVT_EN to 1. The transmitter must also set TX_OPER_CTRL_LO.START_MODE to 0x2 to allow either a write to the TX_FRAME_CTRL.START bit or to the TX_FRAME_TAG_UDATA register to start the transmission.

The receiver DMA trigger is enabled by setting RX_DMA_CTRL.DMA_EVT_EN to 1.

Refer to Section 32.3.2 and Section 32.3.3 for more DMA information specific to each FSI Module.

32.2.6 External Frame Trigger Mux

The FSI has two muxes connected to the transmitter module. These muxes are used to select triggers to start ping frames, and generic frames. These muxes are independently configured for each type of frame. The application can select one trigger source per frame type. Use of these triggers are optional.

The external ping frame trigger is configured by setting TX_PING_CTRL.EXT_TRIG_SEL to the index of the desired trigger. TX_PING_CTRL.EXT_TRIG_EN must also be set to allow the trigger to generate a ping frame.

The generic frame trigger is configured by setting TX_OPER_CTRL_HI.EXT_TRIG_SEL to the index of the desired trigger. TX_OPER_CTRL_LO.START_MODE must be set to 0x1 for a frame to be transmitted by an external trigger.

Note

Triggers generated by the CLB are asynchronous and must be at least 3 SYSCLKs wide.

Table 32-3. External Trigger Sources and Their Index

Index	External Trigger Source
0	EPWMXBAR1-TRIP4
1	EPWMXBAR2-TRIP5
2	EPWMXBAR3-TRIP7
3	EPWMXBAR4-TRIP8
4	EPWMXBAR5-TRIP9
5	EPWMXBAR6-TRIP10
6	EPWMXBAR7-TRIP11
7	EPWMXBAR8-TRIP12
8	EPWM1_SOCA
9	EPWM1_SOCB
10	EPWM2_SOCA
11	EPWM2_SOCB
12	EPWM3_SOCA
13	EPWM3_SOCB
14	EPWM4_SOCA
15	EPWM4_SOCB
16	EPWM5_SOCA
17	EPWM5_SOCB
18	EPWM6_SOCA
19	EPWM6_SOCB
20	EPWM7_SOCA
21	EPWM7_SOCB
22	EPWM8_SOCA
23	EPWM8_SOCB
24	EPWM9_SOCA
25	EPWM9_SOCB
26	EPWM10_SOCA
27	EPWM10_SOCB
28	EPWM11_SOCA
29	EPWM11_SOCB
30	EPWM12_SOCA
31	EPWM12_SOCB
32	EPWM13_SOCA
33	EPWM13_SOCB
34	EPWM14_SOCA
35	EPWM14_SOCB
36	EPWM15_SOCA
37	EPWM15_SOCB
38	EPWM16_SOCA

Table 32-3. External Trigger Sources and Their Index (continued)

Index	External Trigger Source
39	EPWM16_SOCB
40	CLB1_OUT30
41	CLB1_OUT31
42	CLB2_OUT30
43	CLB2_OUT31
44	CLB3_OUT30
45	CLB3_OUT31
46	CLB4_OUT30
47	CLB4_OUT31
48	CLB5_OUT30
49	CLB5_OUT31
50	CLB6_OUT30
51	CLB6_OUT31
52	ADCSOCAO
53	ADCSOCBO
54	CPU1_TINT0
55	CPU1_TINT1
56	CPU1_TINT2
57	CPU2_TINT0
58	CPU2_TINT1
59	CPU2_TINT2
60	CPU1_CLATASKRUN1
61	CPU1_CLATASKRUN2
62	CPU2_CLATASKRUN1
63	CPU2_CLATASKRUN2

32.3 FSI Functional Description

32.3.1 Introduction to Operation

The Fast Serial Interface Transmitter and Receiver modules (FSI_TX/FSI_RX) are two completely independent modules on the device. Each module has an independent set of control registers, clocking, and interrupts. The following sections describe the frame format and the various initialization and configuration procedures for both the transmitter and receiver.

32.3.2 FSI Transmitter Module

The FSI transmitter module handles the framing of data, CRC generation, and signal generation of TXCLK, TXD0, and TXD1, as well as interrupt generation. The operation of the transmitter core is controlled and configured through programmable control registers. The transmitter control registers allow the CPU (or the CLA) to program, control, and monitor the operation of the FSI receiver. The transmit data buffer is accessible by the CPU, CLA, and the DMA.

The transmitter has the following features:

- Automated ping frame generation
- Externally triggered ping frames
- Externally triggered data frames
- Software-configurable frame lengths
- 16-word data buffer
- Data buffer underrun and overrun detection
- Hardware-generated CRC on data bits
- Software ECC calculation on select data
- DMA support
- CLA task triggering

Figure 32-3 shows the high-level block diagram of the FSI transmitter. Figure 32-4 shows the block diagram of the transmitter core submodule.

The following sections describe the various aspects of the FSI transmitter in detail.

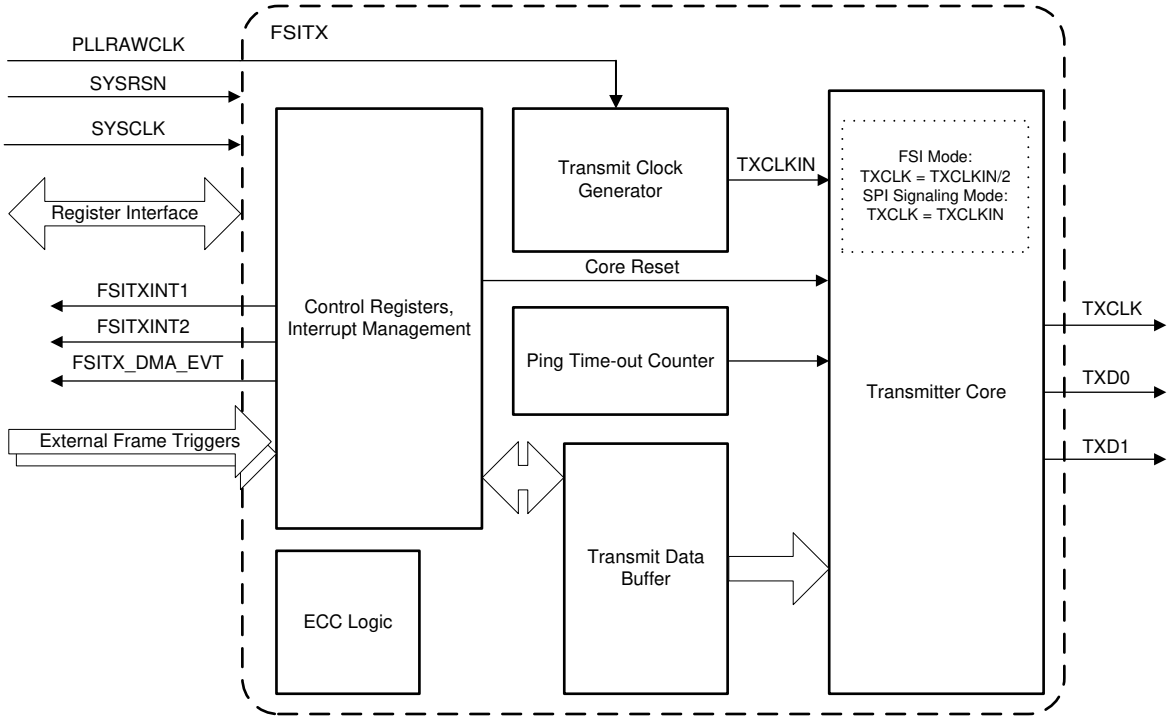


Figure 32-3. FSI Transmitter Block Diagram

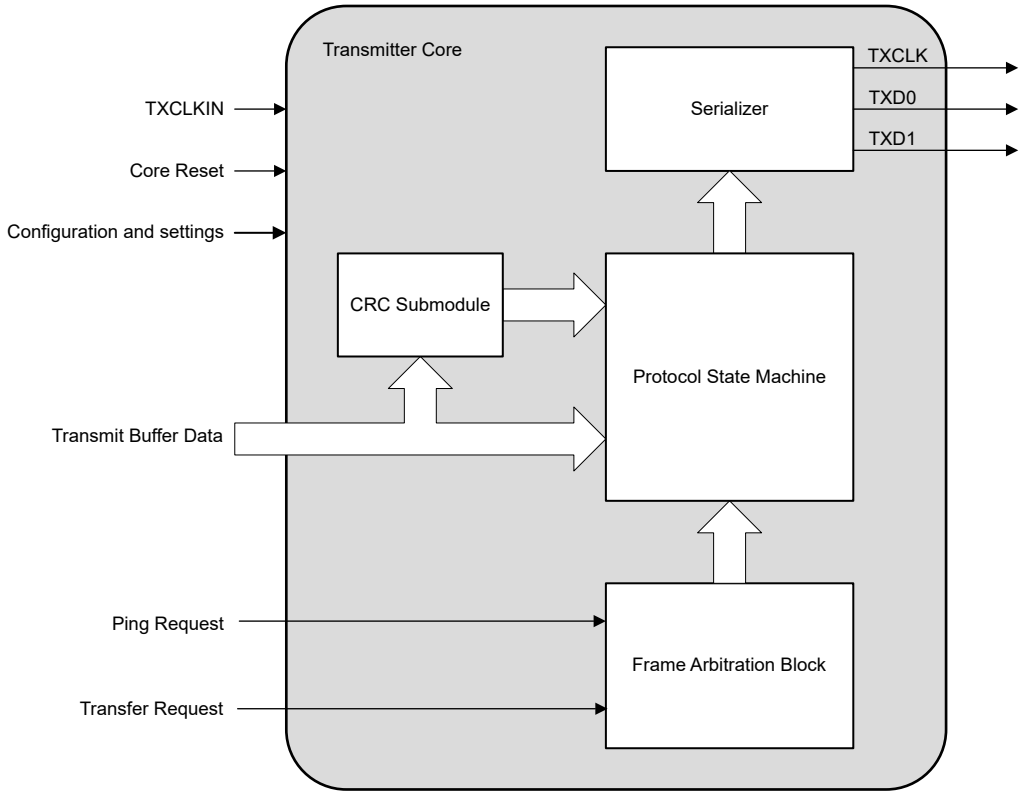


Figure 32-4. FSI Transmitter Core Block Diagram

32.3.2.1 Initialization

On the first initialization or after a module reset due to an underrun condition, the transmitter module executes the following initialization sequence to start or resume transmit operations.

1. Initialize the transmitter clock by setting TX_CLK_CTRL.CLK_RST to 1 and subsequently clearing the bit.
2. Set the clock to the transmitter core to PLLRAWCLK by setting TX_OPER_CTRL_LO.SEL_PLLCLK to 1.
3. Set the clock prescaler value to the desired rate by writing to TX_CLK_CTRL.PRESCALE_VAL.
4. Enable the transmitter clock divider by setting TX_CLK_CTRL.CLK_EN to 1.
5. Assert the transmitter module soft reset by writing 0xA501 to TX_MASTER_CTRL.
6. Wait four TXCLK cycles.
7. Release the transmitter core from reset by writing 0xA500 to TX_MASTER_CTRL.

After initialization and configuration, the transmitter module synchronizes with the receiver module before transmitting. The synchronization sequence is described in Section 32.4.1.

CAUTION

Do not change TX_CLK_CTRL.PRESCALE_VAL while the clock is enabled (TX_CLK_CTRL.CLK_EN = 1). Doing so can cause undefined behavior.

32.3.2.2 FSI_TX Clocking

The transmitter core registers and control logic run off of the device system clock (SYSCLK).

The FSI Transmit Clock (TXCLK) is derived from PLLRAWCLK. PLLRAWCLK is divided down by configuring the clock prescaler value (TX_CLK_CTRL.PRESCALE_VAL) then setting the clock divider enable bit (TX_CLK_CTRL.CLK_EN). The clock prescaler value can be set to divide PLLRAWCLK by 1 (TX_CLK_CTRL.PRESCALE_VAL = 0x0 or 0x1) through 255(TX_CLK_CTRL.PRESCAL_VAL = 0xFF). Though TXCLK and SYSCLK are both derived from PLLRAWCLK, TXCLK is asynchronous with respect to SYSCLK.

CAUTION

TXCLK must never be configured to be faster than SYSCLK/2.

32.3.2.3 Transmitting Frames

On the transmitter, the ping frame is the only frame that can be set up and transmitted without any further software or DMA intervention. Ping frames can be transmitted by any (or all) of the three sources: automatic ping timer, software, or external triggers.

Each available frame type can be sent multiple ways. Generically, the following steps must be executed before the frame is sent. These steps can be executed in any order before the start condition is set.

1. Configure the frame type
2. Set the frame tag
3. If the frame to be sent is a data frame:
   - Set the user data
   - Write to the data buffer
   - Set the word length if the frame is a software defined frame length
4. Set the start condition

Note

Transmit Frame Start Restriction:

A new frame transmission can be initiated by one of the methods selected in the TX_OPER_CTRL_LO.START_MODE bits. If there is already a PING frame transmission taking place, due to a hardware initiated PING timer, the new frame transmission begins as soon as the on-going PING transmission is completed.

Once a START of frame has been initiated, the next START of frame is recognized when the first frame has started transmitting the End-of-Frame (EOF) field. If a new START trigger arrives before the current transmission has reached the EOF field, the trigger is lost without a notification.

Note

There is no hardware check implemented to check whether the type field written by software is valid or not. If an invalid type is used and a frame transmission is initiated, the behavior is:

- The transmitted frame structure is exactly like an NWORD data frame. The size of the data frame is determined by the value in the TX_FRAME_CTRL.N_WORDS register.
- The frame type field of the transmitted data frame is transmitted as programmed. If this is received by an FSI receiver, a Type error is generated.

This mechanism can be used for force a Type error in a received frame for testing purposes.

The following sections describe the specific configuration for each frame type and start condition.

32.3.2.3.1 Software Triggered Frames

The most basic way to transmit a data frame is through software. Each step must be handled by the application. To send a data frame using software, the following steps must be executed. Steps 1-6 can be executed in any order before setting TX_FRAME_CTRL.START. Some fields may not need to be reconfigured for every transmission. The frame tag, user data, and frame type are sticky and are retransmitted in the subsequent frame unless modified by software.

1. Write the data to be transmitted to the next location of the transmit data buffer.
2. Set TX_FRAME_CTRL.FRAME_TYPE to the appropriate value for the type of frame to be transmitted.
3. Set TX_FRAME_CTRL.N_WORDS to 1 less than the number of words to be transmitted if TX_FRAME_CTRL.FRAME_TYPE is set to 0011, the frame type of the software-defined length data frame. That is, if 16 words are transmitted, N = 16, set TX_FRAME_CTRL.N_WORDS to 15.
4. When the frame is assembled before transmitting, the FSITX hardware calculates the CRC to be transmitted. If TX_OPER_CTRL_LO.SW_CRC is 1, the application can calculate a custom CRC value and then set TX_USER_CRC to the result.
5. Set TX_FRAME_TAG_UDATA.FRAME_TAG to the desired tag.
6. Set TX_FRAME_TAG_UDATA.USER_DATA to the desired user data.
7. Set TX_FRAME_CTRL.START to 1 to initiate the transmission of the data frame.

Once the frame transmission has started, the TX_FRAME_CTRL.START is cleared by hardware. To monitor if the frame has completed, the software can poll TX_EVT_STS.FRAME_DONE.

32.3.2.3.2 Externally Triggered Frames

The transmitter can transmit frames when triggered by an external source. See Section 32.2.6 for more information on the available external triggers.

To transmit frames using an external trigger, the application must follow the same procedure as described in Section 32.3.2.3.1. The only difference is that in Step 7, the start condition is automatically set when the external trigger condition is met rather than by software.

Note that by externally triggering frames, the frame information to be sent is pulled from the same registers described in the previous section. Because of this, it is possible to send any type of frame from an external trigger including ping, error, and data frames. Also, there is no hardware mechanism by which the FSI can determine if multiple triggers occur. The FSITX takes the data as is, and the application software makes sure that this data has been updated as necessary.

Using TX_EVT_STS fields either by polling or by interrupts, the application can populate or update the frame information to be sent in the next frame

32.3.2.3.3 Ping Frame Generation

Assuming the FSI transmitter has already been properly initialized, the following sequences can be used to configure and send ping frames.

32.3.2.3.3.1 Automatic Ping Frames

To generate periodic ping frames, the following steps must be followed:

1. Initialize the ping counter by writing 1 to TX_PING_CTRL.CNT_RST.
2. Set the desired ping tag to TX_PING_TAG.TAG.
3. Set the ping timer reference value to TX_PING_TO_REF.TO_REF.
4. Enable the ping timer by writing 1 to TX_PING_CTRL.TIMER_EN.

The ping timer is a free-running counter that counts up from 0. The current value of the ping timer counter is found in TX_PING_TO_CNT. When the current value of TX_PING_TO_CNT matches the reference value TX_PING_TO_REF.TO_REF, the TX_EVT_STS.PING_TRIGGERED is set. TX_PING_TO_CNT resets to 0 and resumes counting until the next match has occurred or the ping timer is halted by software (TX_PING_CTRL.TIMER_EN is set to 0).

32.3.2.3.3.2 Software Triggered Ping Frame

Software can also manually generate a ping frame. The process for sending a ping frame with software is very similar to sending the other types of frames. The following steps must be followed:

1. Set TX_FRAME_CTRL.FRAME_TYPE to 0000'b to denote that the frame being sent is a Ping Frame.
2. Set TX_FRAME_TAG_UDATA.FRAME_TAG to the desired value.
3. Write 1 to TX_FRAME_CTRL.START. This starts the transmission.

Once the frame transmission has started, the TX_FRAME_CTRL.START is cleared by hardware. To monitor if the frame has completed, the software can poll TX_EVT_STS.FRAME_DONE.

32.3.2.3.3.3 Externally Triggered Ping Frame

The last source for generating ping frames is an external trigger. One of up to 32 different triggers can be selected. See Section 32.2.6 for the list of input sources.

CAUTION

Ping frames can be triggered by both an external trigger source and the internal ping timer. If TX_PING_CTRL.EXT_TRIG_EN is set to 1, the external trigger source takes precedence and the ping timer is ignored.

32.3.2.3.4 Transmitting Frames with DMA

The FSI transmitter can send data that is continuously applied with the DMA. A DMA trigger is generated every time a data frame transmission is completed. This is concurrent with the FRAME_DONE signal that sets the TX_EVT_STS.FRAME_DONE flag.

To transmit continuous data with the DMA, some configurations need to be made on the transmitter:

First, set TX_DMA_CTRL.DMA_EVT_EN to 1. This allows the DMA trigger to propagate to the DMA module. Next, TX_OPER_CTRL_LO.START_MODE must be set to 0x2. The transmitter is now able to start a transmission using a software write to TX_FRAME_CTRL.START or TX_FRAME_TAG_UDATA..

The DMA must also be configured properly for the FSI to send the data. One way of using the DMA to continuously feed the transmit buffer is:

- Set up two DMA channels to be triggered by the same FSI transmitter and DMA trigger.
- Configure one channel to fill the transmit buffer.
- Configure the other channel to set the frame tag and user data fields
- Since the FSI transmit buffer is a 16-word circular buffer, make sure the DMA channel servicing the data buffer wraps the after 16 words are copied.

Note

Because the frame tag and user data must be written in to initiate the transmission of the frame, use two consecutive DMA channels. This makes sure that the DMA channels are always executed in sequence. The DMA channel servicing the data buffer must be the lower numbered channel and the tag/user data channel must be the next. For example, configure DMA channel 3 to service the data buffer, and configure DMA channel 4 to service the tag and user data.

32.3.2.4 Transmit Buffer Management

The FSI transmitter has a 16-word buffer that the FSI transmitter pulls data to transmit. This buffer is implemented as a circular buffer, not a FIFO, so some care must be taken to properly interpret buffer overrun and underrun, as well as the TX_BUF_PTR_STS register. These flags and pointers work under the assumption that the software or DMA is using the buffer as a circular buffer. This mode of operation is the only way in which the overrun, underrun, and pointer status are meaningful. If data is being sourced by the DMA and there is some other periodic trigger mechanism trying to initiate transfers, underrun becomes a critical error. If an underrun happens, it means that the buffer went out of sync. This would not only affect the current transfer, but all future transfers also could not be assured due to the ring buffer. Under such conditions, the underrun would need a soft reset to cleanly recover. Alternately, the software could manually stop the transmitting, reset the buffer pointers, clear the remaining error conditions, and then restart transmission. The software method involves a few steps, while the soft reset is a single action and assures a full reset of the control registers.

Due to the flexibility of the transmit buffer, it is possible for software to implement a simple ping-pong buffer, or to randomly load and send from any location of the buffer. If the buffer is used in this manner, error flags and status fields can be ignored without adversely affecting the transmitter capability. Additionally, the CURR_WORD_CNT is also invalid if used in this way. The application can set the buffer pointer manually by writing the 4-bit index to TX_BUF_PTR_LOAD. This forces the transmitter to start picking the data from the indicated location in the buffer.

32.3.2.5 CRC Submodule

The FSI transmitter can supply the CRC to the frame being transmitted through the embedded hardware CRC submodule or by supplying a user-defined value. This is controlled by setting TX_OPER_CTRL_LO.SW_CRC appropriately.

If hardware CRC generation is selected (TX_OPER_CTRL_LO.SW_CRC = 0, the default), the CRC is computed by hardware on the data and user data fields using the CRC polynomial 0x7 ($x^8 + x^2 + x + 1$). The transmitter module automatically computes the CRC on the data fields without user intervention when the frame is transmitted. For more information on how the CRC is generated by the CRC submodule, refer to Section 32.3.7.

If software CRC generation is selected (TX_OPER_CTRL_LO.SW_CRC = 1), the CRC must be computed by software and placed in the TX_USER_CRC register. The next frame to be transmitted uses the value placed in the TX_USER_CRC register in place of the CRC value generated by the hardware.

As the TX_USER_CRC register is software-programmable, the application may use this field as an extra data field for application-specific purposes. If TX_USER_CRC is used in this manner, the CRC detection on the receiver is not valid and must be ignored.

32.3.2.6 Conditions in Which the Transmitter Must Undergo a Soft Reset

Unlike the receiver, there are no detectable errors that require a soft reset. A buffer overrun or underrun interrupt can or cannot require a soft reset to resume proper operation. This determination is up to the application software. Refer to Section 32.3.2.4 for more information on the transmit buffer.

32.3.2.7 Reset

The entire transmitter module and all transmitter registers are reset by SYSRSn. The transmitter core is reset by SYSRSn or by writing a 1 to TX_MASTER_CTRL.CORE_RST.

A module reset causes the registers to be reset to their default state.

32.3.3 FSI Receiver Module

The receiver module interfaces to the FSI clock (RXCLK), and data lines (RXD0 and RXD1) after they pass through an optional programmable delay line. The receiver core handles the data framing, CRC computation, and frame-related error checking. The receiver bit clock and state machine are run by the RXCLK input, which is asynchronous to the device system clock.

The receiver control registers allow the CPU (or the CLA) to program, control, and monitor the operation of the FSI receiver. The receive data buffer is accessible by the CPU, CLA, and the DMA.

The receiver core has the following features:

- 16-word data buffer
- Multiple supported frame types
- Ping frame watchdog
- Frame watchdog
- CRC calculation and comparison in hardware
- ECC detection
- Programmable delay line control on incoming signals
- DMA support
- CLA task triggering
- FSI-SPI compatibility mode

Figure 32-5 provides a high-level overview of the internal modules present in the FSI receiver. Figure 32-6 shows a view of the FSI receiver core submodule. Not all data paths and internal connections are shown.

The following sections describe the various aspects of the FSI receiver module.

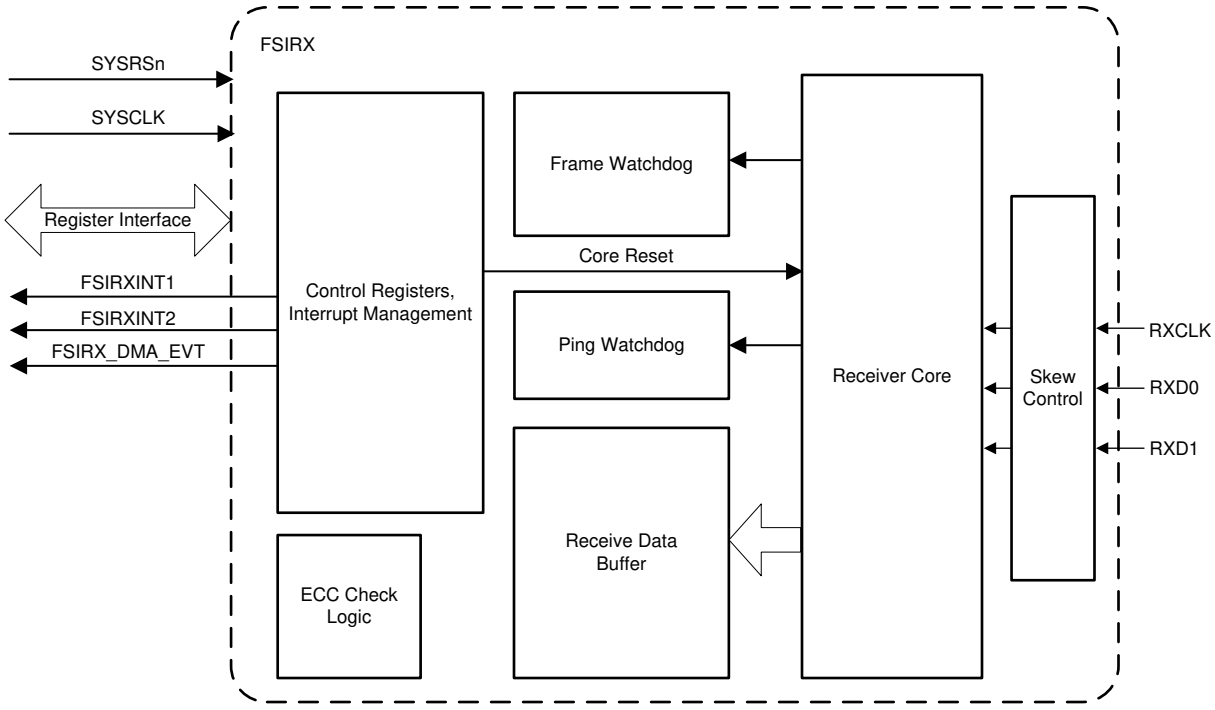


Figure 32-5. FSI Receiver Block Diagram

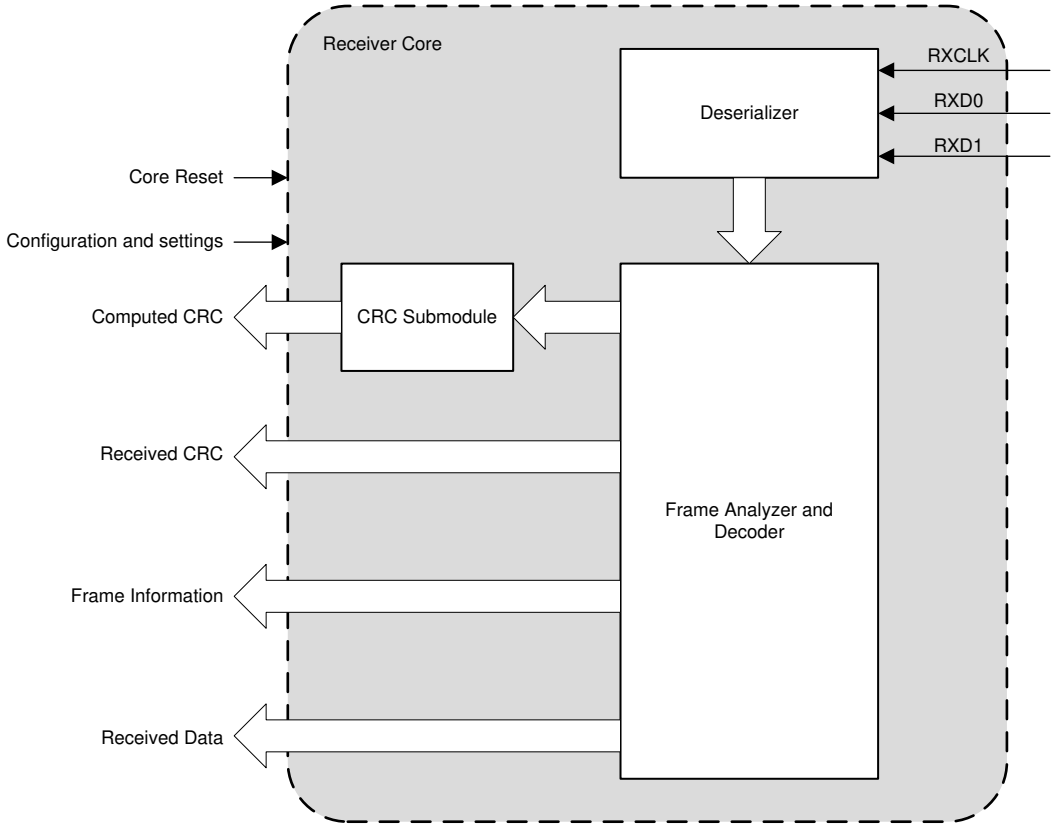


Figure 32-6. FSI Receiver Core Block Diagram

32.3.3.1 Initialization

On the first initialization or after a module reset following any frame error, the receiver module asserts and releases the receiver core reset bit (RX_MASTER_CTRL.CORE_RST) prior to any other initialization. Once the receiver module is initialized, the following steps are executed:

1. If required, assign interrupt sources to the necessary interrupt line.
2. If required, configure the ping watchdog to periodically check for an active link to the transmitter. See Section 32.3.3.4 for configuration details.
3. If required, configure the frame watchdog to make sure that each frame is received within a predetermined window. See Section 32.3.3.5 for configuration details.
4. Initialize the receive buffer pointer by writing to the RX_BUF_PTR_LOAD register. Received data is placed into the buffer starting with the address loaded in this register.
5. Make sure all errors and flags have been cleared from the RX_EVT_STS register.

At this point the receiver is ready to receive any incoming frames. Software can now either poll on the RX_EVT_STS register for various conditions. For example, when the RX_EVT_STS.FRAME_DONE and no other flags are set, the receiver has successfully received a frame without error.

Next, the application configures the various features such as the ping and frame watchdogs, DMA, external triggering, and so on. These features are described in subsequent sections. The receiver module is now ready to synchronize with the transmitter then begin reception. The synchronization sequence is described in Section 32.4.1.

32.3.3.2 FSI_RX Clocking

The receiver module registers and control logic are clocked by the device system clock (SYSCLK). The receiver state machine is clocked by the receiver input clock pin (RXCLK).

CAUTION

RXCLK must never be faster than SYSCLK.

32.3.3.3 Receiving Frames

Once the receiver has been properly configured and synchronized, incoming messages are handled as described below. Note that there is no equivalent to a chip-select signal to gate incoming data. Every valid clock edge latches data into the receiver.

The header information of the received frame is placed in their respective register fields.

- RX_FRAME_INFO.FRAME_TYPE contains the received frame type.
- RX_FRAME_TAG_UDATA.FRAME_TAG contains the received frame tag.
- RX_FRAME_TAG_UDATA.USER_DATA contains the received user data.

If any error conditions occur during reception such as a CRC mismatch, frame error, frame timeout, buffer overrun, or ping watchdog timeout, the corresponding flag is set in the RX_EVT_STS register.

Note

If at any point during operation a frame error occurs, the receiver module must be reset and re-synchronized with the transmitter before the next frame can be successfully received. The follow errors are classified as frame errors:

- Type error
- CRC error
- End of frame error

32.3.3.3.1 Receiving Frames with DMA

The FSI receiver can continuously receive data and move the data from the receiver buffer with the DMA. A DMA trigger is generated every time a data frame has been received. This is concurrent with the FRAME_DONE signal that sets the RX_EVT_STS.FRAME_DONE flag. To receive continuous data with the DMA, some configurations need to be made on the receiver.

First, set RX_DMA_CTRL.DMA_EVT_EN to 1. This allows the DMA trigger to propagate to the DMA module. The receiver is now able to trigger a DMA event upon the reception of a data frame.

The DMA must also be configured properly for the FSI to receive the data. One way for using the receiver to continuously feed the DMA is:

- Set up two DMA channels to be triggered by the FSI Receiver DMA Trigger.
- Configure one DMA channel to copy data from the receive buffer to a larger data buffer.
- Configure the next DMA channel to copy the received frame tag and user data to another data buffer.
- Since the FSI receive buffer is a 16-word circular buffer, make sure the DMA channel servicing the data buffer wraps after 16 words are copied.

Unlike the transmitter, there is no requirement to have the DMA channel which is handling the data buffer, execute before the DMA channel handling the received tag and user data.

32.3.3.4 Ping Frame Watchdog

The ping frame watchdog is a hardware-enabled automatic error detection of the connection status to the transmitter. This watchdog monitors the time elapsed between ping frames. If the transmitter has been set up to

periodically send out a ping frame, the receiver can be set up to monitor whether this frame has been received within a specified amount of time. If the time between ping frames has exceeded the programmed number of clock cycles, an event is triggered that can generate an interrupt or be monitored by software.

This watchdog has a dedicated counter that is reset and restarted upon the successful reception of a ping frame. The watchdog counter is incremented at the rate of SYSCLK. Optionally, the watchdog can be configured to be reset upon the successful reception of any frame. This option allows the receiver to monitor for any successful frame to indicate that the connection is still alive and the transmitter is still functioning as expected.

To configure the ping frame watchdog for operation:

1. Reset the ping watchdog counter by setting RX_PING_WD_CTRL.PING_WD_RST to 1 and then subsequently clearing the bit to 0.
2. Set RX_OPER_CTRL.PING_WD_RST_MODE to the desired watchdog reset event, set to 0 for ping frames only or set to 1 for any frame.
3. Set RX_PING_WD_REF to the maximum time between frames. Add 10 additional SYSCLK cycles to account for clock synchronization.
4. Enable the ping watchdog by setting RX_PING_WD_CTRL.PING_EN to 1.

The ping watchdog is now enabled and can now monitor for ping frames.

If the RX_PING_WD_CNT value reaches the value programmed in RX_PING_WD_REF, the RX_EVT_STS.PING_WD_TO flag is set. If configured, an interrupt can be generated on this event.

32.3.3.5 Frame Watchdog

The frame watchdog is an additional feature the receiver can use to monitor for any error conditions. This dedicated watchdog monitors the duration for a single frame to be received. The watchdog starts incrementing at the time the receiver detects a proper start of frame condition. If the end of frame condition is not detected within the expected number of SYSCLK cycles, the frame watchdog is triggered that can generate an interrupt or be monitored by software.

This watchdog is automatically started and stopped at the start-of-frame and end-of-frame conditions, respectively. The frame watchdog is connected to SYSCLK.

To configure the frame watchdog for operation:

1. Reset the frame watchdog counter by setting RX_FRAME_WD_CTRL. FRAME_WD_CNT_RST to 1 and then subsequently clearing the bit to 0.
2. Set RX_FRAME_WD_REF.FRAME_WD_REF to the maximum number of SYSCLK cycles expected to be in the longest frame that can be received. Add an additional 10 SYSCLK cycles to account for clock synchronization.
3. Enable the frame watchdog by setting RX_FRAME_WD_CTRL.FRAME_WD_CNT_EN to 1.

The frame watchdog is now enabled and can detect a failed frame.

If the RX_FRAME_WD_CNT reaches the value programmed in RX_FRAME_WD_REF, the RX_EVT_STS.FRAME_WD_TO flag is set. If enabled, an interrupt can be generated on this event.

If the frame watchdog interrupt ever occurs, the receiver core is in an invalid state to receive a new transmission. The only way to recover from a frame watchdog time out is to undergo a soft reset, and subsequently resynchronizing with the transmitter.

32.3.3.6 Delay Line Control

The receiver module has a programmable delay line on each of the external signal inputs: RXCLK, RXD0, and RXD1. The delay elements introduce delays on the respective lines. This is to facilitate adjustment for signal delays introduced by system level components such as signal buffers, ferrite beads, isolators, and so on, or board delays such as uneven trace lengths, long cable length, and so on. The length of the delay is controlled by setting the RX_DLY_LINE_CTRL register values for each line. By default, no delay is introduced by the delay line elements. The delay values must only be adjusted while the FSIRX is held in soft reset, making sure that there are no active transmissions during this process. Figure 32-7 shows a representation of the delay line circuitry for the input signals. The implementation for RXCLK, RXD0, and RXD1 are replicas of this diagram. All circuits behave similarly.

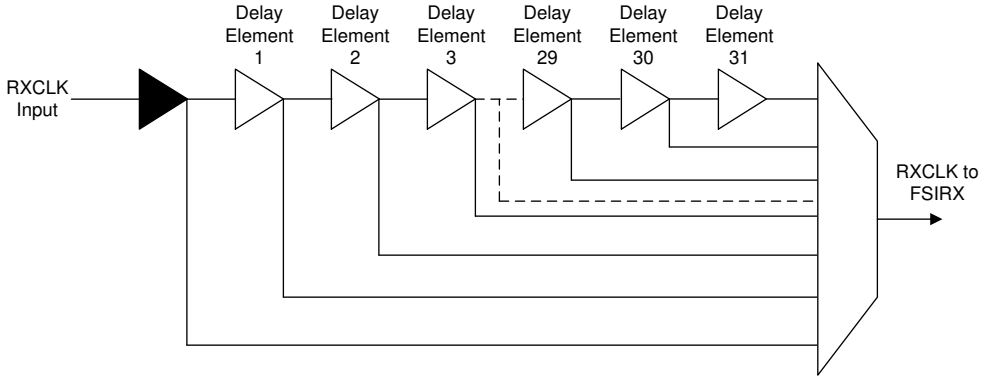


Figure 32-7. Delay Line Control Circuit

For more information on skew compensation, refer to *Fast Serial Interface (FSI) Skew Compensation*.

32.3.3.7 Buffer Management

The FSI receiver has a 16-word buffer that the data is copied to when the data has been received. This buffer is implemented as a circular buffer, not a FIFO, so some care must be taken to properly interpret buffer overrun and underrun as well as the RX_BUF_PTR_STS register. These flags and pointers work under the assumption that the software or DMA is using the buffer as a circular buffer. If the receiver state machine enters into an erroneous state, there is no way for software to cleanly handle this because there is no specified receive clock. For the receiver to detect a clean resynchronization, the state machine needs to be operational and not in the error state. The only way to recover from the error state is to reset the entire receiver module. For overrun and underrun, the receiver can no longer verify that values in the buffer are valid. As such, the best way to recover is to reset the FSI and resynchronize with the transmitter.

Due to the flexibility of the receive buffer, it is possible for software to implement a simple ping-pong buffer, or to randomly receive and read from any location of the buffer. If the buffer is used in this manner, these flags and status fields can be ignored without adversely affecting the receiver capability. Additionally, the CURR_WORD_CNT is also invalid if used in this way. The application can set the buffer pointer manually by writing the 4-bit index to RX_BUF_PTR_LOAD. This forces the receiver to start storing the received data starting at the indicated location in the buffer.

32.3.3.8 CRC Submodule

The receive module automatically calculates the CRC on the incoming data. The received CRC value is placed into RX_CRC_INFO.RX_CRC. The CRC value calculated by hardware on the received data is placed into RX_CRC_INFO.CALC_CRC. These values are compared by hardware and RX_EVT_STS.CRC_ERROR is set if there is a mismatch. The receiver can generate an interrupt based on RX_EVT_STS.CRC_ERROR if enabled.

Since the CRC is only used in data frames, the values found in RX_CRC_INFO.RX_CRC and RX_CRC_INFO.CALC_CRC are undefined during ping and error frames.

For more information on how the CRC is calculated, refer to Section 32.3.7.

If the transmitting module is sending a software-defined CRC value (FSITX.TX_OPER_CTRL_LO.SW_CRC = 1), the receiver module triggers a CRC error event if the received value does not match the hardware-calculated value. As this is an application-level decision, the FSIRX can safely disregard the CRC error event. Application software needs to calculate and verify the incoming CRC using the same custom algorithm used on the transmitter and act appropriately.

The CRC field can also be used as an application-specific value, not a CRC. The application can use the RX_CRC_INFO.RX_CRC as required. All CRC errors and flags can be ignored in this situation.

32.3.3.9 Using the Zero Bits of the Receiver Tag Registers

The receiver tag registers (receiver frame tag and user data (RX_FRAME_TAG_UDATA) register and receiver ping tag (RX_PING_TAG) register)) have the least-significant bit set to 0. The actual received tag is in the bit positions 4:1. The reason for this is to facilitate user software to create a table of functions that can be called depending on the tag value. A function pointer needs a 32-bit storage space and, hence, each successive pointer is offset by 2. If the first pointer is at address x, then the second pointer is at address x + 2, the third at address x + 4, and so on. By keeping the LSB to 0, the five bits of the tag register (bits 4:0) can now be directly used as an index into a table of function pointers.

32.3.3.10 Conditions in Which the Receiver Must Undergo a Soft Reset

The receiver receives data on every clock edge. While there are specific patterns that determine the a start of a frame, and denote the end of a frame, these patterns are able to occur at any point during normal operation inside of the frame. If there ever is a point at which the receiver fails to detect a successful frame, the module must be reset to make sure that subsequent frames are received properly.

When any of the following errors occur in a received frame, the receiver can be required to be reset and resynchronized with the transmitter:

- Frame type error
- End of frame error
- Ping frame watchdog timeout
- Frame watchdog timeout
- Receiver in an invalid state due to noisy clock

The receiver core status (RX_VIS_1.RX_CORE_STS) can be monitored to determine if the receiver core has entered into an error state requiring a soft reset to resume communication. Incorrect frame type and end of frame errors always cause this bit to become set. A soft reset is required in these cases. A frame watchdog timeout always requires a reset due to the fact that the receiver state machine is still expecting more information when the watchdog timed out. RX_CORE_STS can be used to determine if a noise event was the cause of the failed frame. The ping frame watchdog also does not cause RX_CORE_STS to be set. Similar to the frame watchdog, a corrupt receiver may not be the reason for the ping frame to have timed out. The transmitter may have gone offline and never sent a ping frame. Alternately, during idle time, a noise event may have occurred, thereby putting the receiver into a corrupt state. As the receiver is able to detect this during the ping frame watchdog timeout interrupt handler, this type of event is not lost and the application can act appropriately.

As the receiver is clocked by RXCLK, not SYSCLK, a noisy clock or data line can cause some internal design constraints to be violated, putting the receiver core logic into undefined states. Make sure that the clock and data lines satisfy the Electrical Characteristics and timing requirements of the FSI module found in the device data sheet. Failure to do so can cause the receiver state machine to go into an unrecoverable error state. The receiver can only be recovered by undergoing a soft reset. To determine the state of the receiver core after an unexpected frame error, the application must check the receiver core status bit.

In addition to the above errors, buffer overrun or underrun can warrant a soft reset to resynchronize with the local application software. Refer to Section 32.3.3.8 for more information on the receive buffers. The requirement of resetting the receiver due to overrun or underrun is up to the application.

After the receiver has been placed into soft reset, the application must notify the other device's transmitter to begin a new synchronization phase. The simplest way to achieve this is through a ping or error frame sent with a designated tag. If the application is not using the FSITX on the device with the detected error, some other method must be established. The other device must stop transmitting and begin a new synchronization phase.

32.3.3.11 FSI_RX Reset

The receiver module and the registers are reset by SYSRSn. The receiver core is reset by SYSRSn or by writing a 1 to RX_MASTER_CTRL.CORE_RST.

A module reset causes the registers to be reset to their default state. After a module reset, the receiver module must be re-initialized and the data link re-established.

32.3.4 Frame Format

The FSI module transmits and receives information in frames. Each frame contains multiple phases where different information can be found. The number of phases as well as the total length of the frame varies depending on the frame type being transmitted. Frames can be as short as 16-bits long for a ping or error frame or 288-bits long for a 16-word data frame.

In normal transmission mode, there are four preamble clock edges before the start of the frame and four post-frame clock edges (postamble). Data is transmitted on both edges of the clock (double data rate). The basic frame structure is shown in Table 32-4. Each phase of the frame (such as start-of-frame, frame type, and so on) is transmitted with the most-significant bit first. Table 32-4 describes the basic frame structure used by the FSI and adapted according to which frame type is transmitted.

Table 32-4. Basic Frame Structure

Idle State	Preamble	Start of Frame	Frame Type	User Data	Data Words	CRC Byte	Frame Tag	End of Frame	Postamble	Idle State
	1111	1001	4 bits	8 bits	1-16 words	8 bits	4 bits	0110	1111	

The FSI also supports a FSI-SPI compatibility mode. The SPI compatible frame structure is similar to a standard FSI frame, but there are differences. Refer to Section 32.3.12 for more information on how to configure and use the FSI-SPI compatibility mode.

Note

One word of the FSI refers to 16 bits.

The terms “frame” and “packet” can be used interchangeably to describe the signaling format of the FSI.

32.3.4.1 FSI Frame Phases

The different phases of the frame structure are described in detail.

- **Idle State:** During the idle state, the clock and data lines are driven high, the inactive state.
- **Preamble:** The preamble phase contains four clock edges (or two complete clock pulses) with the data signals held in the high state. These clock edges serve to flush the receiver logic and prepare the receiver logic for receiving a new frame. This phase is not present in SPI compatibility mode.
- **Start of Frame:** The start of frame phase contains two clock pulses with four bits, 1001, transmitted on the data lines.
- **Frame Type:** The frame type phase contains two clock pulses with the 4-bit frame type code being transmitted on the data lines. The different frame types are described in detail in Section 32.3.4.2. The transmitter must set the TX_FRAME_CTRL.FRAME_TYPE field before transmitting a frame. The received frame type is stored in the RX_FRAME_INFO.FRAME_TYPE.
- **User Data:** The user data phase contains a fully user-configurable data field. There are no restrictions on how this field is used. This phase is only available in data frames. The user data to be transmitted is set by writing to TX_FRAME_TAG_UDATA.USER_DATA. The received user data is stored in RX_FRAME_TAG_UDATA.USER_DATA.
- **Data:** The data phase contains the data that is being transmitted. The data is pulled from the transmit buffer of the transmitter and is placed in the receive buffer of the receiver. Word 0 is transmitted first. This phase is only present in data frames. Depending on the type of frame transmitted, this can contain anywhere between 1 and 16 words depending on the frame type selected. More information on data frames is found in Section 32.3.4.2.3.
- **CRC Byte:** The CRC byte contains the CRC of the transmitted data. The value present in this phase can be sourced from either hardware or software based on the TX_OPER_CTRL_LO.SW_CRC bit. Refer to the module-specific section of the CRC Submodule for more information on the CRC is generated or used, for the transmitter and receiver modules respectively. The CRC byte is only present in data frames.
- **Frame Tag:** The frame tag contains the 4-bit user-defined frame tag. There are no restrictions on how this field is used in an application. The transmitter supplies this tag into the TX_FRAME_TAG_UDATA.FRAME_TAG bits for data frames. Ping frames use the tag defined in TX_PING_TAG.TAG. The receiver can access the received frame tag in RX_FRAME_TAG_UDATA.FRAME_TAG.
- **End of Frame:** The end of frame contains four clock edges with four bits, 0110, transmitted on the data lines.
- **Postamble:** The postamble contains four additional clock edges with the data lines held in the high state. After the postamble, the clock and data lines are driven high, their inactive state. This phase is not present in FSI-SPI compatibility mode.

32.3.4.2 Frame Types

The FSI hardware can generate and handle many predefined frame types. The different frame types can be used by the application to signal different types of events or convey different information to the receiver. The different frame types influence which phases and data fields to include in the transmitted frames.

Table 32-5 provides a short overview of the different frame types used by the FSI. Each frame type is described in more detail in the following subsections.

Table 32-5. Frame Types and Their 4-bit Codes

Frame Type	4-bit Frame Code	Description
PING	0000	This is the ping frame that can be sent either by software or automatically by hardware.
ERROR	1111	This must be used typically during error conditions or any condition where one side wants to signal the other side for attention. However, the user software can use this for any purpose.
DATA_1_WORD	0100	1 word data packet (16 bits of data)
DATA_2_WORD	0101	2 word data packet (32 bits of data)
DATA_4_WORD	0110	4 word data packet (64 bits of data)
DATA_6_WORD	0111	6 word data packet (96 bits of data)
DATA_N_WORD	0011	N(1-16) word data packet where software has programmed the number of the data words in a designated register. Both transmitter and receiver modules must have the same value programmed.
Reserved	0001, 0010, and 1000-1110	Reserved

32.3.4.2.1 Ping Frames

Ping frames are one of the most basic frames that can be generated by the FSI. Table 32-6 shows the structure of the ping frames.

Table 32-6. Ping Frame

Idle State	Preamble	SOF	Frame Type	Frame Tag	EOF	Postamble	Idle State
	1111	1001	0000	xxxx	0110	1111	

The ping frame type is always 0000. The frame tag is defined by the application. Separate frame tags exist for timer and software initiated ping frames. No data or CRC is transmitted in a ping frame.

The main purpose of the ping frame is to periodically send a notification to the receiver to make sure an active connection between the transmitter and receiver. The transmitter and receiver cores implement different features to allow the ping frame to operate as a line break detect feature.

On the transmitter, the ping frame is the only frame that can be set up and transmitted without any further software or DMA intervention. Ping frames can be transmitted by any (or all) of the three sources: automatic ping timer, software, or external triggers. See Section 32.3.2.3.3 for information on how the transmitter configures and sends the ping frames.

The receiver has a ping watchdog that can detect if a ping frame has not been received in a predetermined window. This allows the receiver to know if the connection between the receiver and the transmitter has been broken. See Section 32.3.3.4 for information on how the receiver handles ping frames.

32.3.4.2.2 Error Frames

Error frames are similar to ping frames in that there are no data fields transmitted. Despite the naming of this frame as an “error frame,” the usage of it is up to the application, as no restrictions are placed on how and when this type of frame is transmitted. Table 32-7 shows the structure of an error frame.

Table 32-7. Error Frame

Idle State	Preamble	SOF	Frame Type	Frame Tag	EOF	Postamble	Idle State
	1111	1001	1111	xxxx	0110	1111	

The structure of the error frame is the same as a ping frame. No data or CRC values are transmitted. The frame type is 1111 for all error frames, and the frame tag is defined by software in the TX_FRAME_TAG_UDATA register.

The receiver can detect if an error frame has been received based on the frame type field. Because of this, the receiver can read the incoming frame tag from the RX_FRAME_TAG_UDATA register and act on up to 16 different conditions.

32.3.4.2.3 Data Frames

Data frames are the most complex frames. As the name indicates, these frames are used to transfer data. Table 32-8 shows the general structure of data frames.

Table 32-8. Data Frame

Idle State	Preamble	SOF	Frame Type	User Data	Data Words	CRC Byte	Frame Tag	EOF	Postamble	Idle State
	1111	1001	0xxx	xxxx xxxx	1-16 words	xxxx xxxx	xxxx	0110	1111	

The frame type field reflects the 4-bit code of the frame type. A list of frame types can be seen in Table 32-5. The number of the data words transmitted is determined by the frame type chosen.

There are four fixed-length data frames supported by the frame type: 1 word, 2 words, 4 words, and 6 words.

Additionally, there is a user-defined data length frame type where the number of data words is fixed by software. Anywhere from 1 to 16 words can be transmitted in this frame type. This length must be configured in the N_WORDS field of the transmitter’s TX_FRAME_CTRL register and receiver’s RX_OPER_CTRL register.

32.3.4.3 Multi-Lane Transmission

The FSI is capable of transmitting and receiving data on two parallel data lines. When enabled, data bits are split between the data lines while the start of frame, frame type, frame tag, and end of frame fields are identical and complete on each line. The user data, data, and CRC fields are split between the data lines. Starting with the most-significant bit, the odd-numbered bits appear on D0 and even-numbered bits appear on D1.

In the following example, assume the following:

8-bit user data: u7u6u5u4u3u2u1u0

16-bit data: d15d14d13d12…d1d0

8-bit CRC: c7c6c5c4c3c2c1c0

Table 32-9. Multi-Lane Frame Format

Idle State	Preamble	SOF	Frame Type	User Data	Data Words	CRC Byte	Frame Tag	EOF	Postamble	Idle State
TXD0	1111	1001	0011	$u^7u^5u^3u^1$	$d^{15}d^{13}…d^1$	$c^7c^5c^3c^1$	xxxx	0110	1111	
TXD1	1111	1001	0011	$u^6u^4u^2u^0$	$d^{14}d^{12}…d^0$	$c^6c^4c^2c^0$	xxxx	0110	1111	

32.3.5 Flush Sequence

Every time there is a soft reset of the receiver, the receiver requires a flush sequence from the transmitter before the receiver can receive and decode frames. The receiver core has an asynchronous reset mechanism that allows the receive module to be reset even in the absence of the receive clocks. However, due to the design, this reset is released synchronous to the receive clock (RXCLK). Thus, the receiver requires five full clock pulses to be able to come out of reset. Sending the flush pattern makes sure that these clock edges are received and any subsequent frames sent to the receiver are correctly interpreted.

The flush sequence consists of a single toggle on both of the data lines as well as five consecutive pulses on the clock line.

If the FSI receiver is receiving data from a standard SPI, a data word of 0xFFFF from the SPI has the same effect as a flush sequence.

Figure 32-8 shows a sample plot of the flush sequence.

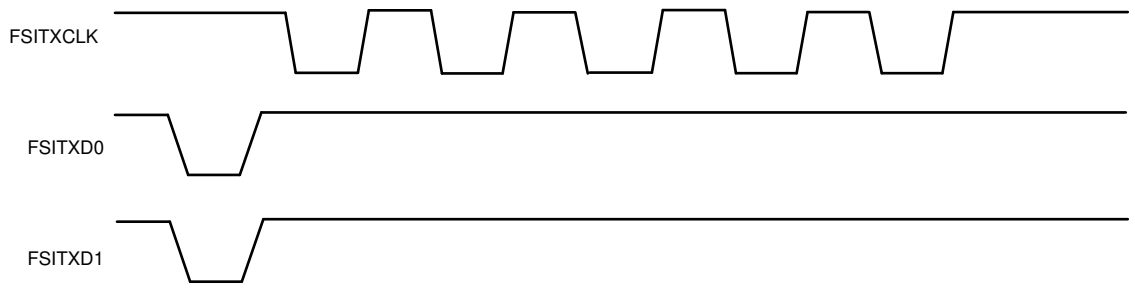


Figure 32-8. Flush Sequence Signals

32.3.6 Internal Loopback

The transmitter and receiver cores can be connected together internally to allow for development and debug. This is achieved by setting RX_MASTER_CTRL.INT_LOOPBACK to 1. Internal loopback routes the signals from the corresponding transmitter to the appropriate receiver pin. No configuration needs to be done in the transmitter.

Figure 32-9 shows the signal connections with internal loopback.

In this device, there are two FSI transmitter cores (A and B), and eight receiver cores (A, B, C, D, E, F, G, and H). When using loopback mode, FSITXA can be used in the loopback mode with FSIRXA, FSIRXB, FSIRXC, and FSIRXD. At the same time, FSITXB can be used in loopback mode with FSIRXE, FSIRXF, FSIRXG, and FSIRXH.

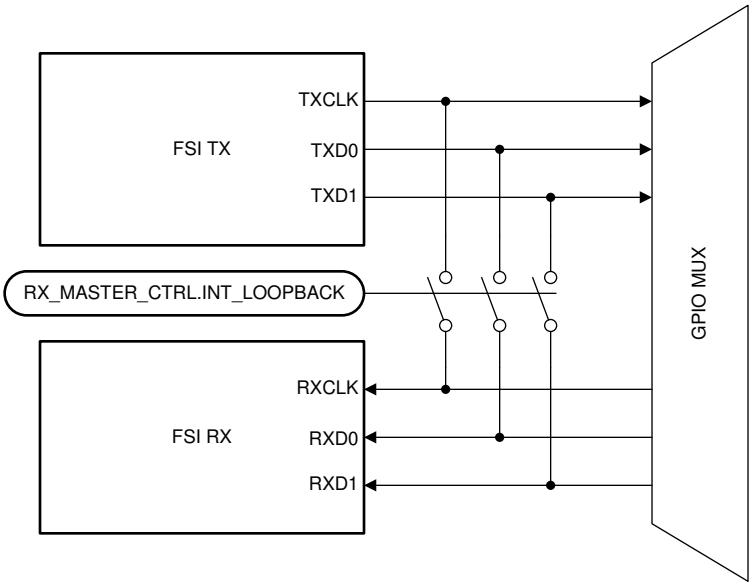


Figure 32-9. FSI with Internal Loopback

32.3.7 CRC Generation

The FSI uses CRC-8 with the polynomial 0x07 for the internal hardware CRC generation. This polynomial is also represented as x^8+x^2+x+1.

For example, for a 2-word data packet the following calculation occurs:

Data-1 = 0x4433

Data-0 = 0x2211

User Data = 0xAA

The CRC is computed with the bytes being taken in the following order (first to last):

0xAA – Byte 0, User Data

0x11 – Byte 1, Data-0, Least-significant byte

0x22 – Byte 2, Data-0, Most-significant byte

0x33 – Byte 3, Data-1, Least-significant byte

0x44 – Byte 4, Data-1, Most-significant byte

32.3.8 ECC Module

The FSI module comes with a 16-bit or 32-bit ECC computation module in both the transmitter and receiver. Use of this module is optional.

Note that the ECC is independent and unrelated to the hardware CRC computation module present in both the transmitter and receiver cores.

The following example shows a scenario in which the application requires ECC be calculated and transmitted on a 2-word data frame.

In the FSITX module:

1. Configure the ECC module for 32-bit data by setting TX_OPER_CTRL_HI.ECC_SEL to 1.
2. Write the data to the TX_ECC_DATA register as well as the transmit buffer.
3. Read TX_ECC_VAL Register. This register contains the 8-bit ECC value calculated on the data.
4. Copy the 8-bit data from TX_ECC_VAL to TX_FRAME_TAG_UDATA.USER_DATA.
5. Set the Start Condition to begin the transmission.

The reverse process is followed on the FSIRX module. Once the data frame is received, user software can do the following:

1. Copy the data from the receive buffer to the RX_ECC_DATA register.
2. Copy the received user data that contains the transmitted ECC value from RX_FRAME_TAG_UDATA.USER_DATA to the RX_ECC_VAL register.
3. Read the RX_ECC_LOG register. This contains the result of the ECC computation using the RX_ECC_DATA and RX_ECC_VAL registers.
   a. If no ECC errors were detected, RX_ECC_LOG is 0. The correct data is available in RX_ECC_SEC_DATA.
   b. If a single bit error was detected, RX_ECC_LOG.SBE is 1. The autocorrected data is available in RX_ECC_SEC_DATA.
   c. If multiple bit errors occurred, RX_ECC_LOG.MBE is 1. The data in RX_ECC_SEC_DATA is invalid and must not be used.

Using a 2-word data frame plus using the user data for the ECC is one possible implementation for ECC detection. Another option is to use a larger data frame a allocate one of the data words to be the ECC value.

32.3.9 Tag Matching

The FSI receiver core has the capability of generating an interrupt when the received data or ping frame's tag matches the reference tag in RX_FRAME_TAG_CMP or RX_PING_TAG_CMP, respectively.

Each tag compare register allows the user to not only select a reference tag (TAG_REF) but also set up a mask (TAG_MASK) to ignore specified bits in the reference tag. For tag matching to be enabled, the CMP_EN bit must be set. When the tag compare is enabled and a successful match occurs, the PING_TAG_MATCH, the DATA_TAG_MATCH or the ERROR_TAG_MATCH in RX_EVT_ERR_STATUS is set. The corresponding match register status can be cleared by writing to RX_EVT_ERR_CLEAR. Also, if required to set the match register status in software, the user must do so by writing to the RX_EVT_ERR_SET register.

The tag matching scheme is not a filtering scheme. Tag matching is only a notification scheme to alert the user when a specific tag is detected in a data or ping frame. Both RX_INTR_EVT_CTRL_1 and RX_INTR_EVT_CTRL_2 interrupts can be set up to generate an interrupt when a tag match event occurs.

Another feature used when tag matching is enabled is the broadcast feature. The broadcast feature can be enabled by setting the BRDCST_EN in RX_FRAME_TAG_CMP or RX_PING_TAG_CMP. When broadcast mode is enabled, the third bit of the tag is treated as a broadcast bit. If the received tag has a third bit set, then the tag is treated as a match regardless of the other tag bit comparisons. Note that a match caused by TAG_REF and TAG_MASK is also considered a match.

Note that the tag matching scheme is not a filtering scheme. For example, if tag matching is enabled and a frame is received with a non-matching tag, the frame is still saved in the buffer and the corresponding event status bits (FRAME_DONE, DATA_FRAME_RCVD) is set.

Tag matching must be used in a TDM configuration as described in Section 32.3.10. However, tag matching can be used in single slave configurations as needed.

32.3.10 TDM Configurations

The FSI module in this device supports multi-slave TDM configurations, whereas a single master device can control multiple slave devices. To use the FSI module in the multi-slave TDM configuration described, the slave device must utilize tag matching and the CLB module.

This multi-slave TDM configuration is supported in the FSI module through time-division multiplexing. When TDM is enabled, each slave device must also have tag matching enabled. Figure 32-10 shows a scheme where a single master device is communicating with multiple slave devices. All FSI receive modules in the slave devices are directly connected to the master device transmit module. The transmit modules of the slave devices are chained serially such that each transmit module is connected to the next slave device and the last slave device output connects to the master device receive module. Each slave device decides, based on the received frame's tag, whether to transmit their own data or to enter bypass mode where the previous slave device transmit module directly connects to the next slave device. This is done by using the FSI transmit module TDM_IN.

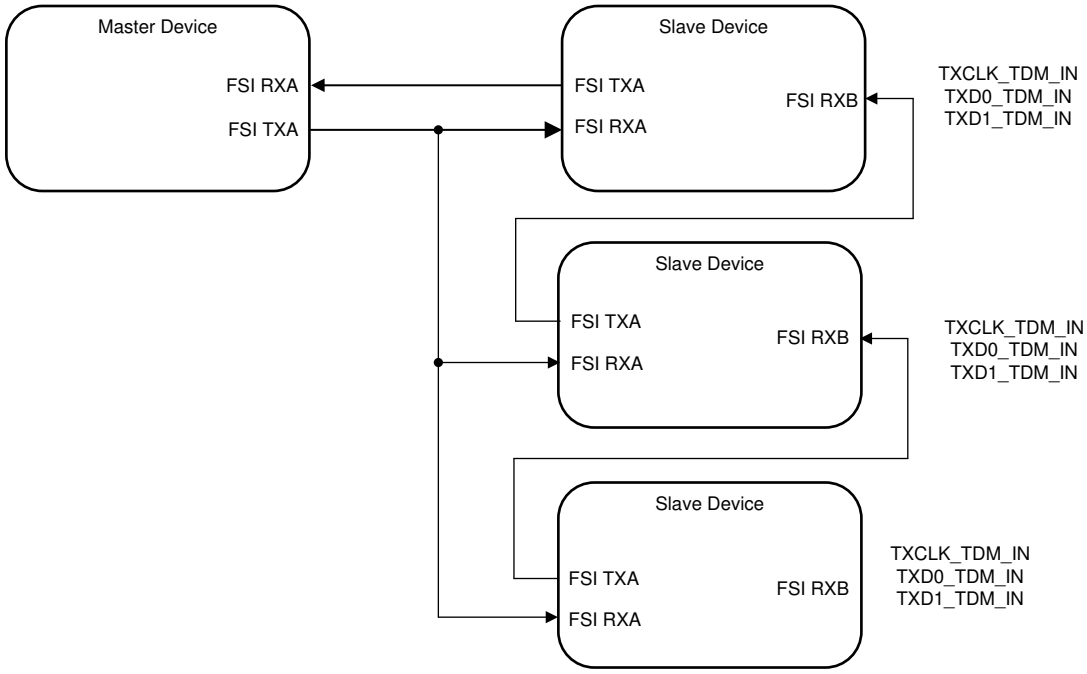


Figure 32-10. FSI Multi-Slave TDM Configuration

When an FSI transmitter module is used in TDM mode, TXCLK_TDM_IN, TXD0_TDM_IN and TXD1_TDM_IN pins are used if the transmitter is required to enter bypass mode. Figure 32-11 shows how the FSI module operates when in multi-slave TDM mode.

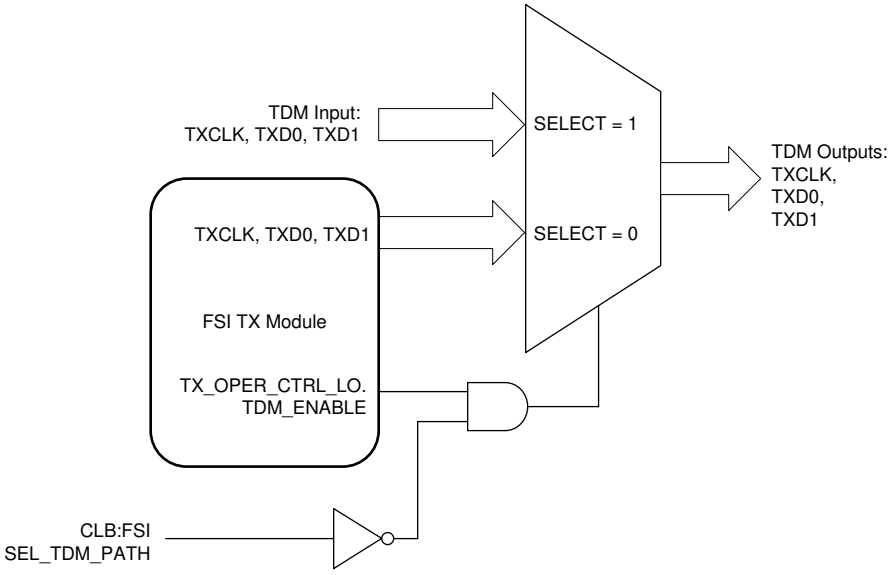


Figure 32-11. FSI Transmitter Multi-Slave TDM Multiplexing

The SEL_TDM_PATH signal is sourced from the CLB module. The CLB module also generates the transmit trigger for the FSI transmitter. The CLB module must be configured to decide when to generate the FSI transmit trigger based on the status of the data, ping, and frame tag match generated by the FSI receiver module. The FSITX module must be configured to transmit on an external trigger and the corresponding CLB trigger input must be selected. In a broadcast scenario (FSI tag match notifies all slave devices that a match has occurred), the CLB module inside each slave device generates a trigger and SEL_TDM_PATH signal. The main key here is that the trigger and the SEL_TDM_PATH signal must be generated at a different time interval in a non-overlapping manner. Figure 32-12 shows an example of FSI transmit triggers and the multi-slave TDM SEL_TDM_PATH signals generated by the RX_TRIGx signal or the CLB module of the slave devices in a broadcast scenario.

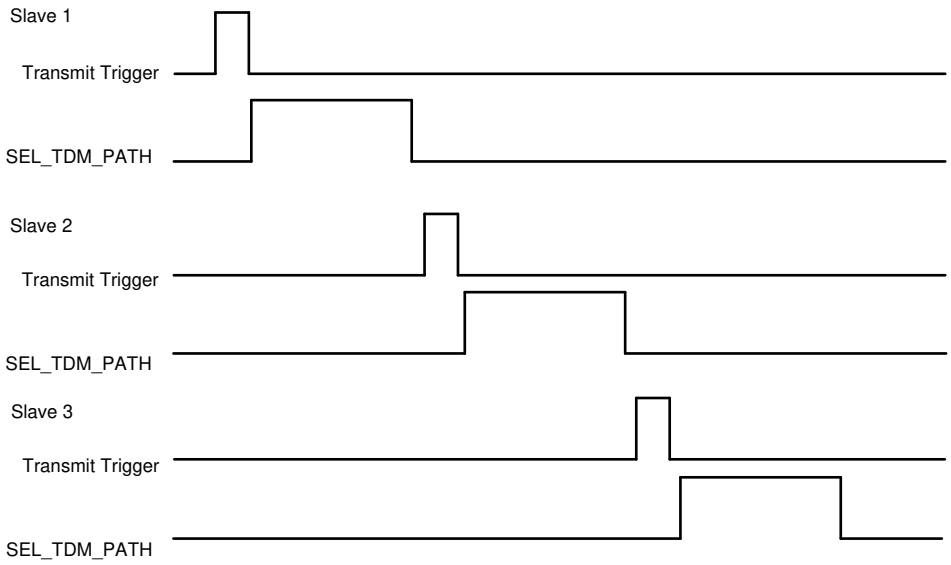


Figure 32-12. Generated Signals for FSI Multi-Slave TDM Configuration

The TXCLK_TDM_IN, TXD0_TDM_IN, and TXD1_TDM_IN for the FSI transmit modules available in this device are shown in Table 32-10.

Table 32-10. FSI TDM Inputs

FSI Transmit Module	TXCLK_TDM_IN	TXD0_TDM_IN	TXD1_TDM_IN
FSITXA	FSIRXB RXCLK	FSIRXB RXD0	FSIRXB RXD1
FSITXB	FSIRXA RXCLK	FSIRXA RXD0	FSIRXA RXD1

The FSI transmit modules are configured to use the external triggers generated by the CLB to initiate the transmission. Through appropriate configurations, the logic is designed to work assuming that only one TDM trigger is generated until the transmission is completed. In the case where there is another trigger before the current transmission is competed, an error is flagged in the TX_EVT_ERR_STATUS register. Figure 32-13 shows the connections between CLB, FSITX, and FSIRX modules.

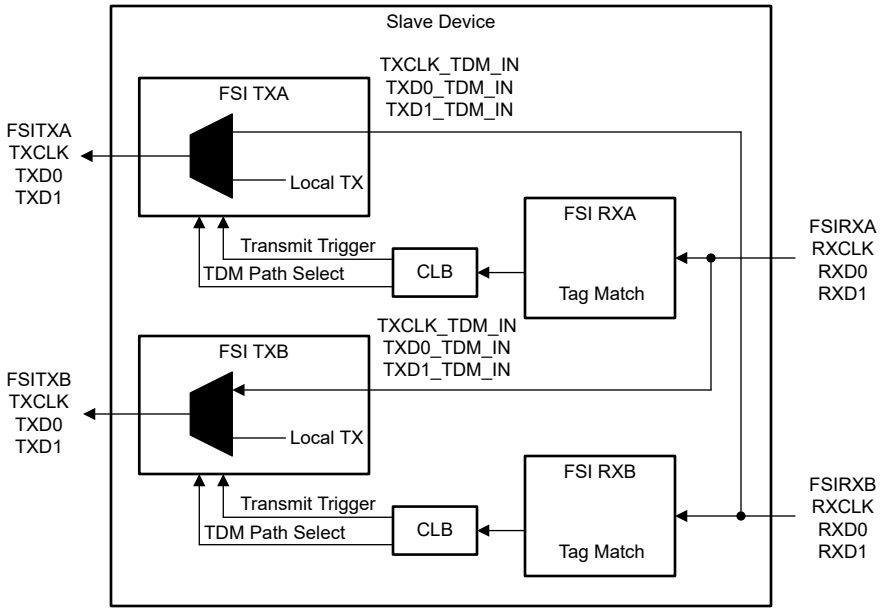


Figure 32-13. FSI and CLB Multi-Slave TDM Connections

In Figure 32-10, the master device is connected to three slave devices. The master device uses FSITXA to transmit frames to the three slave devices. Each slave device receives every frame, using their FSIRXA. The slave devices are chained together using their FSITXA output and FSIRXB inputs (configured as FSITXA TDM input). When a slave device receives a frame using the FSIRXA, the device checks the tag of the frame to see if the tag matches the specified reference value in the tag compare register. When a match does occur, the CLB module configures the FSITXA of the slave device to select the local FSITXA outputs as the source for the output pins. However, if a match does not occur, the FSITXA of the slave device is configured to enter bypass mode. There, the device selects the FSITXA TDM inputs (which is FSIRXB) to be passed on to the output pins.

In this case, when a frame is transmitted by the master device and the frame tag is matched only in one of the slave devices, all other devices with non-matching tags enter the bypass mode to allow the master device to receive frames from the chosen slave device.

32.3.11 FSI Trigger Generation

The RX_TRIGx external trigger can be used to initiate FSITX transmission. RX_TRIG0 must be used if TDM mode (multi-slave configuration) is required. RX_TRIG0 must be used as the trigger source for start of transmission while the programmable stretch width RX_TRIG0 signal is used as the SEL_TDM_PATH signal (which decides whether the local FSITX is active or put in bypass mode).

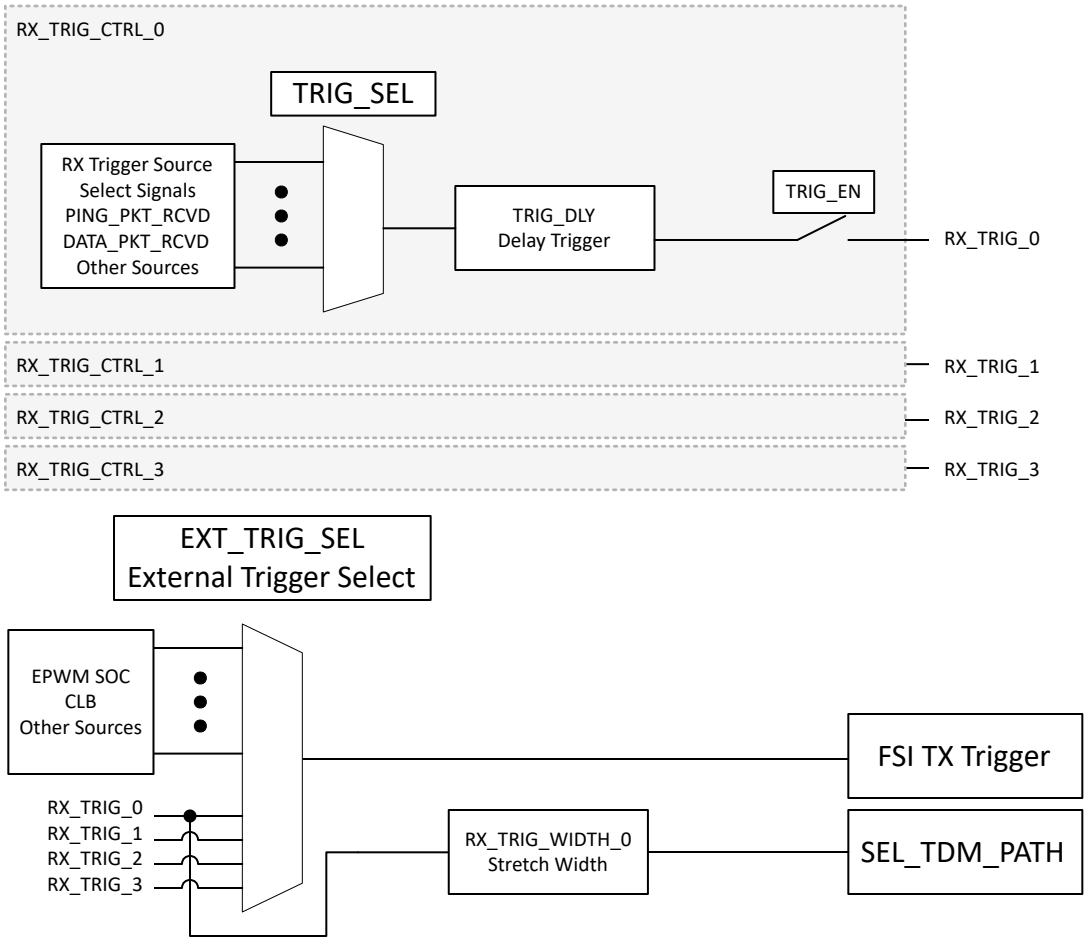


Figure 32-14. RX_TRIGx FSI Trigger

The signal source for the RX_TRIGx signal is selected through the RX_TRIG_CTRL_x.TRIG_SEL bits, as listed in Table 32-11.

Table 32-11. RX_TRIGx Trigger Select Signals

RX_TRIG_CTRLx.TRIG_SEL	Selected Signal
0	Ping Packet Received
1	Data Packet Received
2	Error Packet Received
3	Ping Frame Tag Match Occurred
4	Data Frame Tag Match Occurred
5	Error Frame Tag Match Occurred
6	Frame Done
7	Reserved
8 to 15	Reserved

The RX_TRIGx signals can optionally be delayed (this can be used in TDM scenarios) through the RX_TRIG_CTRL_x.TRIG_DLY.

32.3.12 FSI-SPI Compatibility Mode

The FSI supports a SPI compatibility mode. While the FSI can communicate with a standard SPI module, the FSI supports a limited configuration. The features of this compatibility mode are:

- Data transmits on rising edge and receive on falling edge of the clock.
- Only 16-bit word size is supported.
- TXD1 is driven like an active-low, chip-select signal. The signal is low for the duration for the full frame transmission.
- No receiver chip-select input is required. RXD1 is not used. Data is shifted into the receiver on every active clock edge.
- No preamble or postamble clocks are transmitted. All signals return to the IDLE state after the frame phase is finished.
- It is not possible to transmit in the SPI slave configuration because the FSI TXCLK cannot take an external clock source.

Table 32-12 lists the frame structure of the FSI-SPI compatibility mode. Each frame phase is present in this mode. If the FSI is transmitting to a standard SPI module, the SPI must decode the frame structure. Similarly, if the FSI is configured as a SPI slave, the standard SPI must encode the transmission to be sent.

Table 32-12. FSI-SPI Compatibility Frame Structure

Idle State	Start of Frame	Frame Type	User Data	Data Words	CRC byte(1)	Frame Tag	End of Frame	Idle State
	1001	4 bits	8 bits	1-16 words	8 bits	4 bits	0110	

(1) The CRC byte is present only in data frames.

Because of the requirement that the standard SPI module encodes the various frame data, this limits the type of modules that can be connected to the FSI in SPI mode. The paired SPI module must have enough functionality to encode and decode the frames.

If the FSI is transmitted to a standard 16-bit SPI, the data would be arranged in the following manner. The example provided in Table 32-13 assumes a DATA_2_WORD frame has been sent.

Table 32-13. Contents of Data Received by a Standard SPI

SPI Data	Data Contents
SPI word 0	1001, 0100, 8-bit User Data
SPI word 1	Data word 1
SPI word 2	Data word 2
SPI word 3	8-bit CRC, 4-bit Frame Tag, 0110

32.3.12.1 Available SPI Modes

There are a few wiring schemes available for the FSI to use when communicating with an SPI module.

32.3.12.1.1 FSITX as SPI Master, Transmit Only

The FSITX can operate as an independent SPI master module. In this condition, TXCLK is connected to SPICLK, TXD0 is connected to SPISIMO, and TXD1 is connected to $\overline{\text{SPISTE}}$, the chip select.

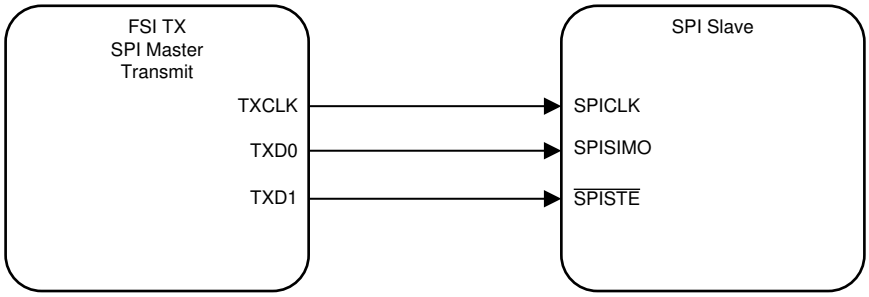


Figure 32-15. FSITX as SPI Master, Transmit Only

When the FSI is an SPI transmitter, the application has the ability to check for frame errors, line breaks, CRC errors, and ECC checks on data. These are all encoded by hardware in every FSI frame. The SPI receiver requires some software to act upon this information.

Table 32-14. FSI as Master Transmitter, SPI as Slave Receiver

Capability	Availability	Comment
Framing checks on the data frames	Yes	Can be implemented in software on the SPI receiver.
Ability to detect line breaks	Yes	Can be implemented in software on the SPI receiver but requires additional software overhead such as a timer or watchdog.
CRC check	Yes	Can be implemented in software on the SPI receiver. For devices that have VCRC , this is more efficient.
ECC on data	Yes	Can be implemented in software on the SPI receiver
Detection of abruptly terminated frames	No	
Double edge data rate	No	
Recovery from glitches on signal lines between frames	No	
Skew adjustment on signal lines	No	

32.3.12.1.1.1 Initialization

To configure the FSITX module to be an SPI master for transmit only, proceed through the standard FSITX initialization procedure. Before releasing the FSITX from reset, set TX_OPER_CTRL_LO.SPI_MODE to 1. This enables the SPI clocking scheme and signaling structure.

32.3.12.1.1.2 Operation

The operation of the FSITX module in FSI-SPI Compatibility mode is the same as if the module is in standard FSI mode. The application can utilize the frame timer, ping frames, external frame triggers, and so on. Refer to Section 32.3.2 for more information on each of these features.

32.3.12.1.2 FSIRX as SPI Slave, Receive Only

The FSIRX can operate as an independent SPI slave module. In this usage, RXCLK is connected to SPICLK and RXD0 is connected to SPISIMO. RXD1 is unused. There is no requirement for a chip select signal to be used when connected to the FSIRX. This is because the FSIRX responds to any incoming clock edge. If there is any noise or unwanted clock transitions, a flush sequence is required to resynchronize the FSIRX module with the master.

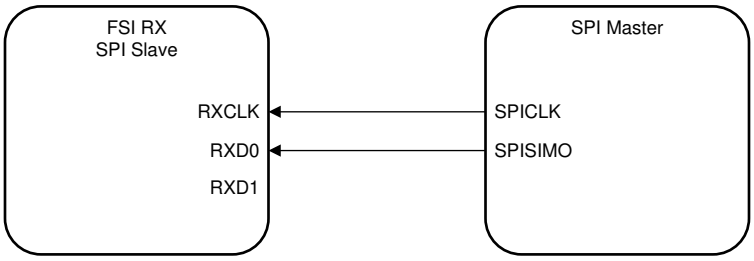


Figure 32-16. FSIRX as SPI Slave, Receive Only

When the FSI is an SPI receiver communicating with an SPI transmitter, the application has the ability to detect frame errors, line breaks, CRC errors, ECC checks on data, as well as abruptly terminated frames. Note that the FSI can handle all of this in hardware, but the SPI transmitter must encode the information into the data to be transmitted.

Table 32-15. SPI as Master Transmitter, FSI as Slave Receiver

Capability	Availability	Comment
Framing checks on the data frames	Yes	Standard on FSI
Ability to detect line breaks	Yes	Can be implemented in software on the SPI transmitter but requires the use of a timer or watchdog in the transmitting SPI device.
CRC check	Yes	Can be implemented in software on the SPI transmitter.
ECC on data	Yes	Can be implemented in software on the SPI transmitter.
Detection of abruptly terminated frames	Yes	This is accomplished with the FSI setting up the frame watchdog counter.
Double edge data rate	No	
Recovery from glitches on signal lines between frames	Yes	Whenever glitches occur on either the clock or data lines in between transmissions, the initial flush pattern of a frame discards the effects of these glitches and causes the receiver to resynchronize when the real “start-of-frame” pattern is seen. So, the ability to reject glitches in between frames is very high.
Skew adjustment on signal lines	Yes	The FSI receiver has the ability to add delays to the incoming signal lines.

32.3.12.1.2.1 Initialization

To configure the FSIRX module to be an SPI slave for receiving only, proceed through the standard FSIRX initialization procedure. Before releasing the FSIRX from reset, set RX_OPER_CTRL.SPI_MODE to 1. This enables the SPI clocking scheme and signaling structure.

32.3.12.1.2.2 Operation

The operation of the FSIRX module in FSI-SPI compatibility mode is the same as if the module is in standard FSI mode. The application can utilize the Frame and Ping Watchdogs, CRC and ECC checks, and so on. Refer to Section 32.3.3 for more information on each of these features.

32.3.12.1.3 FSITX and FSIRX Emulating a Full Duplex SPI Master

In this configuration, the FSITX is the master clock. The FSITX module drives TXCLK (SPICLK), TXD0 (SPISIMO), and TXD1 (SPISTE/chip select) to the SPI slave. The SPISOMI signal is connected back to the RXD0 signal. RXCLK can be applied either using the internal SPI pairing feature or externally wired, depending on the application requirements. Since the FSITX and RX modules are independent, the FSIRX could also be thought of as an additional SPI slave. Some software logic is required in order for the FSI to emulate an SPI master fully.

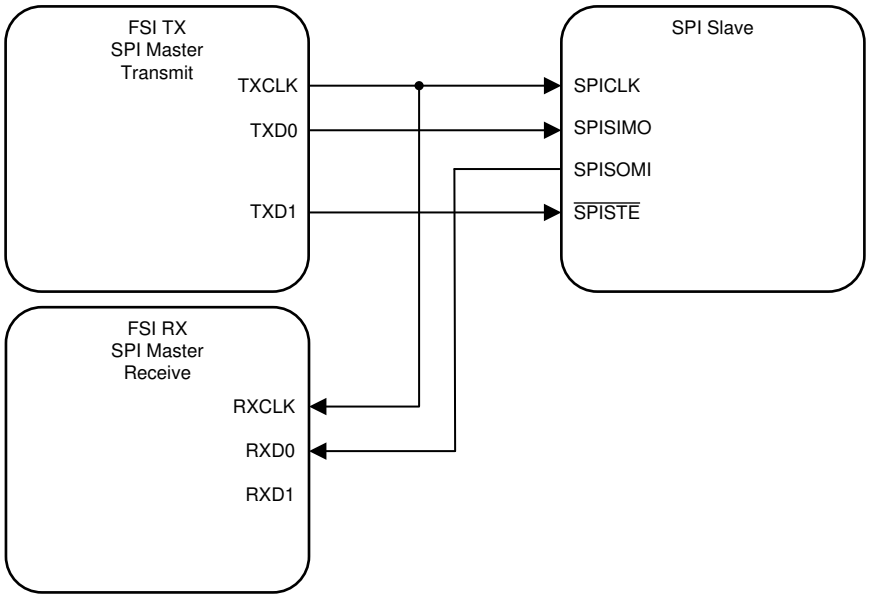


Figure 32-17. FSITX and FSIRX as SPI Master, Full Duplex

32.3.12.1.3.1 Initialization

To configure both FSITX and RX modules for full duplex SPI master operation, follow the initialization instructions for each module described in the preceding sections. Both FSITX and RX modules must set their respective SPI_MODE bits. This enables the SPI clocking scheme and signaling structures.

If internal clock loopback is desired, the FSIRX module must also set RX_MASTER_CTRL.SPI_PAIRING to 1. This internally connects TXCLK to RXCLK. If using internal clock loopback, the GPIO used for RXCLK can be reallocated to other application requirements.

If the application requires an external clock loopback, make sure that TXCLK is connected to RXCLK. This is required if the SPI slave is across an isolation barrier and there is latency between TXCLK being launched and SPISOMI data being received on RXD0.

32.3.12.1.3.2 Operation

In this mode of operation, some higher level software must be written to emulate a full SPI master module. There is no path for the transmit module to determine what the receive module received. Both the TX and RX modules are still able to utilize the various other features available, such as the ping frame timer, ping frame and frame watchdogs, CRC and ECC error checkers, and so on. The procedure for configuring these features is described elsewhere in this document.

32.4 FSI Programing Guide

This section describes various operational sequences and features for the FSI.

32.4.1 Establishing the Communication Link

Once the transmitter and receiver modules have been configured, some synchronization must occur before the modules exchange data. Since the receiver accepts data on any clock transition, the receiver core logic must be flushed to properly interpret the start of a new, valid frame. This is especially true when the FSI modules reside on separate devices and are possibly isolated.

The following example provides a suggested approach for establishing a clean communication link on two separate devices that power up in an arbitrary order. Note that this is only a sample synchronization. Depending on application requirements, a different approach can be followed. The single, most important aspect of synchronization is to make sure that the receiver is properly flushed and ready to receive a complete frame without error. How to achieve this is up to the application.

Figure 32-18 shows the connection of the devices in this example. While there is no true concept of a master device or a slave device node in the FSI protocol, the example uses this nomenclature as a simple way to describe the data flow.

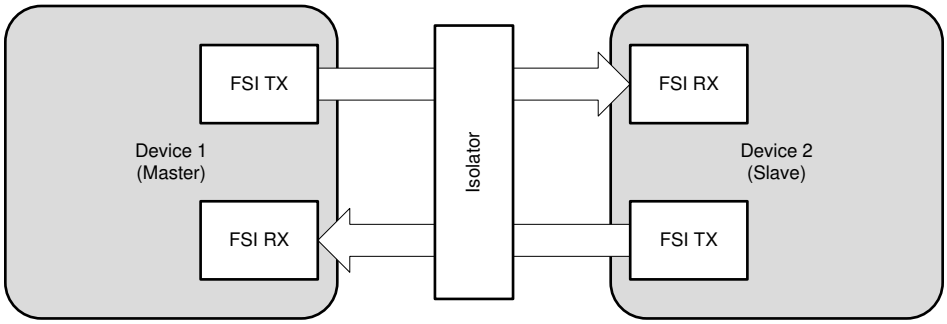


Figure 32-18. Point to Point Connection

Device 1 is the master node; it is the driver of the initialization sequence. Device 2 is the slave node; it responds to the master device commands. In this example, as well as in a real world use-case, neither the master device nor the slave device knows precisely when the other is ready to receive communication.

Sample sequences for both the master device and slave device are provided in the following subsections.

32.4.1.1 Establishing the Communication Link from the Master Device

The following sequence is an example of how the master device node establishes the communication link with the slave device without external signals outside of the standard communication link.

1. Assert the core reset to both the FSITX and FSIRX modules, and then deassert the resets.
2. Configure the transmitter and receiver for desired operation.
3. Set up the receiver interrupts to detect an incoming transmission.
4. Begin the ping loop:
   - Send the flush sequence.
   - Send a ping frame with the frame tag 0000.
   - Wait for some time. (determined by application)
   - If the FSIRX has received a valid ping frame, continue; else iterate the loop again.
   - If the received ping frame tag was 0001, continue; else iterate the loop again.
5. Send a ping frame with the frame tag 0001.

At this point, both the master transmit and receive channels have successfully received a frame from their slave counterparts. The link has been established and standard application communication can begin.

32.4.1.2 Establishing the Communication Link from the Slave Device

The following sequence is an example of how the slave device node establishes the communication link with the master device without external signals outside of the standard communication link.

1. Apply the core reset to both the FSITX and FSIRX modules, and then release the reset.
2. Configure the transmitter and receiver for desired operation.
3. Set up the receiver interrupts to detect an incoming transmission.
4. Wait for a receiver interrupt.
5. If the FSIRX has received a valid ping frame, continue; else return to step 4.
6. If the received frame tag was 0000, continue; else discard the transmission and return to step 4.
7. Send the flush sequence.
8. Send a ping frame with the frame tag 0001.
9. Wait for a receiver interrupt.
10. If the FSIRX has received a valid ping frame, continue; else return to step 4.
11. If the received ping frame tag was 0001, continue; else if the received frame tag was 0000, return to step 9. This can happen if a second ping frame was already in transit before receiving the slave device response in step 8.

At this point, both the transmit and receive modules have successfully received ping frames from their master counterparts. The link has been established and regular communication can now proceed. The application can configure periodic ping frames from the transmitter, initialize the receiver ping and frame watchdogs, and begin the communication required by the application.

32.4.2 Register Protection

Both the FSITX and FSIRX modules contain control registers that have embedded write protection. This is accomplished through EALLOW, register keys, and a master register lock. These protections make sure that no spurious writes or unintentional modifications to these registers are accepted. For the list of registers with write protections available and the register and bit descriptions, refer to Section 32.6.

EALLOW Protection

EALLOW is a device-level register protection. For those registers with EALLOW protection, the EALLOW bit is set before modifying the register. The application then clears the EALLOW bit to re-enable the write protection when access to EALLOW-protected registers are complete.

Register Key Protection

In addition to EALLOW, some bits in the FSI registers are protected by a key. To write to these bits, the key must be written at the same time. For example, to put the transmitter core into reset, TX_MASTER_CTRL.CORE_RST must be set. To do this, write 0xA501 to TX_MASTER_CTRL, where 0xA500 is the KEY value, and 0x0001 is the CORE_RST value. Refer to the *Registers* section for more information on which registers have write keys added.

Control Register Lock Protection

There also exists a master lock to prevent any modifications to the control registers. There is an independent lock for each FSI module. For the list of registers that are protected by this control register lock, refer to the *Registers* section. The control register lock prevents any writes to the control registers until the lock is released. To set the control register lock, write 0xA501 to RX_LOCK_CTRL and TX_LOCK_CTRL for the receiver and transmitter, respectively.

The control register lock cannot be disabled by the application until a SYSRSn has been asserted. This can occur at the device level, or by writing to the appropriate peripheral soft reset register (DEV_CFG_REGS.SOFTPRESx) for the FSI module. Refer to Section 32.3.2.7 for more information on SYSRSn.

32.4.3 Emulation Mode

There is no specific emulation mode or configuration supported. The FSI cores are always in free running mode. CPU halts do not have any effect on the operation of the FSI. Reads of registers and data buffers by the debugger do not affect any flags or status of the data buffers.

If you want to stop the operation of either FSI module when the debugger halts, the following steps are required:

1. Set the debugger to real-time emulation mode.
2. Mark the FSI interrupt group as a time-critical interrupt. That is, enable the corresponding bit in the DBGIER register.
3. The ISR can check the DSTAT register and to determine if the ISR was called when the debugger was halted.
4. FSI operations can be disabled and the ISR can branch to a debug-specific halt location.

32.5 Software

32.5.1 FSI Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/fsi

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

32.5.1.1 FSI Multi-Rx Tag-Match - C28X_DUAL

FILE: fsi_ex1_multiRx_tagmatch_cpu2.c

Example sets up infinite data frame transfers where trigger happens through *CPU*. Multiple receivers receive data as per the received frame tag.

This is a dual core example where FSITxA & FSIRxA instances are owned by CPU1 while FSIRxB, FSIRxC & FSIRxD are owned by CPU2. Internal loopback mode is enabled for FSIRxA, FSIRxB, FSIRxC & FSIRxD which connects data & clock lines of these receivers to FSITxA internally.

FSITxA infinitely sends data frames with alternating tag values. Receivers are configured to receive data frame with different tag values with tag-match feature enabled. Tx doesn't send next frame of data until it all receivers receive the data. Synchronization among all the receivers is maintained through IPC flags.

User can edit some of configuration parameters as per usecase. These are as below. Default values can be referred in code where these globals are defined:-

- *nWords* - Number of words per transfer may be from 1 -16
- *nLanes* - Choice to select single or double lane for frame transfers
- *fsiClock* - FSI Clock used for transfers
- *txUserData* - User data to be sent with Data frame
- *txDataFrameTag* - Frame tag used for Data transfers

For any errors during transfers i.e. *error* events such as Frame Overrun, Underrun, Watchdog timeout and CRC/EOF/TYPE errors, execution will stop immediately and status variables can be looked into for more details. Execution will also stop for any mismatch between received data and sent ones and also if transfers takes unusually long time(detected through software counters - txTimeOutCntr and rxTimeOutCntr)

External Connections
For FSI internal loopback, no external connections needed

Watch Variables

- *dataFrameCntrB* Number of Data frame received by FSIRxB
- *dataFrameCntrC* Number of Data frame received by FSIRxC
- *dataFrameCntrD* Number of Data frame received by FSIRxD
- *error* Non zero for transmit/receive data mismatch

32.5.1.2 FSI Loopback:CPU Control

FILE: fsi_ex1_loopback_cpucontrol.c

Example sets up infinite data frame transfers where trigger happens through *CPU*. Automatic(Hw triggered) Ping frame transmission is also setup along with data.

User can edit some of configuration parameters as per usecase. These are as below. Default values can be referred in code where these globals are defined

- *nWords* - Number of words per transfer may be from 1 -16
- *nLanes* - Choice to select single or double lane for frame transfers
- *fsiClock* - FSI Clock used for transfers
- *txUserData* - User data to be sent with Data frame
- *txDataFrameTag* - Frame tag used for Data transfers
- *txPingFrameTag* - Frame tag used for Ping transfers

- *txPingTimeRefCntr* - Tx Ping timer reference counter
- *rxWdTimeoutRefCntr* - Rx Watchdog timeout reference counter

For any errors during transfers i.e. *error* events such as Frame Overrun, Underrun, Watchdog timeout and CRC/EOF/TYPE errors, execution will stop immediately and status variables can be looked into for more details. Execution will also stop for any mismatch between received data and sent ones and also if transfers takes unusually long time(detected through software counters - txTimeOutCntr and rxTimeOutCntr)

External Connections
For FSI internal loopback (EXTERNAL_FSI_ENABLE == 0), no external connections needed

For FSI external loopback (EXTERNAL_FSI_ENABLE == 1), external connections are required. The FSI TX pins should be connected to the FSI RX pins of the same device. See below for external connections to include and GPIOs used:

External Connections Required between FSI TX and RX of the same device:
- FSIRX_CLK to FSITX_CLK
- FSIRX_RX0 to FSITX_TX0
- FSIRX_RX1 to FSITX_TX1

- ControlCard FSI Header GPIOs:

GPIO_27 -> FSITXA_CLK
- GPIO_26 -> FSITXA_TX0
- GPIO_25 -> FSITXA_TX1
- GPIO_9 -> FSIRXA_CLK
- GPIO_8 -> FSIRXA_RX0
- GPIO_10 -> FSIRXA_RX1

Watch Variables
- *dataFrameCntr* Number of Data frame transfered
- *error* Non zero for transmit/receive data mismatch

32.5.1.3 FSI Multi-Rx Tag-Match - C28X_DUAL

FILE: fsi_ex1_multiRx_tagmatch_cpu1.c

Example sets up infinite data frame transfers where trigger happens through *CPU*. Multiple receivers receive data as per the received frame tag.

This is a dual core example where FSITxA & FSIRxA instances are owned by CPU1 while FSIRxB, FSIRxC & FSIRxD are owned by CPU2. Internal loopback mode is enabled for FSIRxA, FSIRxB, FSIRxC & FSIRxD which connects data & clock lines of these receivers to FSITxA internally.

FSITxA infinitely sends data frames with alternating tag values. Receivers are configured to receive data frame with different tag values with tag-match feature enabled. Tx doesn't send next frame of data until it all receivers receive the data. Synchronization among all the receivers is maintained through IPC flags.

User can edit some of configuration parameters as per usecase. These are as below. Default values can be referred in code where these globals are defined:-
- *nWords* - Number of words per transfer may be from 1 -16
- *nLanes* - Choice to select single or double lane for frame transfers
- *fsiClock* - FSI Clock used for transfers
- *txUserData* - User data to be sent with Data frame
- *txDataFrameTag* - Frame tag used for Data transfers

For any errors during transfers i.e. *error* events such as Frame Overrun, Underrun, Watchdog timeout and CRC/EOF/TYPE errors, execution will stop immediately and status variables can be looked into for more details. Execution will also stop for any mismatch between received data and sent ones and also if transfers takes unusually long time(detected through software counters - txTimeOutCntr and rxTimeOutCntr)

External Connections

For FSI internal loopback, no external connections needed

Watch Variables
- *dataFrameCntrA* Number of Data frame transfered
- *error* Non zero for transmit/receive data mismatch

32.5.1.4 FSI Loopback CLA control

FILE: fsi_ex2_loopback_clacontrol.c

Example sets up infinite data frame transfers where trigger happens through *CLA*. Automatic(Hw triggered) Ping frame transmission is also setup along with data. This example is similar to fsi_ex1_loopback_cpucontrol and only different in in the sense that data frame transfer are trigged from a CLA task. Using CLA will release some of load from CPU and help it in providing time for other tasks.

User can edit some of configuration parameters as per usecase. These are as below. Default values can be referred in code where these globals are defined

- *nWords* - Number of words per transfer may be from 1 -16
- *nLanes* - Choice to select single or double lane for frame transfers
- *fsiClock* - FSI Clock used for transfers
- *txUserData* - User data to be sent with Data frame
- *txDataFrameTag* - Frame tag used for Data transfers
- *txPingFrameTag* - Frame tag used for Ping transfers
- *txPingTimeRefCntr* - Tx Ping timer reference counter
- *rxWdTimeoutRefCntr* - Rx Watchdog timeout reference counter

For any errors during transfers i.e. *error* events such as Frame Overrun, Underrun, Watchdog timeout and CRC/EOF/TYPE errors, execution will stop immediately and status variables can be looked into for more details. Execution will also stop for any mismatch between received data and sent ones and also if transfers takes unusually long time(detected through software counters - txTimeOutCntr and rxTimeOutCntr)

External Connections
For FSI internal loopback (EXTERNAL_FSI_ENABLE == 0), no external connections needed

For FSI external loopback (EXTERNAL_FSI_ENABLE == 1), external connections are required. The FSI TX pins should be connected to the respective FSI RX pins of the same device. See below for external connections to include and GPIOs used:

External Connections Required between FSI TX and RX of the same device:
- FSIRX_CLK to FSITX_CLK
- FSIRX_RX0 to FSITX_TX0
- FSIRX_RX1 to FSITX_TX1

- ControlCard FSI Header GPIOs:

GPIO_27 -> FSITXA_CLK
- GPIO_26 -> FSITXA_TX0
- GPIO_25 -> FSITXA_TX1
- GPIO_9 -> FSIRXA_CLK
- GPIO_8 -> FSIRXA_RX0
- GPIO_10 -> FSIRXA_RX1

Watch Variables
- *dataFrameCntr* Number of Data frame transfered
- *error* Non zero for transmit/receive data mismatch

32.5.1.5 FSI DMA frame transfers:DMA Control

FILE: fsi_ex3_loopback_dmacontrol.c

Example sets up infinite data frame transfers where DMA trigger happens once through CPU and then DMA takes control to transfer data iteratively. This example demonstrates the FSI feature about triggering DMA events which in turn can copy data and trigger next transfer.

Two DMA channels are setup for FSI Tx operation and two for Rx. Four areas in GSx memories are also setup as source and sink for data and tag values of frame under transmission.

Automatic(Hw triggered) Ping frame transmission is also setup along with data.

If there are any comparison failures during transfers or any of error event occurs, execution will stop.

External Connections
For FSI internal loopback (EXTERNAL_FSI_ENABLE == 0), no external connections needed

For FSI external loopback (EXTERNAL_FSI_ENABLE == 1), external connections are required. The FSI TX pins should be connected to the respective FSI RX pins of the same device. See below for external connections to include and GPIOs used:

External Connections Required between FSI TX and RX of the same device:
- FSIRX_CLK to FSITX_CLK
- FSIRX_RX0 to FSITX_TX0
- FSIRX_RX1 to FSITX_TX1

- ControlCard FSI Header GPIOs:

GPIO_27 -> FSITXA_CLK
- GPIO_26 -> FSITXA_TX0
- GPIO_25 -> FSITXA_TX1
- GPIO_9 -> FSIRXA_CLK
- GPIO_8 -> FSIRXA_RX0
- GPIO_10 -> FSIRXA_RX1

Watch Variables
- *countDMAtransfers* Number of Data frame transfered
- *error* Non zero for transmit/receive data mismatch

32.5.1.6 FSI data transfer by external trigger

FILE: fsi_ex4_loopback_epwmtrigger.c

FSI frame transfer can be triggered by external sources. It can connect up to 32 trigger sources but as of now, only 16 ePWMx-SOCy(x-1:8, y-A:B) are supported. FSI supports external trigger for both PING and DATA frame transfers and in this example we demonstrate how to setup infinite DATA transfers using selectable ePWM-SOC as a trigger source. The TB counter for ePWM operation is in up/down count mode for this example.

Automatic(Hw triggered) Ping frame transmission is also setup along with data.

If there are any comparison failures during transfers or any of error event occurs, execution will stop.

External Connections
For FSI internal loopback (EXTERNAL_FSI_ENABLE == 0), no external connections needed

For FSI external loopback (EXTERNAL_FSI_ENABLE == 1), external connections are required. The FSI TX pins should be connected to the respective FSI RX pins of the same device. See below for external connections to include and GPIOs used:

External Connections Required between FSI TX and RX of the same device:
- FSIRX_CLK to FSITX_CLK
- FSIRX_RX0 to FSITX_TX0
- FSIRX_RX1 to FSITX_TX1

- ControlCard FSI Header GPIOs:

GPIO_27 -> FSITXA_CLK

- GPIO_26 -> FSITXA_TX0
- GPIO_25 -> FSITXA_TX1
- GPIO_9 -> FSIRXA_CLK
- GPIO_8 -> FSIRXA_RX0
- GPIO_10 -> FSIRXA_RX1

Watch Variables
- *dataFrameCntr* Number of Data frame transfered
- *error* Non zero for transmit/receive data mismatch

32.5.1.7 FSI data transfers upon CPU Timer event

FILE: fsi_ex5_periodic_frame.c

Example sets up infinite data frame transfers where trigger comes from ISR handling the periodic CPU Timer event. Automatic(Hw triggered) Ping frame transmission is also setup along with data.

CPU Timer0 is chosen for setting up periodic timer events. User can choose any other Timer-1/Timer-2 as well.

Automatic(Hw triggered) Ping frame transmission is also setup along with data.

If there are any comparison failures during transfers or any of error event occurs, execution will stop.

External Connections
For FSI internal loopback (EXTERNAL_FSI_ENABLE == 0), no external connections needed

For FSI external loopback (EXTERNAL_FSI_ENABLE == 1), external connections are required. The FSI TX pins should be connected to the respective FSI RX pins of the same device. See below for external connections to include and GPIOs used:

External Connections Required between FSI TX and RX of the same device:
- FSIRX_CLK to FSITX_CLK
- FSIRX_RX0 to FSITX_TX0
- FSIRX_RX1 to FSITX_TX1

- ControlCard FSI Header GPIOs:
- GPIO_27 -> FSITXA_CLK
- GPIO_26 -> FSITXA_TX0
- GPIO_25 -> FSITXA_TX1
- GPIO_9 -> FSIRXA_CLK
- GPIO_8 -> FSIRXA_RX0
- GPIO_10 -> FSIRXA_RX1

Watch Variables
- *dataFrameCntr* Number of Data frame transfered
- *error* Non zero for transmit/receive data mismatch

32.5.1.8 FSI and SPI communication(fsi_ex6_spi_main_tx)

FILE: fsi_ex6_spi_main_tx.c

FSI supports SPI compatibility mode to talk to the devices not having FSI but SPI module. Example sets up infinite data frame transfers where FSI acts like main Tx and SPI as remote Rx. API to decode FSI frame received at SPI end is implemented and checks are made to ensure received details(frame tag/type, userdata, data) match with transfered frame.

If there are any comparison failures during transfers or any of error event occurs, execution will stop.

External Connections
For FSI <-> SPI communication, make below connections in GPIO settings

- GPIO_2 -> GPIO_18 :: To connect FSITX_CLK with SPICLKA
- GPIO_0 -> GPIO_16 :: To connect FSITX_TX0 with SPIPICOA

- GPIO_1 -> GPIO_19 :: To connect FSITX_TX1 with SPIPTEA

Watch Variables
- *dataFrameCntr* Number of Data frame transfered
- *error* Non zero for transmit/receive data mismatch

32.5.1.9 FSI and SPI communication(fsi_ex7_spi_remote_rx)

FILE: fsi_ex7_spi_remote_rx.c

FSI supports SPI compatibility mode to talk to the devices not having FSI but SPI module. Example sets up infinite data frame transfers where FSI acts like remote Rx and SPI as main Rx. API to build the FSI frame at SPI end before transfer is implemented in SW and checks are made to ensure received details(frame tag/type, userdata, data) on FSI Rx match with transferred data.

If there are any comparison failures during transfers or any of error event occurs, execution will stop.

External Connections
For FSI(Rx) <-> SPI(Tx) communication, make connections in GPIO settings

There is no requirement for a chip select signal to be used when connected to the FSIRX. This is because the FSIRX will respond to any incoming clock edge.

- GPIO_13 -> GPIO_18 :: To connect FSIRXCLKA with SPICLKA
- GPIO_12 -> GPIO_16 :: To connect FSIRXD0A with SPIPICOA

Watch Variables
- *dataFrameCntr* Number of Data frame transfered
- *error* Non zero for transmit/receive data mismatch

32.5.1.10 FSI P2Point Connection:Rx Side

FILE: fsi_ex8_ext_p2pconnection_rx.c

Example sets up FSI receiving device in a point to point connection to the FSI transmitting device. Example code to set up FSI transmit device is implemented in a separate file.

In a real scenario two separate devices may power up in arbitrary order and there is a need to establish a clean communication link which ensures that receiver side is flushed to properly interpret the start of a new valid frame.

There is no true concept of a main or a remote node in the FSI protocol, but to simplify the data flow and connection we can consider transmitting device as main and receiving side as remote. Transmitting side will be driver of initialization sequence.

Handshake mechanism which must take place before actual data transmission can be usecase specific; points described below can be taken as an example on how to implement the handshake from receiving side -

- Setup the receiver interrupts to detect PING type frame reception
- Begin the first PING loop
  - Wait for receiver interrupt
  - If the FSI Rx has received a PING frame with *FSI_FRAME_TAG0*, come out of loop. Otherwise iterate the loop again.
- Begin the second PING loop
  - Send the Flush sequence
  - Send the PING frame with tag
  - Wait for receiver interrupt
  - If the FSI Rx has received a PING frame with *FSI_FRAME_TAG1*, come out of loop. Otherwise iterate the loop again.

Now, the receiver side has received the acknowledged PING frame(tag1), so it is ready for normal operation further.

After above synchronization steps, FSI Rx can be configured as per usecase i.e. nWords, lane width, enabling events etc and start the infinite transfers. More details on establishing the communication link can be found in device TRM.

User can edit some of configuration parameters as per usecase, similar to other examples.

nWords - Number of words per transfer may be from 1 -16 *nLanes* - Choice to select single or double lane for frame transfers *fsiClock* - FSI Clock used for transfers *txUserData* - User data to be sent with Data frame *txDataFrameTag* - Frame tag used for Data transfers *txPingFrameTag* - Frame tag used for Ping transfers *txPingTimeRefCntr* - Tx Ping timer reference counter *rxWdTimeoutRefCntr* - Rx Watchdog timeout reference counter

External Connections
For FSI external P2P connection, external connections are required to be made between two devices. Device 1's FSI TX and RX pins need to be connected to device 2's FSI RX and TX pins respectively. See below for external connections to make and GPIOs used:

External connections required between independent RX and TX devices:
- FSIRX_CLK to FSITX_CLK
- FSIRX_RX0 to FSITX_TX0
- FSIRX_RX1 to FSITX_TX1

- ControlCard FSI Header GPIOs:
- GPIO_27 -> FSITXA_CLK
- GPIO_26 -> FSITXA_TX0
- GPIO_25 -> FSITXA_TX1
- GPIO_9 -> FSIRXA_CLK
- GPIO_8 -> FSIRXA_RX0
- GPIO_10 -> FSIRXA_RX1

Watch Variables
- *dataFrameCntr* Number of Data frame received
- *error* Non zero for transmit/receive data mismatch

32.5.1.11 FSI P2Point Connection:Tx Side

FILE: fsi_ex8_ext_p2pconnection_tx.c

Example sets up FSI transmitting device in a point to point connection to the FSI receiving device. Example code to set up FSI receiving device is implemented in a separate file.

In a real scenario two separate devices may power up in arbitrary order and there is a need to establish a clean communication link which ensures that receiver side is flushed to properly interpret the start of a new valid frame.

There is no true concept of a main or a remote node in the FSI protocol, but to simplify the data flow and connection we can consider transmitting device as main and receiving side as remote. Transmitting side will be driver of initialization sequence.

Handshake mechanism which must take place before actual data transmission can be usecase specific; points described below can be taken as an example on how to implement the handshake from transmitting side -

- Setup the receiver interrupts to detect PING type frame reception
- Begin the PING loop
  - Send the Flush sequence
  - Send a PING frame with the frame tag *FSI_FRAME_TAG0*
  - Wait for some time(determined by application)
  - If the FSI Rx has received a PING frame with *FSI_FRAME_TAG1*, come out of loop. Otherwise iterate the loop again

Send a PING frame with the frame tag FSI_FRAME_TAG1

After above synchronization steps, FSI Tx can be configured as per usecase i.e. nWords, lane width, enabling events etc and start the infinite transfers. More details on establishing the communication link can be found in device TRM.

User can edit some of configuration parameters as per usecase, similar to other examples.

nWords - Number of words per transfer may be from 1 -16 *nLanes* - Choice to select single or double lane for frame transfers *fsiClock* - FSI Clock used for transfers *txUserData* - User data to be sent with Data frame *txDataFrameTag* - Frame tag used for Data transfers *txPingFrameTag* - Frame tag used for Ping transfers *txPingTimeRefCntr* - Tx Ping timer reference counter *rxWdTimeoutRefCntr* - Rx Watchdog timeout reference counter

External Connections
For FSI external P2P connection, external connections are required to be made between two devices. Device 1's FSI TX and RX pins need to be connected to device 2's FSI RX and TX pins respectively. See below for external connections to make and GPIOs used:

External connections required between independent RX and TX devices:
- FSIRX_CLK to FSITX_CLK
- FSIRX_RX0 to FSITX_TX0
- FSIRX_RX1 to FSITX_TX1

- ControlCard FSI Header GPIOs:

GPIO_27 -> FSITXA_CLK
- GPIO_26 -> FSITXA_TX0
- GPIO_25 -> FSITXA_TX1
- GPIO_9 -> FSIRXA_CLK
- GPIO_8 -> FSIRXA_RX0
- GPIO_10 -> FSIRXA_RX1

Watch Variables
- *dataFrameCntr* Number of Data frame transmitted
- *error* Non zero for transmit/receive data mismatch

32.5.1.12 FSI star connection topology example. FSI communication using CPU control

FILE: fsi_ex9_star_broadcast.c fsi_ex9_star_broadcast is for the central device in the star topology, fsi_ex16_daisy_handshake_node (CPU Control Only) is used for the N node devices. The fsi_ex16_daisy_handshake_node software example is currently only available for the F28004x devices.

In a real scenario two separate devices may power up in arbitrary order and there is a need to establish a clean communication link which ensures that receiver side is flushed to properly interpret the start of a new valid frame.

The central device in the star topology initiates and drives the handshake sequence and subsequent broadcast data transmissions. The node devices receive and respond to the broadcasts.

After above synchronization steps, FSI Rx can be configured as per use case i.e. nWords, lane width, enabling events, etc and start the infinite transfers. More details on establishing the communication link can be found in the device TRM.

Preprocessor Directives FSI_RXA_ENABLE, FSI_RXB_ENABLE, FSI_RXC_ENABLE are used to enable different FSI RX instances. Each node device's FSI TX should be connected to a FSI RX instance of the central device. The FSI TX of the central device should be connected to each node device's FSI RX.

User can edit some of configuration parameters as per use case, similar to other examples.

nWords - Number of words per transfer may be from 1 -16 *nLanes* - Choice to select single or double lane for frame transfers *txUserData* - User data to be sent with Data frame *txDataFrameTag* - Frame tag used for Data transfers *txPingFrameTag* - Frame tag used for Ping transfers *txPingTimeRefCntr* - Tx Ping timer reference

counter *rxWdTimeoutRefCntr* - Rx Watchdog timeout reference counter *rxTimeOutCntr* - Rx timeout reference counter used in handshake sequence

External Connections
For the FSI star connection topology, external connections are required to be made between the included devices. The FSI TXA pins of the central device (F2838x) needs to be connected to the FSI RX pins of all node devices (broadcast to node devices). The FSI RXA, RXB, RXC pins of the central device should be connected to each individual node device's FSI TX pins. See below for external connections to include and GPIOs used:

External Connections Required:
- FSIRX_CLK to FSITX_CLK
- FSIRX_RX0 to FSITX_TX0
- FSIRX_RX1 to FSITX_TX1

ControlCard FSI Header GPIOs:

- GPIO_27 -> FSITXA_CLK
- GPIO_26 -> FSITXA_TX0
- GPIO_25 -> FSITXA_TX1
- GPIO_9 -> FSIRXA_CLK
- GPIO_8 -> FSIRXA_RX0
- GPIO_10 -> FSIRXA_RX1
- GPIO_60 -> FSIRXB_CLK
- GPIO_58 -> FSIRXB_RX0
- GPIO_59 -> FSIRXB_RX1
- GPIO_14 -> FSIRXC_CLK
- GPIO_12 -> FSIRXC_RX0
- GPIO_13 -> FSIRXC_RX1

FSI TXA of the central device needs to be connected to FSI RX of all node devices (broadcast to nodes). FSI RXA, RXB, RXC of the central device should be connected to each individual node device's FSI TX.

Watch Variables
- *dataFrameCntr_A* Number of Data frame received on FSI RXA
- *frame_type_error_A* Counts number of RXA frame type errors
- *frame_tag_error_A* Counts number of RXA frame tag errors
- *user_data_error_A* Counts number of RXA user data errors
- *data_error_A* Counts number of RXA data packet errors
- *dataFrameCntr_B* Number of Data frame received on FSI RXB
- *frame_type_error_B* Counts number of RXB frame type errors
- *frame_tag_error_B* Counts number of RXB frame tag errors
- *user_data_error_B* Counts number of RXB user data errors
- *data_error_B* Counts number of RXB data packet errors
- *dataFrameCntr_C* Number of Data frame received on FSI RXC
- *frame_type_error_C* Counts number of RXC frame type errors
- *frame_tag_error_C* Counts number of RXC frame tag errors
- *user_data_error_C* Counts number of RXC user data errors
- *data_error_C* Counts number of RXC data packet errors

32.5.1.13 FSI daisy chain topology, lead device example

FILE: fsi_ex16_daisy_handshake_lead.c fsi_ex16_daisy_handshake_lead is for the lead device in the daisy-chain loop, fsi_ex16_daisy_handshake_node for the other N-1 devices(N>=2).

In the code, there are different settings provided: [#define FSI_DMA_ENABLE 0] represents FSI communication using CPU control. [#define FSI_DMA_ENABLE 1] represents FSI communication using DMA control, enabling FSIRX to trigger a DMA event and move the RX FSI data to the TX FSI buffer

In a real scenario two separate devices may power up in arbitrary order and there is a need to establish a clean communication link which ensures that receiver side is flushed to properly interpret the start of a new valid frame.

The node devices in the daisy chain topology respond to the handshake sequence and forwards the information to the next device in the chain.

After above synchronization steps, FSI Rx can be configured as per use case i.e. nWords, lane width, enabling events, etc and start the infinite transfers. More details on establishing the communication link can be found in the device TRM.

User can edit some of configuration parameters as per use case, similar to other examples.

nWords - Number of words per transfer may be from 1 -16 *nLanes* - Choice to select single or double lane for frame transfers *txUserData* - User data to be sent with Data frame *txDataFrameTag* - Frame tag used for Data transfers *txPingFrameTag* - Frame tag used for Ping transfers *txPingTimeRefCntr* - Tx Ping timer reference counter *rxWdTimeoutRefCntr* - Rx Watchdog timeout reference counter

External Connections
For the FSI daisy-chain topology external connections are required to be made between the devices in the chain. Each devices FSI TX pins need to be connected to the FSI RX pins of the next device in the chain (or ring). See below for external connections to include and GPIOs used:

External Connections Required:
- FSIRX_CLK to FSITX_CLK
- FSIRX_RX0 to FSITX_TX0
- FSIRX_RX1 to FSITX_TX1

- ControlCard FSI Header GPIOs:

GPIO_27 -> FSITXA_CLK
- GPIO_26 -> FSITXA_TX0
- GPIO_25 -> FSITXA_TX1
- GPIO_9 -> FSIRXA_CLK
- GPIO_8 -> FSIRXA_RX0
- GPIO_10 -> FSIRXA_RX1

Watch Variables
- *dataFrameCntr* Number of Data frames received back
- *error* Non zero for transmit/receive data mismatch

32.5.1.14 FSI daisy chain topology, node device example

FILE: fsi_ex16_daisy_handshake_node.c fsi_ex16_daisy_handshake_lead is for the lead device in the daisy-chain loop, fsi_ex16_daisy_handshake_node for the other N-1 devices(N>=2).

In the code, there are different settings provided: [#define FSI_DMA_ENABLE 0] represents FSI communication using CPU control. [#define FSI_DMA_ENABLE 1] represents FSI communication using DMA control, enabling FSIRX to trigger a DMA event and move the RX FSI data to the TX FSI buffer

In a real scenario two separate devices may power up in arbitrary order and there is a need to establish a clean communication link which ensures that receiver side is flushed to properly interpret the start of a new valid frame.

The node devices in the daisy chain topology respond to the handshake sequence and forwards the information to the next device in the chain.

After above synchronization steps, FSI Rx can be configured as per use case i.e. nWords, lane width, enabling events, etc and start the infinite transfers. More details on establishing the communication link can be found in the device TRM.

User can edit some of configuration parameters as per use case, similar to other examples.

nWords - Number of words per transfer may be from 1 -16 *nLanes* - Choice to select single or double lane for frame transfers *txUserData* - User data to be sent with Data frame *txDataFrameTag* - Frame tag used for Data transfers *txPingFrameTag* - Frame tag used for Ping transfers *txPingTimeRefCntr* - Tx Ping timer reference counter *rxWdTimeoutRefCntr* - Rx Watchdog timeout reference counter

External Connections
For the FSI daisy-chain topology external connections are required to be made between the devices in the chain. Each devices FSI TX pins need to be connected to the FSI RX pins of the next device in the chain (or ring). See below for external connections to include and GPIOs used:

External Connections Required:
- FSIRX_CLK to FSITX_CLK
- FSIRX_RX0 to FSITX_TX0
- FSIRX_RX1 to FSITX_TX1

- ControlCard FSI Header GPIOs:

GPIO_27 -> FSITXA_CLK
- GPIO_26 -> FSITXA_TX0
- GPIO_25 -> FSITXA_TX1
- GPIO_9 -> FSIRXA_CLK
- GPIO_8 -> FSIRXA_RX0
- GPIO_10 -> FSIRXA_RX1

Watch Variables
- *dataFrameCntr* Number of Data frames received back
- *error* Non zero for transmit/receive data mismatch

32.6 FSI Registers

This section describes the Fast Serial Interface Registers. The FSI module contains two distinct sets of registers. One for the FSI receiver and one for the FSI transmitter.

32.6.1 FSI Base Address Table (C28)

Table 32-16. FSI Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
FsiTxaRegs	FSI_TX_REGS	FSITXA_BASE	0x0000_6600	YES	YES	YES	YES	YES
FsiRxaRegs	FSI_RX_REGS	FSIRXA_BASE	0x0000_6680	YES	YES	YES	YES	YES
FsiTxbRegs	FSI_TX_REGS	FSITXB_BASE	0x0000_6700	YES	YES	YES	YES	YES
FsiRxbRegs	FSI_RX_REGS	FSIRXB_BASE	0x0000_6780	YES	YES	YES	YES	YES
FsiRxcRegs	FSI_RX_REGS	FSIRXC_BASE	0x0000_6880	YES	YES	YES	YES	YES
FsiRxdRegs	FSI_RX_REGS	FSIRXD_BASE	0x0000_6980	YES	YES	YES	YES	YES
FsiRxeRegs	FSI_RX_REGS	FSIRXE_BASE	0x0000_6A80	YES	YES	YES	YES	YES
FsiRxfRegs	FSI_RX_REGS	FSIRXF_BASE	0x0000_6B80	YES	YES	YES	YES	YES
FsiRxgRegs	FSI_RX_REGS	FSIRXG_BASE	0x0000_6C80	YES	YES	YES	YES	YES
FsiRxhRegs	FSI_RX_REGS	FSIRXH_BASE	0x0000_6D80	YES	YES	YES	YES	YES

32.6.2 FSI_TX_REGS Registers

Table 32-17 lists the memory-mapped registers for the FSI_TX_REGS registers. All register offset addresses not listed in Table 32-17 should be considered as reserved locations and the register contents should not be modified.

Table 32-17. FSI_TX_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	TX_MASTER_CTRL	Transmit master control register	EALLOW	Go
2h	TX_CLK_CTRL	Transmit clock control register	EALLOW and LOCK	Go
4h	TX_OPER_CTRL_LO	Transmit operation control register low	EALLOW and LOCK	Go
5h	TX_OPER_CTRL_HI	Transmit operation control register high	EALLOW and LOCK	Go
6h	TX_FRAME_CTRL	Transmit frame control register		Go
7h	TX_FRAME_TAG_UDATA	Transmit frame tag and user data register		Go
8h	TX_BUF_PTR_LOAD	Transmit buffer pointer control load register	EALLOW	Go
9h	TX_BUF_PTR_STS	Transmit buffer pointer control status register		Go
Ah	TX_PING_CTRL	Transmit ping control register	EALLOW and LOCK	Go
Bh	TX_PING_TAG	Transmit ping tag register		Go
Ch	TX_PING_TO_REF	Transmit ping timeout counter reference	EALLOW and LOCK	Go
Eh	TX_PING_TO_CNT	Transmit ping timeout current count		Go
10h	TX_INT_CTRL	Transmit interrupt event control register	EALLOW and LOCK	Go
11h	TX_DMA_CTRL	Transmit DMA event control register	EALLOW and LOCK	Go
12h	TX_LOCK_CTRL	Transmit lock control register	EALLOW and LOCK	Go
14h	TX_EVT_STS	Transmit event and error status flag register		Go
16h	TX_EVT_CLR	Transmit event and error clear register	EALLOW	Go
17h	TX_EVT_FRC	Transmit event and error flag force register	EALLOW	Go
18h	TX_USER_CRC	Transmit user-defined CRC register		Go
20h	TX_ECC_DATA	Transmit ECC data register		Go
22h	TX_ECC_VAL	Transmit ECC value register		Go
40h + formula	TX_BUF_BASE_y	Base address for transmit buffer		Go

Complex bit access types are encoded to fit into small table cells. Table 32-18 shows the codes that are used for access types in this section.

Table 32-18. FSI_TX_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 32-18. FSI_TX_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

32.6.2.1 TX_MASTER_CTRL Register (Offset = 0h) [Reset = 0h]

TX_MASTER_CTRL is shown in Figure 32-19 and described in Table 32-19.

Return to the Summary Table.

Transmit master control register

Figure 32-19. TX_MASTER_CTRL Register

15	14	13	12	11	10	9	8
KEY							
W-0h							
7	6	5	4	3	2	1	0
RESERVED						FLUSH	CORE_RST
R-0h						R/W-0h	R/W-0h

Table 32-19. TX_MASTER_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	KEY	W	0h	Write Key In order to write to any bit in this register, 0xA5 must be written to this field at the same time. Otherwise, writes are ignored. The key is cleared immediately after writing, so it must be written again for every change to this register. Reset type: SYSRSn
7-2	RESERVED	R	0h	Reserved
1	FLUSH	R/W	0h	Flush Operation Start bit This bit will cause the transmitter to initiate a flush pattern of a single toggle on the TXD0 and TXD1 followed by five full cycles of TXCLK. This bit should be written only when the CORE_RST bit is 0 and the clock to the Transmitter core is turned on. 0h (R/W) = Clear this bit. 1h (R/W) = Setting this bit will Initiate flush sequence. To properly execute a flush sequence, Set FLUSH to 1, wait for five TXCLK cycles then clear FLUSH to 0. Note: The KEY field must contain 0xA5 for any write to this bit to take effect. The software must keep this bit set to 1 for at least five TXCLK cycles before setting it back to 0. Reset type: SYSRSn
0	CORE_RST	R/W	0h	Transmitter Main Core Reset bit This bit controls the transmitter main core reset. In order to send any frame, this bit must be cleared. 0h (R/W) = Transmitter core is not in reset and can transmit frames. 1h (R/W) = Transmitter core is held in reset. Note: The KEY field must contatin 0xA5 for any write to this bit to take effect. Reset type: SYSRSn

32.6.2.2 TX_CLK_CTRL Register (Offset = 2h) [Reset = 0h]

TX_CLK_CTRL is shown in Figure 32-20 and described in Table 32-20.

Return to the Summary Table.

Transmit clock control register

Figure 32-20. TX_CLK_CTRL Register

15	14	13	12	11	10	9	8
RESERVED						PRESCALE_VAL	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
PRESCALE_VAL						CLK_EN	CLK_RST
R/W-0h						R/W-0h	R/W-0h

Table 32-20. TX_CLK_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-2	PRESCALE_VAL	R/W	0h	Clock Divider Prescale Value The input clock is divided by this 8-bit value and fed into the transmitter core. This divided clock is the rate at which TXCLK will operate. 0h (R/W) = Reserved 1h (R/W) = Input clock /1 2h (R/W) = Input clock /2 3h (R/W) = Input clock /3 4h (R/W) = Input clock /4 ... FFh (R/W) = Input clock /255 TXCLKIN = Input clock / PRESCALE_VAL In FSI mode: TXCLK = TXCLKIN / 2 In SPI mode: TXCLK = TXCLKIN Reset type: SYSRSn
1	CLK_EN	R/W	0h	Clock Divider Enable bit This bit will enable and disable the input clock divider and start the clock to the transmitter core. 0h (R/W) = The input clock divider is not enabled and the clock is not connected to the transmitter core. 1h (R/W) = The input clock to the transmitter core is being divided by the PRESCALE_VAL and enabled. Reset type: SYSRSn
0	CLK_RST	R/W	0h	Clock Divider Reset bit This bit will reset the clock counter in the clock divider. 0h (R/W) = The clock divider is set based on the value in PRESCALE_VAL. The input clock will be divided by PRESCALE_VAL if CLK_EN is set. 1h (R/W) = The clock divider will be reset to 0 and will stay reset until software writes a 0 to this bit. Reset type: SYSRSn

32.6.2.3 TX_OPER_CTRL_LO Register (Offset = 4h) [Reset = 0h]

TX_OPER_CTRL_LO is shown in Figure 32-21 and described in Table 32-21.

Return to the Summary Table.

Transmit operation control register low

Figure 32-21. TX_OPER_CTRL_LO Register

15	14	13	12	11	10	9	8
RESERVED						TDM_ENABLE	SEL_PLLCLK
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
PING_TO_MODE	SW_CRC	START_MODE			SPI_MODE	DATA_WIDTH	
R/W-0h	R/W-0h	R/W-0h			R/W-0h	R/W-0h	

Table 32-21. TX_OPER_CTRL_LO Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9	TDM_ENABLE	R/W	0h	Transmit TDM Mode Enable bit. This bit enables the TDM Mode for multi-remote TDM operation. 0h (R/W) Transmit TDM Mode is not enabled. 1h (R/W) Transmit TDM Mode is enabled. Reset type: SYSRSn
8	SEL_PLLCLK	R/W	0h	Input Clock Select bit This bit selects the input clock source for the transmitter core. 0h (R/W) = SYSCLK is the source of the transmitter clock into the clock prescaler. 1h (R/W) = PLLRAWCLK is the source of the transmitter core clock into the clock prescaler. Reset type: SYSRSn
7	PING_TO_MODE	R/W	0h	Ping Counter Reset Mode Select bit This bit selects when the ping counter will reset. 0h (R/W) = The ping counter will reset and restart only on hardware initiated ping frames, when ping counter has timed out. 1h (R/W) = The ping counter will reset and restart on any software initiated frame as well as a ping counter timeout Reset type: SYSRSn
6	SW_CRC	R/W	0h	CRC Source Select bit This bit selects the source of the CRC value that is transmitted. 0h (R/W) = The transmitted CRC value is computed by hardware. 1h (R/W) = The transmitted CRC value is sourced from the value programmed in the TX_USER_CRC register. Reset type: SYSRSn
5-3	START_MODE	R/W	0h	Transmission Start Mode Select bit These bits select the method by which a new frame transmission is started. 0h (R/W) = Only a software write to TX_FRAME_CTRL.START initiate a new transmission. 1h (R/W) = The configured external trigger will initiate a new transmission. 2h (R/W) = Either writing to TX_FRAME_CTRL.START or the TX_FRAME_TAG_UDATA register will initiate a new transmission. All other combinations of bits are illegal and reserved for future use. Reset type: SYSRSn
2	SPI_MODE	R/W	0h	SPI Mode Select bit This bit enables and disables SPI compatibility mode. 0h (R/W) = FSI is in normal mode of operation. 1h (R/W) = FSI is operating in SPI compatibility mode. Reset type: SYSRSn

Table 32-21. TX_OPER_CTRL_LO Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	DATA_WIDTH	R/W	0h	Transmit Data Width Select bits These bits define the number of data lines used by the transmitter. 0h (R/W) = Data will be transmitted on one data line (TXD0) 1h (R/W) = Data will be transmitted on two data lines (TXD0 and TXD1). The format of the data is described in the preceeding chapter. 2h, 3h (R/W) = Reserved Reset type: SYSRSn

32.6.2.4 TX_OPER_CTRL_HI Register (Offset = 5h) [Reset = 0h]

TX_OPER_CTRL_HI is shown in Figure 32-22 and described in Table 32-22.

Return to the Summary Table.

Transmit operation control register high

Figure 32-22. TX_OPER_CTRL_HI Register

15	14	13	12	11	10	9	8
RESERVED			EXT_TRIG_SEL				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
EXT_TRIG_SEL	ECC_SEL	FORCE_ERR	RESERVED				
R/W-0h	R/W-0h	R/W-0h	R-0h				

Table 32-22. TX_OPER_CTRL_HI Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-7	EXT_TRIG_SEL	R/W	0h	External Trigger Select bit These bits define which of the 64 external inputs will be used as the source for the external input trigger. 00h (R/W) = Trigger 1 is the source. 01h (R/W) = Trigger 2 is the source. 02h (R/W) = Trigger 3 is the source. ... 3Fh (R/W) = Trigger 64 is the source. Reset type: SYSRSn
6	ECC_SEL	R/W	0h	ECC Data Width Select bit This bit selects between 16-bit and 32-bit ECC computation. 0h (R/W) = 32-bit ECC is used. 1h (R/W) = 16-bit ECC is used. Reset type: SYSRSn
5	FORCE_ERR	R/W	0h	Error Frame Force bit This bit will force the the CRC value of the transmitted data frame to 0 whenever there is a buffer overrun or underrun condition. This can be used to force a corrupted CRC as the data is not guaranteed to be reliable. The receiver will treat the data as invalid and can handle this as needed. Note: DO NOT use FORCE_ERR if using the SW CRC mode (FSI Transmit). 0h (R/W) = The CRC will not be forced to 0. 1h (R/W) = The CRC will be forced to 0 in a buffer overrun or underrun condition. Reset type: SYSRSn
4-0	RESERVED	R	0h	Reserved

32.6.2.5 TX_FRAME_CTRL Register (Offset = 6h) [Reset = 0h]

TX_FRAME_CTRL is shown in Figure 32-23 and described in Table 32-23.

Return to the Summary Table.

Transmit frame control register

Figure 32-23. TX_FRAME_CTRL Register

15	14	13	12	11	10	9	8
START	RESERVED						
R/W-0h	R-0h						
7	6	5	4	3	2	1	0
N_WORDS				FRAME_TYPE			
R/W-0h				R/W-0h			

Table 32-23. TX_FRAME_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	START	R/W	0h	Start Transmission bit This bit will cause the FSI to start transmitting the next frame. 0h (R/W) = Writing a 0 to this bit will have no effect. 1h (R/W) = Start the next transmission. This bit will be cleared by hardware. Reset type: SYSRSn
14-8	RESERVED	R	0h	Reserved
7-4	N_WORDS	R/W	0h	Number of Words to be Transmitted This field defines the number of words which will be transmitted in a DATA_N_WORD frame. This is a user-defined field that must match the corresponding field in the receiver. Set this bitfield to be one less than the number of words to be transmitted. 0h (R/W) = 1 data word frame (16-bit data). 1h (R/W) = 2 data word frame (32-bit data). .. Fh (R/W) = 16 data word frame (256-bit data). Reset type: SYSRSn
3-0	FRAME_TYPE	R/W	0h	Transmit Frame Type This field determines the type of frame that will be transmitted next. 0000b (R/W) = Ping Frame. This frame can be sent either by software or automatically by hardware. 0100b (R/W) = DATA_1_WORD Frame. One word data frame (16-bit data). 0101b (R/W) = DATA_2_WORD Frame. Two word data frame (32-bit data). 0110b (R/W) = DATA_4_WORD Frame. Four word data frame (64-bit data). 0111b (R/W) = DATA_6_WORD Frame. Six word data frame (96-bit data). 0011b (R/W) = DATA_N_WORD Frame. The N_WORDS field will determine the number of words (1 to 16) to be sent. Both the transmitter and receiver must have the same value programmed. 1111b (R/W) = Error Frame. This frame can be used during error conditions or any condition where the transmitter wants to notify the receiver of a high priorty status. However, the user software is at liberty to use this for any purpose. 0001b, 0010b, and 1000b through 1110b are Reserved and should not be used. Reset type: SYSRSn

32.6.2.6 TX_FRAME_TAG_UDATA Register (Offset = 7h) [Reset = 0h]

TX_FRAME_TAG_UDATA is shown in Figure 32-24 and described in Table 32-24.

Return to the Summary Table.

Transmit frame tag and user data register

Figure 32-24. TX_FRAME_TAG_UDATA Register

15	14	13	12	11	10	9	8
USER_DATA							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED				FRAME_TAG			
R-0h				R/W-0h			

Table 32-24. TX_FRAME_TAG_UDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	USER_DATA	R/W	0h	User Data bits This is a user-defined value that will be loaded into the the user data phase of the frame. This 8-bit value can be used by the receiver for any application need. This value will not impact any hardware behavior. Reset type: SYSRSn
7-4	RESERVED	R	0h	Reserved
3-0	FRAME_TAG	R/W	0h	This will be used only for software initiated transmissions. Frame tag bits This is a user-defined value that will be loaded into the frame tag phase of the next transmission. The receiver may use the frame tag for any application need. This value will not impact any hardware behavior For external triggers do not use this register. Use the TX_PING_TAG register instead. Reset type: SYSRSn

32.6.2.7 TX_BUF_PTR_LOAD Register (Offset = 8h) [Reset = 0h]

TX_BUF_PTR_LOAD is shown in Figure 32-25 and described in Table 32-25.

Return to the Summary Table.

Transmit buffer pointer control load register

Figure 32-25. TX_BUF_PTR_LOAD Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				BUF_PTR_LOAD			
R-0h				R/W-0h			

Table 32-25. TX_BUF_PTR_LOAD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3-0	BUF_PTR_LOAD	R/W	0h	Buffer Pointer Load bits These bits are used to force the transmit buffer pointer to a desired index within the transmit buffer. The next transmission will begin picking data from this index and increment appropriately. This value will be reflected in TX_BUF_PTR_STS only after a minimum 3 SYSCLK cycles + 3 TXCLK cycles. This value should not be written while there is an active transmission as it may corrupt the ongoing frame or other undefined behavior. Reset type: SYSRSn

32.6.2.8 TX_BUF_PTR_STS Register (Offset = 9h) [Reset = 0h]

TX_BUF_PTR_STS is shown in Figure 32-26 and described in Table 32-26.

Return to the Summary Table.

Transmit buffer pointer control status register

Figure 32-26. TX_BUF_PTR_STS Register

15	14	13	12	11	10	9	8
RESERVED			CURR_WORD_CNT				
R-0h			R-0h				
7	6	5	4	3	2	1	0
RESERVED				CURR_BUF_PTR			
R-0h				R-0h			

Table 32-26. TX_BUF_PTR_STS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-8	CURR_WORD_CNT	R	0h	Words Remaining in the transmit buffer This value indicates the number of words present in the data buffer which have not yet been transmitted. This value is only valid when there is no active transmission. Note: This value will not be valid if there is a buffer overrun or underrun condition. Reset type: SYSRSn
7-4	RESERVED	R	0h	Reserved
3-0	CURR_BUF_PTR	R	0h	Current Buffer Pointer Index This bitfield will show the current index of the buffer pointer. This value is only valid when there is no active transmission. Reset type: SYSRSn

32.6.2.9 TX_PING_CTRL Register (Offset = Ah) [Reset = 0h]

TX_PING_CTRL is shown in Figure 32-27 and described in Table 32-27.

Return to the Summary Table.

Transmit ping control register

Figure 32-27. TX_PING_CTRL Register

15	14	13	12	11	10	9	8
RESERVED							EXT_TRIG_SEL
R-0h							R/W-0h
7	6	5	4	3	2	1	0
EXT_TRIG_SEL					EXT_TRIG_EN	TIMER_EN	CNT_RST
R/W-0h					R/W-0h	R/W-0h	R/W-0h

Table 32-27. TX_PING_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	RESERVED	R	0h	Reserved
8-3	EXT_TRIG_SEL	R/W	0h	External Trigger Select bits This bitfield will select one of the 64 external trigger inputs to as the source to generate a ping frame. A ping frame will only be generated if the EXT_TRIG_EN bit is set. 0h (R/W) = Trigger 1 will be used to generate a ping frame. 1h (R/W) = Trigger 2 will be used to generate a ping frame. .. 3Fh (R/W) = Trigger 64 will be used to generate a ping frame. Reset type: SYSRSn
2	EXT_TRIG_EN	R/W	0h	External Trigger Enable bit This bit will allow the external trigger logic to generate a ping frame. 0h (R/W) = External triggers will not be used to generate ping frames. 1h (R/W) = The selected external trigger (selected by EXT_TRIG_SEL bits) will be able to generate a ping frame. The ping timer will be ignored if this bit is set. Reset type: SYSRSn
1	TIMER_EN	R/W	0h	Ping Timer Enable bit This bit will enable the ping timer for generating periodic ping frames. 0h (R/W) = The ping timer is disabled and will not generate ping frames. 1h (R/W) = The ping timer is enabled and can be used to generate ping frames.Once the timer count reaches the value set by the TX_PING_TO_REF register, it will initiate a ping frame transmission. Note: If the ping timer is used, EXT_TRIG_EN should not be set as it will override this function. Reset type: SYSRSn
0	CNT_RST	R/W	0h	Ping Counter Reset bit Writing a 1 to this bit will reset the ping counter to 0. The counter will stay in reset as long as this bit is set to 1. This bit needs to be cleared to 0 to use the counter. 0h (R/W) = Clear the CNT_RST. 1h (R/W) = The ping counter will be reset to 0. Reset type: SYSRSn

32.6.2.10 TX_PING_TAG Register (Offset = Bh) [Reset = 0h]

TX_PING_TAG is shown in Figure 32-28 and described in Table 32-28.

Return to the Summary Table.

Transmit ping tag register

Figure 32-28. TX_PING_TAG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				TAG			
R-0h				R/W-0h			

Table 32-28. TX_PING_TAG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3-0	TAG	R/W	0h	Ping Frame Tag This field contains a 4-bit tag which will be sent in any ping frame that is initiated by an external trigger or the ping timer. This field is user-defined and can be set based on the application requirement. If a ping frame is generated manually, the transmitted tag will be from TX_FRAME_TAG_UDATA.FRAME_TAG, not this value. Reset type: SYSRSn

32.6.2.11 TX_PING_TO_REF Register (Offset = Ch) [Reset = 0h]

TX_PING_TO_REF is shown in Figure 32-29 and described in Table 32-29.

Return to the Summary Table.

Transmit ping timeout counter reference

Figure 32-29. TX_PING_TO_REF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TO_REF																															
R/W-0h																															

Table 32-29. TX_PING_TO_REF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TO_REF	R/W	0h	Ping Timer Reference Value. This is the 32-bit reference value for the ping timer. The timer will increment the counter starting from 0. When the reference value is reached, it will generate a timeout event, triggering a ping frame transmission. The counter will then reset to 0 and continue counting. Reset type: SYSRSn

32.6.2.12 TX_PING_TO_CNT Register (Offset = Eh) [Reset = 0h]

TX_PING_TO_CNT is shown in Figure 32-30 and described in Table 32-30.

Return to the Summary Table.

Transmit ping timeout current count

Figure 32-30. TX_PING_TO_CNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TO_CNT																															
R-0h																															

Table 32-30. TX_PING_TO_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TO_CNT	R	0h	Ping Timer Counter Value This register contains the current value of the ping timer counter. After reset, this counter will increment until it reaches the reference value (TX_PING_TO_REF), at which point it generates a ping frame transmission. After this point, the counter will reset to 0 and continue counting. This is a free-running counter Reset type: SYSRSn

32.6.2.13 TX_INT_CTRL Register (Offset = 10h) [Reset = 0h]

TX_INT_CTRL is shown in Figure 32-31 and described in Table 32-31.

Return to the Summary Table.

Transmit interrupt event control register

Figure 32-31. TX_INT_CTRL Register

15	14	13	12	11	10	9	8
RESERVED				INT2_EN_PING_TO	INT2_EN_BUF_OVERRUN	INT2_EN_BUF_UNDERRUN	INT2_EN_FRAME_DONE
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED				INT1_EN_PING_TO	INT1_EN_BUF_OVERRUN	INT1_EN_BUF_UNDERRUN	INT1_EN_FRAME_DONE
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 32-31. TX_INT_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	INT2_EN_PING_TO	R/W	0h	Enable PING Timer Interrupt to INT2 This bit allows the event to generate an interrupt on the INT2 line. 0h (R/W) = This event will not trigger an interrupt on TX_INT2. 1h (R/W) = The ping timer event will trigger an interrupt on TX_INT2. Reset type: SYSRSn
10	INT2_EN_BUF_OVERRUN	R/W	0h	Enable Buffer Overrun Interrupt to INT2 This bit allows the event to generate an interrupt on the INT2 line. 0h (R/W) = This event will not trigger an interrupt on TX_INT2. 1h (R/W) = A Buffer Overrun condition will trigger an interrupt on TX_INT2. Reset type: SYSRSn
9	INT2_EN_BUF_UNDERRUN	R/W	0h	Enable Buffer Underrun Interrupt to INT2 This bit allows the event to generate an interrupt on the INT2 line. 0h (R/W) = This event will not trigger an interrupt on TX_INT2. 1h (R/W) = A Buffer Underrun condition will trigger an interrupt on TX_INT2. Reset type: SYSRSn
8	INT2_EN_FRAME_DONE	R/W	0h	Enable Frame Done interrupt to INT2 This bit allows the event to generate an interrupt on the INT2 line. 0h (R/W) = This event will not trigger an interrupt on TX_INT2. 1h (R/W) = A Frame Done event will trigger an interrupt on TX_INT2. Reset type: SYSRSn
7-4	RESERVED	R	0h	Reserved
3	INT1_EN_PING_TO	R/W	0h	Enable Ping Timer Interrupt to INT1 This bit allows the event to generate an interrupt on the INT1 line. 0h (R/W) = This event will not trigger an interrupt on TX_INT1. 1h (R/W) = The ping timer event will trigger an interrupt on TX_INT1. Reset type: SYSRSn
2	INT1_EN_BUF_OVERRUN	R/W	0h	Enable Buffer Overrun Interrupt to INT1 This bit allows the event to generate an interrupt on the INT1 line. 0h (R/W) = This event will not trigger an interrupt on TX_INT1. 1h (R/W) = A Buffer Overrun condition will trigger an interrupt on TX_INT1. Reset type: SYSRSn
1	INT1_EN_BUF_UNDERRUN	R/W	0h	Enable Buffer Underrun Interrupt to INT1 This bit allows the event to generate an interrupt on the INT1 line. 0h (R/W) = This event will not trigger an interrupt on TX_INT1. 1h (R/W) = A Buffer Underrun condition will trigger an interrupt on TX_INT1. Reset type: SYSRSn

Table 32-31. TX_INT_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INT1_EN_FRAME_DONE	R/W	0h	Enable Frame Done interrupt to INT1 This bit allows the event to generate an interrupt on the INT1 line. 0h (R/W) = This event will not trigger an interrupt on TX_INT1. 1h (R/W) = A Frame Done event will trigger an interrupt on TX_INT1. Reset type: SYSRSn

32.6.2.14 TX_DMA_CTRL Register (Offset = 11h) [Reset = 0h]

TX_DMA_CTRL is shown in Figure 32-32 and described in Table 32-32.

Return to the Summary Table.

Transmit DMA event control register

Figure 32-32. TX_DMA_CTRL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							DMA_EVT_EN
R-0h							R/W-0h

Table 32-32. TX_DMA_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	DMA_EVT_EN	R/W	0h	DMA Event Enable bit This bit will enable the DMA event to be generated upon the completion of a transmit frame. 0h (R/W) = A DMA event will not be generated. 1h (R/W) = A DMA event will be generated upon the completion of a transmitted frame. Note: The DMA event will only be generated for data frames. Reset type: SYSRSn

32.6.2.15 TX_LOCK_CTRL Register (Offset = 12h) [Reset = 0h]

TX_LOCK_CTRL is shown in Figure 32-33 and described in Table 32-33.

Return to the Summary Table.

Transmit lock control register

Figure 32-33. TX_LOCK_CTRL Register

15	14	13	12	11	10	9	8
KEY							
W-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK
R-0h							R/W-0h

Table 32-33. TX_LOCK_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	KEY	W	0h	Write Key In order to write to this register, 0xA5 must be written to this field at the same time. Otherwise, writes are ignored. The key is cleared immediately after writing, so it must be written again for every change to this register. Reset type: SYSRSn
7-1	RESERVED	R	0h	Reserved
0	LOCK	R/W	0h	Control Register Lock Enable bit This bit locks the contents of all the transmit control registers that support a lock protection. Once locked, further writes will not take effect until a SYSRS has reset this register. Once set, further writes to this bit will be ignored. 0h (R/W) = Transmit control registers can be modified and are not locked. 1h (R/W) = Transmit control registers are locked and cannot be modified until this bit is cleared by SYSRS. Any further writes to this bit are ignored. Note: The KEY field must contatin 0xA5 for any write to this bit to take effect. Reset type: SYSRSn

32.6.2.16 TX_EVT_STS Register (Offset = 14h) [Reset = 0h]

TX_EVT_STS is shown in Figure 32-34 and described in Table 32-34.

Return to the Summary Table.

Transmit event and error status flag register

Figure 32-34. TX_EVT_STS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				PING_TRIGGERED	BUF_OVERRUN	BUF_UNDERRUN	FRAME_DONE
R-0h				R-0h	R-0h	R-0h	R-0h

Table 32-34. TX_EVT_STS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	PING_TRIGGERED	R	0h	Ping Frame Triggered Flag Bit This bit indicates that a ping frame has been triggered. This bit is set by hardware when either the ping timer or an external trigger event have occured. Software can also force this bit to get set by writing to the TX_EVT_FRC register. 0h (R) = A ping frame has not been triggered. 1h (R) = A ping frame has been triggered by either the ping timer or external trigger. To clear this bit, write to the corresponding bit in the TX_EVT_CLR register. Reset type: SYSRSn
2	BUF_OVERRUN	R	0h	Buffer Overrun Flag Bit This bit inditcates that buffer overrun has occured.Software can also force this bit to get set by writing to the TX_EVT_FRC register. 0h (R) = Buffer Overrun has not occured. 1h (R) = Buffer Overrun has occured. To clear this bit, write to the corresponding bit in the TX_EVT_CLR register. Reset type: SYSRSn
1	BUF_UNDERRUN	R	0h	Buffer Underrun Flag Bit This bit inditcates that buffer underrun has occured.Software can also force this bit to get set by writing to the TX_EVT_FRC register. 0h (R) = Buffer Underrun has not occured. 1h (R) = Buffer Underrun has occured. To clear this bit, write to the corresponding bit in the TX_EVT_CLR register. Reset type: SYSRSn
0	FRAME_DONE	R	0h	Frame Done Flag Bit This bit inditcates a Frame Done condition. This bit is set by hardware when a frame transmission has been completed. Software can also force this bit to get set by writing to the TX_EVT_FRC register. 0h (R) = Frame Done condition has not occured. 1h (R) = Frame Done condition has occured. To clear this bit, write to the corresponding bit in the TX_EVT_CLR register. Reset type: SYSRSn

32.6.2.17 TX_EVT_CLR Register (Offset = 16h) [Reset = 0h]

TX_EVT_CLR is shown in Figure 32-35 and described in Table 32-35.

Return to the Summary Table.

Transmit event and error clear register

Figure 32-35. TX_EVT_CLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				PING_TRIGGERED	BUF_OVERRUN	BUF_UNDERRUN	FRAME_DONE
R-0h				W-0h	W-0h	W-0h	W-0h

Table 32-35. TX_EVT_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	PING_TRIGGERED	W	0h	Ping Frame Triggered Flag Clear bit This bit clears the corresponding bit in the TX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the TX_EVT_STS register to 0. Note: This bit may not always be cleared when writing to the corresponding TX_EVT_CLR bit. If PING_TIMEOUT MODE is configured to be 0, a hardware ping timeout may occur when another frame is actively being transmitted. In this case, if this bit still shows as 1 after the clear bit is written then the ping frame has been triggered but not serviced. This bit does not indicate that the ping frame has been completely sent, only that it has been triggered by the timeout event. Reset type: SYSRSn
2	BUF_OVERRUN	W	0h	Buffer Overrun Flag Clear bit This bit clears the corresponding bit in the TX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the TX_EVT_STS register to 0. Reset type: SYSRSn
1	BUF_UNDERRUN	W	0h	Buffer Underrun Flag Clear bit This bit clears the corresponding bit in the TX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the TX_EVT_STS register to 0. Reset type: SYSRSn
0	FRAME_DONE	W	0h	Frame Done Flag Clear bit This bit clears the corresponding bit in the TX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the TX_EVT_STS register to 0. Reset type: SYSRSn

32.6.2.18 TX_EVT_FRC Register (Offset = 17h) [Reset = 0h]

TX_EVT_FRC is shown in Figure 32-36 and described in Table 32-36.

Return to the Summary Table.

Transmit event and error flag force register

Figure 32-36. TX_EVT_FRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				PING_TRIGGERED	BUF_OVERRUN	BUF_UNDERRUN	FRAME_DONE
R-0h				W-0h	W-0h	W-0h	W-0h

Table 32-36. TX_EVT_FRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	PING_TRIGGERED	W	0h	Ping Frame Triggered Flag Force bit This bit will cause the corresponding bit in the TX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding flag bit in the TX_EVT_STS Register. Reset type: SYSRSn
2	BUF_OVERRUN	W	0h	Buffer Overrun Flag Force bit This bit will cause the corresponding bit in the TX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (R/W) = Writing a 0 to this bit will have no effect. 1h (R/W) = Force the corresponding flag bit in the TX_EVT_STS Register. Reset type: SYSRSn
1	BUF_UNDERRUN	W	0h	Buffer Underrun Flag Force bit This bit will cause the corresponding bit in the TX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding flag bit in the TX_EVT_STS Register. Reset type: SYSRSn
0	FRAME_DONE	W	0h	Frame Done Flag Force bit This bit will cause the corresponding bit in the TX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding flag bit in the TX_EVT_STS Register. Reset type: SYSRSn

32.6.2.19 TX_USER_CRC Register (Offset = 18h) [Reset = 0h]

TX_USER_CRC is shown in Figure 32-37 and described in Table 32-37.

Return to the Summary Table.

Transmit user-defined CRC register

Figure 32-37. TX_USER_CRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
USER_CRC							
R/W-0h							

Table 32-37. TX_USER_CRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	USER_CRC	R/W	0h	User-defined CRC This register contains the 8-bit CRC value to be transmitted in the next frame if the transmission is set for user-defined CRC option (TX_OPER_CTRL_LO.SW_CRC = 1). This register is ignored if the hardware CRC generation is enabled. Reset type: SYSRSn

32.6.2.20 TX_ECC_DATA Register (Offset = 20h) [Reset = 0h]

TX_ECC_DATA is shown in Figure 32-38 and described in Table 32-38.

Return to the Summary Table.

Transmit ECC data register

Figure 32-38. TX_ECC_DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_HIGH																DATA_LOW															
R/W-0h																R/W-0h															

Table 32-38. TX_ECC_DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA_HIGH	R/W	0h	Upper 16 bits of ECC Data Writing to this bitfield will cause the ECC logic to compute the ECC(SEC-DED) the entire 32-bit register and update TX_ECC_VAL register with the results. Software should write to these 16 bits of the register in a 32-bit write when needing to compute ECC for 32-bits for the full TX_ECC_DATA register. Reset type: SYSRSn
15-0	DATA_LOW	R/W	0h	Lower 16 bits of ECC Data Writing to this bitfield will cause the ECC logic to compute the ECC(SEC-DED) for these 16 bits and update the TX_ECC_VAL register with the results. Software should write to these register bits as a 16-bit write when needing to compute ECC for 16-bits. Reset type: SYSRSn

32.6.2.21 TX_ECC_VAL Register (Offset = 22h) [Reset = Ch]

TX_ECC_VAL is shown in Figure 32-39 and described in Table 32-39.

Return to the Summary Table.

Transmit ECC value register

Figure 32-39. TX_ECC_VAL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	ECC_VAL						
R-0h	R-Ch						

Table 32-39. TX_ECC_VAL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6-0	ECC_VAL	R	Ch	Computed ECC Value This field contains the ECC value computed using SEC-DED either for 16-bit or 32-bit data in the TX_ECC_DATA register. Reset type: SYSRSn

32.6.2.22 TX_BUF_BASE_y Register (Offset = 40h + formula) [Reset = 0h]

TX_BUF_BASE_y is shown in Figure 32-40 and described in Table 32-40.

Return to the Summary Table.

Base address for transmit buffer

Offset = 40h + (y * 1h); where y = 0h to Fh

Figure 32-40. TX_BUF_BASE_y Register

15	14	13	12	11	10	9	8
BASE_ADDRESS							
R/W-0h							
7	6	5	4	3	2	1	0
BASE_ADDRESS							
R/W-0h							

Table 32-40. TX_BUF_BASE_y Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	BASE_ADDRESS	R/W	0h	Transmit Data Buffer Base Address This is the base address of the 16-word data buffer used by the transmitter. Reset type: SYSRSn

32.6.3 FSI_RX_REGS Registers

Table 32-41 lists the memory-mapped registers for the FSI_RX_REGS registers. All register offset addresses not listed in Table 32-41 should be considered as reserved locations and the register contents should not be modified.

Table 32-41. FSI_RX_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	RX_MASTER_CTRL	Receive main control register	EALLOW	Go
4h	RX_OPER_CTRL	Receive operation control register	EALLOW and LOCK	Go
6h	RX_FRAME_INFO	Receive frame control register		Go
7h	RX_FRAME_TAG_UDATA	Receive frame tag and user data register		Go
8h	RX_DMA_CTRL	Receive DMA event control register	EALLOW and LOCK	Go
Ah	RX_EVT_STS	Receive event and error status flag register		Go
Bh	RX_CRC_INFO	Receive CRC info of received and computed CRC		Go
Ch	RX_EVT_CLR	Receive event and error clear register	EALLOW	Go
Dh	RX_EVT_FRC	Receive event and error flag force register	EALLOW	Go
Eh	RX_BUF_PTR_LOAD	Receive buffer pointer load register	EALLOW	Go
Fh	RX_BUF_PTR_STS	Receive buffer pointer status register		Go
10h	RX_FRAME_WD_CTRL	Receive frame watchdog control register	EALLOW and LOCK	Go
12h	RX_FRAME_WD_REF	Receive frame watchdog counter reference	EALLOW and LOCK	Go
14h	RX_FRAME_WD_CNT	Receive frame watchdog current count		Go
16h	RX_PING_WD_CTRL	Receive ping watchdog control register	EALLOW and LOCK	Go
17h	RX_PING_TAG	Receive ping tag register		Go
18h	RX_PING_WD_REF	Receive ping watchdog counter reference	EALLOW and LOCK	Go
1Ah	RX_PING_WD_CNT	Receive pingwatchdog current count		Go
1Ch	RX_INT1_CTRL	Receive interrupt control register for RX_INT1	EALLOW and LOCK	Go
1Dh	RX_INT2_CTRL	Receive interrupt control register for RX_INT2	EALLOW and LOCK	Go
1Eh	RX_LOCK_CTRL	Receive lock control register		Go
20h	RX_ECC_DATA	Receive ECC data register		Go
22h	RX_ECC_VAL	Receive ECC value register		Go
24h	RX_ECC_SEC_DATA	Receive ECC corrected data register		Go
26h	RX_ECC_LOG	Receive ECC log and status register		Go
28h	RX_FRAME_TAG_CMP	Receive frame tag compare register	EALLOW and LOCK	Go
29h	RX_PING_TAG_CMP	Receive ping tag compare register	EALLOW and LOCK	Go
30h	RX_DLYLINE_CTRL	Receive delay line control register	EALLOW and LOCK	Go
38h	RX_VIS_1	Receive debug visibility register 1		Go
40h + formula	RX_BUF_BASE_y	Base address for receive data buffer	EALLOW and LOCK	Go

Complex bit access types are encoded to fit into small table cells. Table 32-42 shows the codes that are used for access types in this section.

Table 32-42. FSI_RX_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

32.6.3.1 RX_MASTER_CTRL Register (Offset = 0h) [Reset = 0h]

RX_MASTER_CTRL is shown in Figure 32-41 and described in Table 32-43.

Return to the Summary Table.

Receive main control register

Figure 32-41. RX_MASTER_CTRL Register

15	14	13	12	11	10	9	8
KEY							
W-0h							
7	6	5	4	3	2	1	0
RESERVED					SPI_PAIRING	INT_LOOPBACK	CORE_RST
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 32-43. RX_MASTER_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	KEY	W	0h	Write Key. In order to write to this register, 0xA5 must be written to this field at the same time. Otherwise, writes are ignored. The key is cleared immediately after writing, so it must be written again for every change to this register. Reset type: SYSRSn
7-3	RESERVED	R	0h	Reserved
2	SPI_PAIRING	R/W	0h	Clock Pairing for SPI-like Behavior Enable bit This bit enables the internal clock pairing with the FSI TX module. This feature internally connects the TXCLK to RXCLK allowing the FSI TX module, acting as a SPI controller, to clock data into the receiver and out of the transmitter like a standard SPI module. This configuration is valid when the Module is in SPI mode only (RX_OPER_CTRL.SPI_MODE = 1) 0h (R/W) = SPI clock pairing is not enabled. 1h (R/W) = SPI clock pairing is enabled. The RXCLK will be internally connected to the TXCLK of the corresponding FSI module. Note: The KEY field must contatin 0xA5 for any write to this bit to take effect. Reset type: SYSRSn
1	INT_LOOPBACK	R/W	0h	Internal Loopback Enable bit This bit enables the internal loopback functionality of the FSI receiver. By enabling this bit, a mux will select the signals coming directly from the corresponding FSI transmitter module rather than from the pins. 0h (R/W) = Internal loopback is disabled. The FSI RX module will receive signals coming from the pins. 1h (R/W) = Internal loopback is enabled. The FSI RX module will receive signals from the directly from FSI TX module rather than the pins. Note: The KEY field must contatin 0xA5 for any write to this bit to take effect. Reset type: SYSRSn
0	CORE_RST	R/W	0h	Receiver Main Core Reset bit This bit controls the receiver main core reset. In order to receive any frame, this bit must be cleared. 0h (R/W) = Receiver core is not in reset and can receive frames. 1h (R/W) = Receiver core is held in reset. Note: The KEY field must contatin 0xA5 for any write to this bit to take effect. Reset type: SYSRSn

32.6.3.2 RX_OPER_CTRL Register (Offset = 4h) [Reset = 0h]

RX_OPER_CTRL is shown in Figure 32-42 and described in Table 32-44.

Return to the Summary Table.

Receive operation control register

Figure 32-42. RX_OPER_CTRL Register

15	14	13	12	11	10	9	8
RESERVED							PING_WD_RST_MODE
R-0h							R/W-0h
7	6	5	4	3	2	1	0
ECC_SEL	N_WORDS				SPI_MODE	DATA_WIDTH	
R/W-0h	R/W-0h				R/W-0h	R/W-0h	

Table 32-44. RX_OPER_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	RESERVED	R	0h	Reserved
8	PING_WD_RST_MODE	R/W	0h	Ping Watchdog Timeout Mode Select bit This bit selects the mode by which the ping watchdog counter is reset. The watchdog counter can be reset and restarted only by ping frames or by any received frame. 0h (R/W) = The ping watchdog counter will reset and restart only by ping frames. 1h (R/W) = The ping watchdog counter will reset and restart by any received frame. Reset type: SYSRSn
7	ECC_SEL	R/W	0h	ECC Data Width Select bit This bit selects between whether the ECC computation is done on 16-bit or 32-bit words. 0h (R/W) = 32-bit ECC is used. 1h (R/W) = 16-bit ECC is used. Reset type: SYSRSn
6-3	N_WORDS	R/W	0h	Number of Words to Receive This field defines the number of words which will be received in a DATA_N_WORD frame. This is a user-defined field that must match the corresponding field in the transmitter. Set this bitfield to be one less than the number of words to be received. This value is only applicable when the frame type received is DATA_N_WORD. 0h (R/W) = 1 data word frame (16-bit data). 1h (R/W) = 2 data word frame (32-bit data). .. Fh (R/W) = 16 data word frame (256-bit data). Reset type: SYSRSn
2	SPI_MODE	R/W	0h	SPI Mode Enable bit This bit enables and disables the SPI compatibility mode of the FSI RX. The received data must be formatted as an FSI frame in order for the data to properly be received. SPI compatibility mode will allow FSI RX to receive data that is sent using SPI signal format. Refer to the applicable section in the FSI TRM chapter for more information. 0h (R/W) = FSI is in normal mode of operation. 1h (R/W) = FSI is operating in SPI compatibility mode. Reset type: SYSRSn
1-0	DATA_WIDTH	R/W	0h	Receive Data Width Select bit These bits decide the number of data lines used for receiving data. 0h (R/W) = Data will be received on one data line, RXD0. 1h (R/W) = Data will be received on two data lines, RXD0 and RXD1. 2h, 3h (R/W) = Reserved Reset type: SYSRSn

32.6.3.3 RX_FRAME_INFO Register (Offset = 6h) [Reset = 0h]

RX_FRAME_INFO is shown in Figure 32-43 and described in Table 32-45.

Return to the Summary Table.

Receive frame control register

Figure 32-43. RX_FRAME_INFO Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				FRAME_TYPE			
R-0h				R-0h			

Table 32-45. RX_FRAME_INFO Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3-0	FRAME_TYPE	R	0h	Received Frame Type This field indicates the type of non-ping frame that was successfully received last. Note: Ping frame reception does not update this field, we want to retain the last successful non-ping frame FRAME_TYPE and PING_FRAME_RCVD flag already conveys PING info to the user. 0100b (R/W) = A DATA_1_WORD frame was received (16-bit data). 0101b (R/W) = A DATA_2_WORD frame was received (32-bit data). 0110b (R/W) = A DATA_4_WORD frame was received (64-bit data). 0111b (R/W) = A DATA_6_WORD frame was received (96-bit data). 0011b (R/W) = A DATA_N_WORD frame was received. The N_WORD field will determine the number of words (1 to 16) to be sent. The number of words received must equal the value programmed in RX_OPER_CTRL.N_WORDS. 1111b (R/W) = An error frame was received. This frame can be used during error conditions or any condition where the transmitter wants to signal the receiver for attention. However, the user software is at liberty to use this for any purpose. 0001b, 0010b, and 1000b through 1110b are Reserved and should not be used. Reset type: SYSRSn

32.6.3.4 RX_FRAME_TAG_UDATA Register (Offset = 7h) [Reset = 0h]

RX_FRAME_TAG_UDATA is shown in Figure 32-44 and described in Table 32-46.

Return to the Summary Table.

Receive frame tag and user data register

Figure 32-44. RX_FRAME_TAG_UDATA Register

15	14	13	12	11	10	9	8
USER_DATA							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			FRAME_TAG				RESERVED
R-0h			R-0h				R-0h

Table 32-46. RX_FRAME_TAG_UDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	USER_DATA	R	0h	Received User Data This field contains the 8-bit user data field of the last successfully received frame. Reset type: SYSRSn
7-5	RESERVED	R	0h	Reserved
4-1	FRAME_TAG	R	0h	Received Frame Tag This field contains the 4-bit frame tag from the last successfully received frame. This is intentionally shifted into bits 4:1 so that the register can be used as a 32-bit address index based on the received tag. Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

32.6.3.5 RX_DMA_CTRL Register (Offset = 8h) [Reset = 0h]

RX_DMA_CTRL is shown in Figure 32-45 and described in Table 32-47.

Return to the Summary Table.

Receive DMA event control register

Figure 32-45. RX_DMA_CTRL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							DMA_EVT_EN
R-0h							R/W-0h

Table 32-47. RX_DMA_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	DMA_EVT_EN	R/W	0h	DMA Event Enable bit This bit will enable a DMA Event to be generated upon the completion of a frame reception. 0h (R/W) = A DMA event will not be generated. 1h (R/W) = A DMA event will be generated upon the reception of a frame. Note: The DMA event will only be generated for data frames. Reset type: SYSRSn

32.6.3.6 RX_EVT_STS Register (Offset = Ah) [Reset = 0h]

RX_EVT_STS is shown in Figure 32-46 and described in Table 32-48.

Return to the Summary Table.

Receive event and error status flag register

Figure 32-46. RX_EVT_STS Register

15	14	13	12	11	10	9	8
RESERVED	ERROR_TAG_MATCH	DATA_TAG_MATCH	PING_TAG_MATCH	DATA_FRAME	FRAME_OVERRUN	PING_FRAME	ERR_FRAME
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
BUF_UNDERRUN	FRAME_DONE	BUF_OVERRUN	EOF_ERR	TYPE_ERR	CRC_ERR	FRAME_WD_TO	PING_WD_TO
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 32-48. RX_EVT_STS Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	ERROR_TAG_MATCH	R	0h	Error Tag Match Flag This bit indicates that an error frame was received with a tag comparison matching the masked tag reference. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = No tag-matched error frame received. 1h (R) = A tag-matched error frame has been received. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
13	DATA_TAG_MATCH	R	0h	Data Tag Match Flag This bit indicates that a dataframe was received with a tag comparison matching the masked tag reference. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = No tag-matched data frame received. 1h (R) = A tag-matched data frame has been received. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
12	PING_TAG_MATCH	R	0h	Ping Tag Match Flag This bit indicates that a ping frame was received with a tag comparison matching the masked tag reference. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = No tag-matched ping frame received. 1h (R) = A tag-matched ping frame has been received. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
11	DATA_FRAME	R	0h	Data Frame Received Flag This bit indicates that an data frame has been received. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = No data frame has been received. 1h (R) = A data frame has been received. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn

Table 32-48. RX_EVT_STS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	FRAME_OVERRUN	R	0h	Frame Overrun Flag This bit indicates that a frame overrun condition has occured. This bit gets set to 1 when a new DATA/ERROR frame is received and the corresponding DATA_FRAME_RCVD/ERROR_FRAME_RCVD flag is still set to 1. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = Frame overrun has not ocurred. 1h (R) = Frame overrun has ocurred. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
9	PING_FRAME	R	0h	Ping Frame Received Flag This bit indicates that an ping frame has been received. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = No ping frame has been received. 1h (R) = A ping frame has been received. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
8	ERR_FRAME	R	0h	Error Frame Received Flag This bit indicates that an error frame has been received. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = No error frame has been received. 1h (R) = An error frame has been received. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
7	BUF_UNDERRUN	R	0h	Receive Buffer Underrun Flag This bit indicates that a buffer underrun condition has occured in the receive buffer. This will happen when software reads the buffer which is empty and has no valid data. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = Receive Buffer Underrun has not ocurred. 1h (R) = Receive Buffer Underrun has ocurred. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
6	FRAME_DONE	R	0h	Frame Done Flag This bit indicates that a frame has been successfully received without error. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = No frame has been successfully received. 1h (R) = A frame has been successfully received. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
5	BUF_OVERRUN	R	0h	Receive Buffer Overrun Flag This bit indicates that a buffer overrun condition has occured in the receive buffer. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = Receive buffer overrun has not ocurred. 1h (R) = Receive buffer overrun has ocurred. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn

Table 32-48. RX_EVT_STS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	EOF_ERR	R	0h	End-of-Frame Error Flag This bit indicates that an invalid end-of-frame bit pattern has been received. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = Invalid end-of-frame has not been received. 1h (R) = Invalid end-of-frame has been received To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
3	TYPE_ERR	R	0h	Frame Type Error Flag This bit inditcates that an invalid frame type has been received. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = Invalid frame type has not been received. 1h (R) = Invalid frame type has been received To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
2	CRC_ERR	R	0h	CRC Error Flag This bit indicates that a CRC error has occured. A CRC error will be generated on a data frame where the received CRC and the computed CRC do not match. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = CRC error has not occured. 1h (R) = CRC error has occured. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
1	FRAME_WD_TO	R	0h	Frame Watchdog Timeout Flag This bit indicates that the frame watchdog timer has timed out. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = Frame watchdog timeout has not occured. 1h (R) = Frame watchdog timeout has occured. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
0	PING_WD_TO	R	0h	Ping Watchdog Timeout Flag This bit indicates that the ping watchdog timer has timed out. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = Ping watchdog timeout has not occured. 1h (R) = Ping watchdog timeout has occured. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn

32.6.3.7 RX_CRC_INFO Register (Offset = Bh) [Reset = 0h]

RX_CRC_INFO is shown in Figure 32-47 and described in Table 32-49.

Return to the Summary Table.

Receive CRC info of received and computed CRC

Figure 32-47. RX_CRC_INFO Register

15	14	13	12	11	10	9	8
CALC_CRC							
R-0h							
7	6	5	4	3	2	1	0
RX_CRC							
R-0h							

Table 32-49. RX_CRC_INFO Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	CALC_CRC	R	0h	Harware Calculated CRC Value This bitfield contains the CRC value that was calculated on the last received data. The contents of this bitfield are valid only when data frames are received. Note: The contents of this bitfield are invalid for ping and error frames. Reset type: SYSRSn
7-0	RX_CRC	R	0h	Received CRC Value This bitfield contains the CRC value that was last received a frame. The contents of this bitfield are valid only when data frames are received. Note: The contents of this bitfield are invalid for ping and error frames. Reset type: SYSRSn

32.6.3.8 RX_EVT_CLR Register (Offset = Ch) [Reset = 0h]

RX_EVT_CLR is shown in Figure 32-48 and described in Table 32-50.

Return to the Summary Table.

Receive event and error clear register

Figure 32-48. RX_EVT_CLR Register

15	14	13	12	11	10	9	8
RESERVED	ERROR_TAG_MATCH	DATA_TAG_MATCH	PING_TAG_MATCH	DATA_FRAME	FRAME_OVERRUN	PING_FRAME	ERR_FRAME
R-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
BUF_UNDERRUN	FRAME_DONE	BUF_OVERRUN	EOF_ERR	TYPE_ERR	CRC_ERR	FRAME_WD_TO	PING_WD_TO
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 32-50. RX_EVT_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	ERROR_TAG_MATCH	W	0h	Error Tag Match Glag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
13	DATA_TAG_MATCH	W	0h	Data Tag Match Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
12	PING_TAG_MATCH	W	0h	Ping Tag Match Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
11	DATA_FRAME	W	0h	Data Frame Received Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
10	FRAME_OVERRUN	W	0h	Frame Overrun Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
9	PING_FRAME	W	0h	Ping Frame Received Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn

Table 32-50. RX_EVT_CLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	ERR_FRAME	W	0h	Error Frame Received Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
7	BUF_UNDERRUN	W	0h	Receive Buffer Underrun Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (R/W) = Writing a 0 to this bit will have no effect. 1h (R/W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
6	FRAME_DONE	W	0h	Frame Done Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
5	BUF_OVERRUN	W	0h	Receive Buffer Overrun Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
4	EOF_ERR	W	0h	End-of-Frame Error Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
3	TYPE_ERR	W	0h	Frame Type Error Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
2	CRC_ERR	W	0h	CRC Error Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
1	FRAME_WD_TO	W	0h	Frame Watchdog Timeout Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
0	PING_WD_TO	W	0h	Ping Watchdog Timeout Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn

32.6.3.9 RX_EVT_FRC Register (Offset = Dh) [Reset = 0h]

RX_EVT_FRC is shown in Figure 32-49 and described in Table 32-51.

Return to the Summary Table.

Receive event and error flag force register

Figure 32-49. RX_EVT_FRC Register

15	14	13	12	11	10	9	8
RESERVED	ERROR_TAG_MATCH	DATA_TAG_MATCH	PING_TAG_MATCH	DATA_FRAME	FRAME_OVERRUN	PING_FRAME	ERR_FRAME
R-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
BUF_UNDERRUN	FRAME_DONE	BUF_OVERRUN	EOF_ERR	TYPE_ERR	CRC_ERR	FRAME_WD_TO	PING_WD_TO
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 32-51. RX_EVT_FRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	ERROR_TAG_MATCH	W	0h	Error Tag Match Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
13	DATA_TAG_MATCH	W	0h	Data Tag Match Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
12	PING_TAG_MATCH	W	0h	Ping Tag Match Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
11	DATA_FRAME	W	0h	Data Frame Received Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
10	FRAME_OVERRUN	W	0h	Frame Overrun Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn

Table 32-51. RX_EVT_FRC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	PING_FRAME	W	0h	Ping Frame Received Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
8	ERR_FRAME	W	0h	Error Frame Received Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
7	BUF_UNDERRUN	W	0h	Receive Buffer Underrun Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
6	FRAME_DONE	W	0h	Frame Done Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
5	BUF_OVERRUN	W	0h	Receive Buffer Overrun Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
4	EOF_ERR	W	0h	End-of-Frame Error Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
3	TYPE_ERR	W	0h	Frame Type Error Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
2	CRC_ERR	W	0h	CRC Error Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn

Table 32-51. RX_EVT_FRC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	FRAME_WD_TO	W	0h	Frame Watchdog Timeout Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
0	PING_WD_TO	W	0h	Ping Watchdog Timeout Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn

32.6.3.10 RX_BUF_PTR_LOAD Register (Offset = Eh) [Reset = 0h]

RX_BUF_PTR_LOAD is shown in Figure 32-50 and described in Table 32-52.

Return to the Summary Table.

Receive buffer pointer load register

Figure 32-50. RX_BUF_PTR_LOAD Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				BUF_PTR_LOAD			
R-0h				R/W-0h			

Table 32-52. RX_BUF_PTR_LOAD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3-0	BUF_PTR_LOAD	R/W	0h	Buffer Pointer Load. This is the value to be loaded into the receive word pointer when written. This is to allow software to force the receiver to start storing the received data starting at a specific location in the buffer. NOTE: The value of the CURR_BUF_PTR in the RX_BUF_PTR_STS will not get reflected immediately. This will take effect only when there is a valid receive operation with incoming clocks after (3 RXCLK + 3 SYCLK) cycles. Reset type: SYSRSn

32.6.3.11 RX_BUF_PTR_STS Register (Offset = Fh) [Reset = 0h]

RX_BUF_PTR_STS is shown in Figure 32-51 and described in Table 32-53.

Return to the Summary Table.

Receive buffer pointer status register

Figure 32-51. RX_BUF_PTR_STS Register

15	14	13	12	11	10	9	8
RESERVED			CURR_WORD_CNT				
R-0h			R-0h				
7	6	5	4	3	2	1	0
RESERVED				CURR_BUF_PTR			
R-0h				R-0h			

Table 32-53. RX_BUF_PTR_STS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-8	CURR_WORD_CNT	R	0h	Words Available in the Receive Buffer This bitfield indicates the number of valid data words present in the receive buffer that have not been read by the application software. This bitfield is only valid when there is no active transfer. Note: This value will not be valid if there has been a buffer overrun or underrun condition. Reset type: SYSRSn
7-4	RESERVED	R	0h	Reserved
3-0	CURR_BUF_PTR	R	0h	Current Buffer Pointer Index This bitfield will show the current index of the buffer pointer. This value is only valid when there is no active transmission. Reset type: SYSRSn

32.6.3.12 RX_FRAME_WD_CTRL Register (Offset = 10h) [Reset = 0h]

RX_FRAME_WD_CTRL is shown in Figure 32-52 and described in Table 32-54.

Return to the Summary Table.

Receive frame watchdog control register

Figure 32-52. RX_FRAME_WD_CTRL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						FRAME_WD_EN	FRAME_WD_CNT_RST
R-0h						R/W-0h	R/W-0h

Table 32-54. RX_FRAME_WD_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R	0h	Reserved
1	FRAME_WD_EN	R/W	0h	Frame Watchdog Counter Enable bit This bit will enable or disable the frame watchdog counter. The counter (RX_FRAME_WD_CNT) will begin counting from 0 when a valid start-of-frame pattern is received. When the reference value (RX_FRAME_WD_REF) is reached, it will generate a frame watchdog timeout event (RX_EVT_STS.FRAME_WD_TO) and the counter value will reset to 0 and continue counting on the next valid start-of-frame. 0h (R/W) = The frame watchdog counter is disabled and not running. 1h (R/W) = The frame watchdog counter logic is enabled and running. Reset type: SYSRSn
0	FRAME_WD_CNT_RST	R/W	0h	Frame Watchdog Counter Reset bit This bit will reset the frame watchdog counter to 0. Writing a 1 to this bit will reset the frame watchdog counter to 0. The counter will stay in reset as long as this bit is set to 1. This bit needs to be cleared to 0 to use the counter 0h (R/W) = Clear the FRAME_WD_CNT_RST. 1h (W) = The frame watchdog counter will be reset to 0. Reset type: SYSRSn

32.6.3.13 RX_FRAME_WD_REF Register (Offset = 12h) [Reset = 0h]

RX_FRAME_WD_REF is shown in Figure 32-53 and described in Table 32-55.

Return to the Summary Table.

Receive frame watchdog counter reference

Figure 32-53. RX_FRAME_WD_REF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRAME_WD_REF																															
R/W-0h																															

Table 32-55. RX_FRAME_WD_REF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FRAME_WD_REF	R/W	0h	Frame Watchdog Counter Reference Value This is the 32-bit reference value for the frame watchdog timeout counter. The counter will count up starting from 0 at a valid start-of-frame pattern and continue counting until this value is reached. Reset type: SYSRSn

32.6.3.14 RX_FRAME_WD_CNT Register (Offset = 14h) [Reset = 0h]

RX_FRAME_WD_CNT is shown in Figure 32-54 and described in Table 32-56.

Return to the Summary Table.

Receive frame watchdog current count

Figure 32-54. RX_FRAME_WD_CNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRAME_WD_CNT																															
R-0h																															

Table 32-56. RX_FRAME_WD_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FRAME_WD_CNT	R	0h	Frame Watchdog Counter Value This is the 32-bit read-only register which shows the current value of the frame watchdog counter. This counter is reset to 0 in a variety of ways: A write to FRME_WD_CNT_RST, a match with FRAME_WD_REF, or the reception of a successful data frame. Reset type: SYSRSn

32.6.3.15 RX_PING_WD_CTRL Register (Offset = 16h) [Reset = 0h]

RX_PING_WD_CTRL is shown in Figure 32-55 and described in Table 32-57.

Return to the Summary Table.

Receive ping watchdog control register

Figure 32-55. RX_PING_WD_CTRL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						PING_WD_EN	PING_WD_RST
R-0h						R/W-0h	R/W-0h

Table 32-57. RX_PING_WD_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R	0h	Reserved
1	PING_WD_EN	R/W	0h	Ping Watchdog Counter Enable bit This bit will enable or disable the ping watchdog counter. The counter (RX_PING_WD_CNT) will begin counting from 0 when it is enabled. When the reference value (RX_PING_WD_REF) is reached, it will generate a ping watchdog timeout event (RX_EVT_STS.PING_WD_TO) and the counter value will reset to 0, and resume counting 0h (R/W) = The ping watchdog counter is disabled and not running. 1h (R/W) = The ping watchdog counter logic is enabled and running. Reset type: SYSRSn
0	PING_WD_RST	R/W	0h	Ping Watchdog Counter Reset bit This bit will reset the ping watchdog counter to 0. Writing a 1 to this bit will reset the ping watchdog counter to 0. The counter will stay in reset as long as this bit is set to 1. This bit needs to be cleared to 0 to use the counter 0h (R/W) = Clear the PING_WD_RST. 1h (W) = The ping watchdog counter will be reset to 0. Reset type: SYSRSn

32.6.3.16 RX_PING_TAG Register (Offset = 17h) [Reset = 0h]

RX_PING_TAG is shown in Figure 32-56 and described in Table 32-58.

Return to the Summary Table.

Receive ping tag register

Figure 32-56. RX_PING_TAG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			PING_TAG				RESERVED
R-0h			R-0h				R-0h

Table 32-58. RX_PING_TAG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4-1	PING_TAG	R	0h	Received Ping Frame Tag This field contains the 4-bit frame tag from the last successfully received ping frame. This is intentionally shifted into bits 4:1 so that the register can be used as a 32-bit address index based on the received tag. Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

32.6.3.17 RX_PING_WD_REF Register (Offset = 18h) [Reset = 0h]

RX_PING_WD_REF is shown in Figure 32-57 and described in Table 32-59.

Return to the Summary Table.

Receive ping watchdog counter reference

Figure 32-57. RX_PING_WD_REF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PING_WD_REF																															
R/W-0h																															

Table 32-59. RX_PING_WD_REF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PING_WD_REF	R/W	0h	Ping Watchdog Counter Reference Value This is the 32-bit reference value for the ping watchdog timeout counter. The counter will count up starting from 0 and continue counting until this value is reached. Reset type: SYSRSn

32.6.3.18 RX_PING_WD_CNT Register (Offset = 1Ah) [Reset = 0h]

RX_PING_WD_CNT is shown in Figure 32-58 and described in Table 32-60.

Return to the Summary Table.

Receive pingwatchdog current count

Figure 32-58. RX_PING_WD_CNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PING_WD_CNT																															
R-0h																															

Table 32-60. RX_PING_WD_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PING_WD_CNT	R	0h	Ping Watchdog Counter Value This is the 32-bit read-only register which shows the current value of the ping watchdog counter. This counter is reset to 0 in a variety of ways: A write to PING_WD_RST, a match with PING_WD_REF, or the reception of a ping frame. Reset type: SYSRSn

32.6.3.19 RX_INT1_CTRL Register (Offset = 1Ch) [Reset = 0h]

RX_INT1_CTRL is shown in Figure 32-59 and described in Table 32-61.

Return to the Summary Table.

Receive interrupt control register for RX_INT1

Figure 32-59. RX_INT1_CTRL Register

15	14	13	12	11	10	9	8
RESERVED	INT1_EN_ERROR_TAG_MATCH	INT1_EN_DATA_TAG_MATCH	INT1_EN_PING_TAG_MATCH	INT1_EN_DATA_FRAME	INT1_EN_FRAME_OVERRUN	INT1_EN_PING_FRAME	INT1_EN_ERR_FRAME
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INT1_EN_UNDERRUN	INT1_EN_FRAME_DONE	INT1_EN_OVERRUN	INT1_EN_EOF_ERR	INT1_EN_TYPE_ERR	INT1_EN_CRC_ERR	INT1_EN_FRAME_WD_TO	INT1_EN_PING_WD_TO
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 32-61. RX_INT1_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	INT1_EN_ERROR_TAG_MATCH	R/W	0h	Enable Error Frame Received with Tag Match Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = An error frame received with matching tag will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
13	INT1_EN_DATA_TAG_MATCH	R/W	0h	Enable Data Frame Received with Tag Match Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A data frame received with matching tag will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
12	INT1_EN_PING_TAG_MATCH	R/W	0h	Enable Ping Frame Received with Tag Match Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A ping frame received with matching tag will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
11	INT1_EN_DATA_FRAME	R/W	0h	Enable Data Frame Received Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A data frame received event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
10	INT1_EN_FRAME_OVERRUN	R/W	0h	Enable Frame Overrun Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A frame overrun event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn

Table 32-61. RX_INT1_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	INT1_EN_PING_FRAME	R/W	0h	Enable Ping Frame Received Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A ping frame received event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
8	INT1_EN_ERR_FRAME	R/W	0h	Enable ERROR Frame Received Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A error frame received event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
7	INT1_EN_UNDERRUN	R/W	0h	Enable Buffer Underrun Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A buffer underrun event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
6	INT1_EN_FRAME_DONE	R/W	0h	Enable Frame Done Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A frame done event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
5	INT1_EN_OVERRUN	R/W	0h	Enable Receive Buffer Overrun Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A receive buffer overrun event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
4	INT1_EN_EOF_ERR	R/W	0h	Enable End-of-Frame Error Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = An end-of-frame error event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
3	INT1_EN_TYPE_ERR	R/W	0h	Enable Frame Type Error Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A frame type error event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn

Table 32-61. RX_INT1_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	INT1_EN_CRC_ERR	R/W	0h	Enable CRC Error Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A CRC error will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
1	INT1_EN_FRAME_WD_TO	R/W	0h	Enable Frame Watchdog Timeout Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A frame watchdog timeout event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
0	INT1_EN_PING_WD_TO	R/W	0h	Enable Ping Watchdog Timeout Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A ping watchdog timeout event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn

32.6.3.20 RX_INT2_CTRL Register (Offset = 1Dh) [Reset = 0h]

RX_INT2_CTRL is shown in Figure 32-60 and described in Table 32-62.

Return to the Summary Table.

Receive interrupt control register for RX_INT2

Figure 32-60. RX_INT2_CTRL Register

15	14	13	12	11	10	9	8
RESERVED	INT2_EN_ERROR_TAG_MATCH	INT2_EN_DATA_TAG_MATCH	INT2_EN_PING_TAG_MATCH	INT2_EN_DATA_FRAME	INT2_EN_FRAME_OVERRUN	INT2_EN_PING_FRAME	INT2_EN_ERR_FRAME
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INT2_EN_UNDERRUN	INT2_EN_FRAME_DONE	INT2_EN_OVERRUN	INT2_EN_EOF_ERR	INT2_EN_TYPE_ERR	INT2_EN_CRC_ERR	INT2_EN_FRAME_WD_TO	INT2_EN_PING_WD_TO
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 32-62. RX_INT2_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	INT2_EN_ERROR_TAG_MATCH	R/W	0h	Enable Error Frame Received with Tag Match Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = An error frame received with matching tag will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
13	INT2_EN_DATA_TAG_MATCH	R/W	0h	Enable Data Frame Received with Tag Match Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A data frame received with matching tag will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
12	INT2_EN_PING_TAG_MATCH	R/W	0h	Enable Ping Frame Received with Tag Match Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A ping frame received with matching tag will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
11	INT2_EN_DATA_FRAME	R/W	0h	Enable Data Frame Received Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A data frame received event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
10	INT2_EN_FRAME_OVERRUN	R/W	0h	Enable Frame Overrun Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A frame overrun event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn

Table 32-62. RX_INT2_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	INT2_EN_PING_FRAME	R/W	0h	Enable Ping Frame Received Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A ping frame received event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
8	INT2_EN_ERR_FRAME	R/W	0h	Enable Error Frame Received Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A error frame received event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
7	INT2_EN_UNDERRUN	R/W	0h	Enable Buffer Underrun Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A buffer underrun event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
6	INT2_EN_FRAME_DONE	R/W	0h	Enable Frame Done Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A frame done event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
5	INT2_EN_OVERRUN	R/W	0h	Enable Buffer Overrun Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A buffer overrun event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
4	INT2_EN_EOF_ERR	R/W	0h	Enable End-of-Frame Error Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = An end-of-frame error event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
3	INT2_EN_TYPE_ERR	R/W	0h	Enable Frame Type Error Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A frame type error event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn

Table 32-62. RX_INT2_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	INT2_EN_CRC_ERR	R/W	0h	Enable CRC Error Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A CRC error will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
1	INT2_EN_FRAME_WD_TO	R/W	0h	Enable Frame Watchdog Timeout Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A frame watchdog timeout event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
0	INT2_EN_PING_WD_TO	R/W	0h	Enable Ping Watchdog Timeout Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A ping watchdog timeout event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn

32.6.3.21 RX_LOCK_CTRL Register (Offset = 1Eh) [Reset = 0h]

RX_LOCK_CTRL is shown in Figure 32-61 and described in Table 32-63.

Return to the Summary Table.

Receive lock control register

Figure 32-61. RX_LOCK_CTRL Register

15	14	13	12	11	10	9	8
KEY							
W-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK
R-0h							R/W-0h

Table 32-63. RX_LOCK_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	KEY	W	0h	Write Key. In order to write to this register, 0xA5 must be written to this field at the same time. Otherwise, writes are ignored. The key is cleared immediately after writing, so it must be written again for every change to this register. Reset type: SYSRSn
7-1	RESERVED	R	0h	Reserved
0	LOCK	R/W	0h	Control Register Lock Enable bit This bit locks the contents of all the receive control registers that support a lock protection. Once locked, further writes will not take effect until SYSRS unlocks the register. Once set, further writes even to this bit will be ignored. 0h (R/W) = Receive control registers can be modified and are not locked. 1h (R/W) = Receive control registers are locked and cannot be modified until this bit is cleared by SYSRS. Any further writes to this bit are ignored. Note: The KEY field must contatin 0xA5 for any write to this bit to take effect. Reset type: SYSRSn

32.6.3.22 RX_ECC_DATA Register (Offset = 20h) [Reset = 0h]

RX_ECC_DATA is shown in Figure 32-62 and described in Table 32-64.

Return to the Summary Table.

Receive ECC data register

Figure 32-62. RX_ECC_DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_HIGH																DATA_LOW															
R/W-0h																R/W-0h															

Table 32-64. RX_ECC_DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA_HIGH	R/W	0h	Upper 16 bits of ECC Data Writing to this bitfield will cause the ECC logic to compute the ECC(SEC-DED) the entire 32-bit register and update TX_ECC_VAL register with the results. Software should write to these 16 bits of the register in a 32-bit write when needing to compute ECC for 32-bits for the full TX_ECC_DATA register. Reset type: SYSRSn
15-0	DATA_LOW	R/W	0h	Lower 16 bits of ECC Data Writing to this bitfield will cause the ECC logic to compute the ECC(SEC-DED) for these 16 bits and update the TX_ECC_VAL register with the results. Software should write to these register bits as a 16-bit write when needing to compute ECC for 16-bits. Reset type: SYSRSn

32.6.3.23 RX_ECC_VAL Register (Offset = 22h) [Reset = 0h]

RX_ECC_VAL is shown in Figure 32-63 and described in Table 32-65.

Return to the Summary Table.

Receive ECC value register

Figure 32-63. RX_ECC_VAL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	ECC_VAL						
R-0h	R/W-0h						

Table 32-65. RX_ECC_VAL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6-0	ECC_VAL	R/W	0h	ECC Value for SEC-DED check This field contains the ECC value to be used for SEC-DED either for 16-bit or 32-bit data in the RX_ECC_DATA register. Reset type: SYSRSn

32.6.3.24 RX_ECC_SEC_DATA Register (Offset = 24h) [Reset = 0h]

RX_ECC_SEC_DATA is shown in Figure 32-64 and described in Table 32-66.

Return to the Summary Table.

Receive ECC corrected data register

Figure 32-64. RX_ECC_SEC_DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEC_DATA																															
R-0h																															

Table 32-66. RX_ECC_SEC_DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SEC_DATA	R	0h	ECC Single Error Corrected Data The ECC corrected data will be available in this register. This value is valid only when there are no bit errors, or a single bit error was detected. Otherwise, the contents of this register are invalid and should not be used. Reset type: SYSRSn

32.6.3.25 RX_ECC_LOG Register (Offset = 26h) [Reset = 3h]

RX_ECC_LOG is shown in Figure 32-65 and described in Table 32-67.

Return to the Summary Table.

Receive ECC log and status register

Figure 32-65. RX_ECC_LOG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						MBE	SBE
R-0h						R-1h	R-1h

Table 32-67. RX_ECC_LOG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R	0h	Reserved
1	MBE	R	1h	Multiple Bit Errors Detected This bit indicates the occurrence of multiple bit errors.The data is corrupted and cannot be corrected. If this bit is set, the data present in RX_ECC_SEC_DATA is invalid and should not be used. 0h (R) Multiple Bit Errors were not detected. Check the SBE bit for single bit errors. 1h (R) Multiple Bit Errors were detected. The data is not able to be corrected. The value present in RX_ECC_SEC_DATA is invalid and should not be used. Reset type: SYSRSn
0	SBE	R	1h	Single Bit Error Detected This bit indicates the occurrence of a single bit error in the data. The data is autocorrected and placed into the RX_ECC_SEC_DATA register. This bit is valid only if MBE is 0. 0h (R) No bit errors were detected. The value in RX_ECC_SEC_DATA is correct. 1h (R) A single bit error was detected and corrected. The corrected data is present in RX_ECC_SEC_DATA. Reset type: SYSRSn

32.6.3.26 RX_FRAME_TAG_CMP Register (Offset = 28h) [Reset = 0h]

RX_FRAME_TAG_CMP is shown in Figure 32-66 and described in Table 32-68.

Return to the Summary Table.

Receive frame tag compare register

Figure 32-66. RX_FRAME_TAG_CMP Register

15	14	13	12	11	10	9	8
RESERVED						BROADCAST_EN	CMP_EN
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TAG_MASK				TAG_REF			
R/W-0h				R/W-0h			

Table 32-68. RX_FRAME_TAG_CMP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9	BROADCAST_EN	R/W	0h	Broadcast Enable bit This will enable the reception of a ping frame broadcast. When this bit is set, bit 3 of the received tag will be treated as a broadcast notification. If bit 3 of the received tag is set to 1, a ping tag match event will be triggered regardless of the. A match caused by the comparison of TAG_MASK and TAG_REF will still be considered a match and the frame tag match event will be triggered as normal This bit only takes effect only if CMP_EN is set to 1. 0h (R/W) Broadcast frame match disabled. 1h (R/W) Broadcast frame match enabled. Reset type: SYSRSn
8	CMP_EN	R/W	0h	Frame Tag Compare Enable bit Set this bit to enable the comparison of an incoming frame tag and the value stored in the frame tag reference. A match caused by the comparison of TAG_MASK, TAG_REF, and the incoming frame tag will trigger the apprpriate frame tag match event. 0h (R/W) Frame tag comparison is disabled. 1h (R/W) Frame tag comparison is enabled. Reset type: SYSRSn
7-4	TAG_MASK	R/W	0h	Frame Tag Mask Any bit position in this register set to 0 will be used in the comparison of the incoming frame tag and the value stored in TAG_REF. A bit position set to 1 will be ignored in the tag comparison. This mask value is used only for non-ping frames. Reset type: SYSRSn
3-0	TAG_REF	R/W	0h	Frame Tag Reference The reference tag to check against when comparing the TAG_MASK and the incoming frame tag. This reference value is used only for non-ping frames. Reset type: SYSRSn

32.6.3.27 RX_PING_TAG_CMP Register (Offset = 29h) [Reset = 0h]

RX_PING_TAG_CMP is shown in Figure 32-67 and described in Table 32-69.

Return to the Summary Table.

Receive ping tag compare register

Figure 32-67. RX_PING_TAG_CMP Register

15	14	13	12	11	10	9	8
RESERVED						BROADCAST_EN	CMP_EN
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TAG_MASK				TAG_REF			
R/W-0h				R/W-0h			

Table 32-69. RX_PING_TAG_CMP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9	BROADCAST_EN	R/W	0h	Broadcast Enable bit This will enable the reception of a ping frame broadcast. When this bit is set, bit 3 of the received tag will be treated as a broadcast notification. If bit 3 of the received tag is set to 1, a ping tag match event will be triggered regardless of the. A match caused by the comparison of TAG_MASK and TAG_REF will still be considered a match and the ping tag match event will be triggered as normal This bit only takes effect only if CMP_EN is set to 1. 0h (R/W) Broadcast frame match disabled. 1h (R/W) Broadcast frame match enabled. Reset type: SYSRSn
8	CMP_EN	R/W	0h	Ping Tag Compare Enable bit Set this bit to enable the comparison of an incoming ping tag and the value stored in the ping tag reference. A match caused by the comparison of TAG_MASK, TAG_REF, and the incoming ping tag will trigger a ping frame tag match event. 0h (R/W) Ping tag comparison is disabled. 1h (R/W) Ping tag comparison is enabled. Reset type: SYSRSn
7-4	TAG_MASK	R/W	0h	Ping Tag Mask Any bit position in this register set to 0 will be used in the comparison of the incoming ping frame tag and the value stored in TAG_REF. A bit position set to 1 will be ignored in the tag comparison. This mask value is used only for ping frames. Reset type: SYSRSn
3-0	TAG_REF	R/W	0h	Ping Tag Reference The reference tag to check against when comparing the TAG_MASK and the incoming ping tag. This reference value is used only for ping frames. Reset type: SYSRSn

32.6.3.28 RX_DLYLINE_CTRL Register (Offset = 30h) [Reset = 0h]

RX_DLYLINE_CTRL is shown in Figure 32-68 and described in Table 32-70.

Return to the Summary Table.

Receive delay line control register

Figure 32-68. RX_DLYLINE_CTRL Register

15	14	13	12	11	10	9	8
RESERVED	RXD1_DLY					RXD0_DLY	
R-0h	R/W-0h					R/W-0h	
7	6	5	4	3	2	1	0
RXD0_DLY			RXCLK_DLY				
R/W-0h			R/W-0h				

Table 32-70. RX_DLYLINE_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14-10	RXD1_DLY	R/W	0h	Delay Line Tap Select for RXD1 This bitfield selects the number of delay elements inserted into the RXD1 path from the pin boundary to the receiver core. 0h (R/W) Zero delay elements are included in the RXD1 path. RXD1 is taken directly from the pin. 1h (R/W) One delay element is included in the RXD1 path. 2h (R/W) Two delay elements are included in the RXD1 path. ... 1Fh (R/W) 31 delay elements are included in the RXD1 path, the maximum. Reset type: SYSRSn
9-5	RXD0_DLY	R/W	0h	Delay Line Tap Select for RXD0 This bitfield selects the number of delay elements inserted into the RXD0 path from the pin boundary to the receiver core. 0h (R/W) Zero delay elements are included in the RXD0 path. RXD0 is taken directly from the pin. 1h (R/W) One delay element is included in the RXD0 path. 2h (R/W) Two delay elements are included in the RXD0 path. ... 1Fh (R/W) 31 delay elements are included in the RXD0 path, the maximum. Reset type: SYSRSn
4-0	RXCLK_DLY	R/W	0h	Delay Line Tap Select for RXCLK This bitfield selects the number of delay elements inserted into the RXCLK path from the pin boundary to the receiver core. 0h (R/W) Zero delay elements are included in the RXCLK path. RXCLK is taken directly from the pin. 1h (R/W) One delay element is included in the RXCLK path. 2h (R/W) Two delay elements are included in the RXCLK path. ... 1Fh (R/W) 31 delay elements are included in the RXCLK path, the maximum. Reset type: SYSRSn

32.6.3.29 RX_VIS_1 Register (Offset = 38h) [Reset = 0h]

RX_VIS_1 is shown in Figure 32-69 and described in Table 32-71.

Return to the Summary Table.

Receive debug visibility register 1

Figure 32-69. RX_VIS_1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RX_CORE_STS	RESERVED		
R-0h				R-0h	R-0h		

Table 32-71. RX_VIS_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	RX_CORE_STS	R	0h	Receiver Core Status bit This bit indicates the status of the receiver core. If this bit is set, the receiver should undergo a reset and subsequent resynchronization with the transmitter. This bit will be always be set when the receiver has detected and end of frame error or a frame type error. This bit can also be set if the receiver becomes corrupted due to noise on the signal lines. If the receiver has experienced a ping watchdog or frame watchdog timeout, this bit should be read to determine if the cause was due to a corrupt transaction, thus putting the receiver core into an unrecoverable state. Only a soft reset will reset the recevier core and thus reset this bit. 0h (R) The receiver core is operating normally. 1h (R) The receiver core has entered into an error state and should be reset. Reset type: SYSRSn
2-0	RESERVED	R	0h	Reserved

32.6.3.30 RX_BUF_BASE_y Register (Offset = 40h + formula) [Reset = 0h]

RX_BUF_BASE_y is shown in Figure 32-70 and described in Table 32-72.

Return to the Summary Table.

Base address for receive data buffer

Offset = 40h + (y * 1h); where y = 0h to Fh

Figure 32-70. RX_BUF_BASE_y Register

15	14	13	12	11	10	9	8
BASE_ADDRESS							
R-0h							
7	6	5	4	3	2	1	0
BASE_ADDRESS							
R-0h							

Table 32-72. RX_BUF_BASE_y Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	BASE_ADDRESS	R	0h	Receive Data Buffer Base Address This is the base address of the 16-word data buffer used by the receiver. Reset type: SYSRSn

32.6.4 FSI Registers to Driverlib Functions

Table 32-73. FSI Registers to Driverlib Functions

File	Driverlib Function
TX_MAIN_CTRL	
fsi.c	FSI_resetTxModule
fsi.c	FSI_clearTxModuleReset
fsi.h	FSI_sendTxFlush
fsi.h	FSI_stopTxFlush
TX_CLK_CTRL	
fsi.c	FSI_resetTxModule
fsi.c	FSI_clearTxModuleReset
fsi.h	FSI_enableTxClock
fsi.h	FSI_disableTxClock
fsi.h	FSI_configPrescalar
TX_OPER_CTRL_LO	
fsi.h	FSI_selectTxPLLClock
fsi.h	FSI_setTxDataWidth
fsi.h	FSI_enableTxSPIMode
fsi.h	FSI_disableTxSPIMode
fsi.h	FSI_setTxStartMode
fsi.h	FSI_setTxPingTimeoutMode
fsi.h	FSI_enableTxTDMMode
fsi.h	FSI_disableTxTDMMode
fsi.h	FSI_enableTxUserCRC
fsi.h	FSI_disableTxUserCRC
TX_OPER_CTRL_HI	
fsi.h	FSI_setTxExtFrameTrigger

Table 32-73. FSI Registers to Driverlib Functions (continued)

File	Driverlib Function
fsi.h	FSI_enableTxCRCForceError
fsi.h	FSI_disableTxCRCForceError
fsi.h	FSI_setTxECCComputeWidth
TX_FRAME_CTRL	
fsi.h	FSI_setTxFrameType
fsi.h	FSI_setTxSoftwareFrameSize
fsi.h	FSI_startTxTransmit
TX_FRAME_TAG_UDATA	
fsi.h	FSI_setTxFrameTag
fsi.h	FSI_setTxUserDefinedData
TX_BUF_PTR_LOAD	
fsi.h	FSI_setTxBufferPtr
TX_BUF_PTR_STS	
fsi.h	FSI_getTxBufferPtr
fsi.h	FSI_getTxWordCount
TX_PING_CTRL	
fsi.c	FSI_resetTxModule
fsi.c	FSI_clearTxModuleReset
fsi.h	FSI_enableTxPingTimer
fsi.h	FSI_disableTxPingTimer
fsi.h	FSI_enableTxExtPingTrigger
fsi.h	FSI_disableTxExtPingTrigger
TX_PING_TAG	
fsi.h	FSI_enableTxPingTimer
fsi.h	FSI_setTxPingTag
TX_PING_TO_REF	
fsi.h	FSI_enableTxPingTimer
TX_PING_TO_CNT	
fsi.h	FSI_getTxCurrentPingTimeoutCounter
TX_INT_CTRL	
fsi.h	FSI_enableTxInterrupt
fsi.h	FSI_disableTxInterrupt
TX_DMA_CTRL	
fsi.h	FSI_enableTxDMAEvent
fsi.h	FSI_disableTxDMAEvent
TX_LOCK_CTRL	
fsi.h	FSI_lockTxCtrl
TX_EVT_STS	
fsi.h	FSI_getTxEventStatus
TX_EVT_CLR	
fsi.h	FSI_clearTxEvents
TX_EVT_FRC	
fsi.h	FSI_forceTxEvents
TX_USER_CRC	
fsi.h	FSI_enableTxUserCRC

Table 32-73. FSI Registers to Driverlib Functions (continued)

File	Driverlib Function
TX_ECC_DATA	
fsi.h	FSI_setTxECCdata
TX_ECC_VAL	
fsi.h	FSI_getTxECCValue
TX_BUF_BASE	
fsi.c	FSI_writeTxBuffer
fsi.h	FSI_getTxBufferAddress
RX_MAIN_CTRL	
fsi.c	FSI_resetRxModule
fsi.c	FSI_clearRxModuleReset
fsi.h	FSI_enableRxInternalLoopback
fsi.h	FSI_disableRxInternalLoopback
fsi.h	FSI_enableRxSPIPairing
fsi.h	FSI_disableRxSPIPairing
RX_OPER_CTRL	
fsi.h	FSI_setRxDataWidth
fsi.h	FSI_enableRxSPIMode
fsi.h	FSI_disableRxSPIMode
fsi.h	FSI_setRxSoftwareFrameSize
fsi.h	FSI_setRxECCComputeWidth
fsi.h	FSI_setRxPingTimeoutMode
RX_FRAME_INFO	
fsi.h	FSI_getRxFrameType
RX_FRAME_TAG_UDATA	
fsi.h	FSI_getRxFrameTag
fsi.h	FSI_getRxUserDefinedData
RX_DMA_CTRL	
fsi.h	FSI_enableRxDMAEvent
fsi.h	FSI_disableRxDMAEvent
RX_EVT_STS	
fsi.h	FSI_getRxEventStatus
RX_CRC_INFO	
fsi.h	FSI_getRxReceivedCRC
fsi.h	FSI_getRxComputedCRC
RX_EVT_CLR	
fsi.h	FSI_clearRxEvents
RX_EVT_FRC	
fsi.h	FSI_forceRxEvents
RX_BUF_PTR_LOAD	
fsi.h	FSI_setRxBufferPtr
RX_BUF_PTR_STS	
fsi.h	FSI_getRxBufferPtr
fsi.h	FSI_getRxWordCount
RX_FRAME_WD_CTRL	
fsi.c	FSI_resetRxModule

Table 32-73. FSI Registers to Driverlib Functions (continued)

File	Driverlib Function
fsi.c	FSI_clearRxModuleReset
fsi.h	FSI_enableRxFrameWatchdog
fsi.h	FSI_disableRxFrameWatchdog
RX_FRAME_WD_REF	
fsi.h	FSI_enableRxFrameWatchdog
RX_FRAME_WD_CNT	
fsi.h	FSI_getRxFrameWatchdogCounter
RX_PING_WD_CTRL	
fsi.c	FSI_resetRxModule
fsi.c	FSI_clearRxModuleReset
fsi.h	FSI_enableRxPingWatchdog
fsi.h	FSI_disableRxPingWatchdog
RX_PING_TAG	
fsi.h	FSI_getRxPingTag
fsi.h	FSI_setRxPingTagRef
fsi.h	FSI_getRxPingTagRef
fsi.h	FSI_setRxPingTagMask
fsi.h	FSI_getRxPingTagMask
fsi.h	FSI_enableRxPingTagCompare
fsi.h	FSI_disableRxPingTagCompare
fsi.h	FSI_enableRxPingBroadcast
fsi.h	FSI_disableRxPingBroadcast
RX_PING_WD_REF	
fsi.h	FSI_enableRxPingWatchdog
RX_PING_WD_CNT	
fsi.h	FSI_getRxPingWatchdogCounter
RX_INT1_CTRL	
fsi.h	FSI_enableRxInterrupt
fsi.h	FSI_disableRxInterrupt
RX_INT2_CTRL	
fsi.h	FSI_enableRxInterrupt
fsi.h	FSI_disableRxInterrupt
RX_LOCK_CTRL	
fsi.h	FSI_lockRxCtrl
RX_ECC_DATA	
fsi.h	FSI_setRxECCData
RX_ECC_VAL	
fsi.h	FSI_setRxReceivedECCValue
RX_ECC_SEC_DATA	
fsi.h	FSI_getRxECCCorrectedData
RX_ECC_LOG	
fsi.h	FSI_getRxECCLog
RX_FRAME_TAG_CMP	
fsi.h	FSI_setRxFrameTagRef
fsi.h	FSI_getRxFrameTagRef

Table 32-73. FSI Registers to Driverlib Functions (continued)

File	Driverlib Function
fsi.h	FSI_setRxFrameTagMask
fsi.h	FSI_getRxFrameTagMask
fsi.h	FSI_enableRxFrameTagCompare
fsi.h	FSI_disableRxFrameTagCompare
fsi.h	FSI_enableRxFrameBroadcast
fsi.h	FSI_disableRxFrameBroadcast
RX_PING_TAG_CMP	
fsi.h	FSI_setRxPingTagRef
fsi.h	FSI_getRxPingTagRef
fsi.h	FSI_setRxPingTagMask
fsi.h	FSI_getRxPingTagMask
fsi.h	FSI_enableRxPingTagCompare
fsi.h	FSI_disableRxPingTagCompare
fsi.h	FSI_enableRxPingBroadcast
fsi.h	FSI_disableRxPingBroadcast
RX_DLYLINE_CTRL	
fsi.c	FSI_configRxDelayLine
RX_VIS_1	
-	
RX_BUF_BASE	
fsi.c	FSI_readRxBuffer
fsi.h	FSI_getRxBufferAddress

This page intentionally left blank.

Chapter 33

Inter-Integrated Circuit Module (I2C)

This chapter describes the features and operation of the inter-integrated circuit (I2C) module. The I2C module provides an interface between one of these devices and devices compliant with the NXP Semiconductors Inter-IC bus (I2C bus) specification version 2.1, and connected by way of an I2C bus. External components attached to this 2-wire serial bus can transmit/receive 1- to 8-bit data to/from the device through the I2C module. This chapter assumes the reader is familiar with the I2C bus specification.

Note

A unit of data transmitted or received by the I2C module can have fewer than 8 bits; however, for convenience, a unit of data is called a data byte throughout this document. The number of bits in a data byte is selectable by way of the BC bits of the mode register, I2CMDR.

33.1 Introduction

The I2C module supports any slave or master I2C-compatible device. Figure 33-1 shows an example of multiple I2C modules connected for a two-way transfer from one device to other devices.

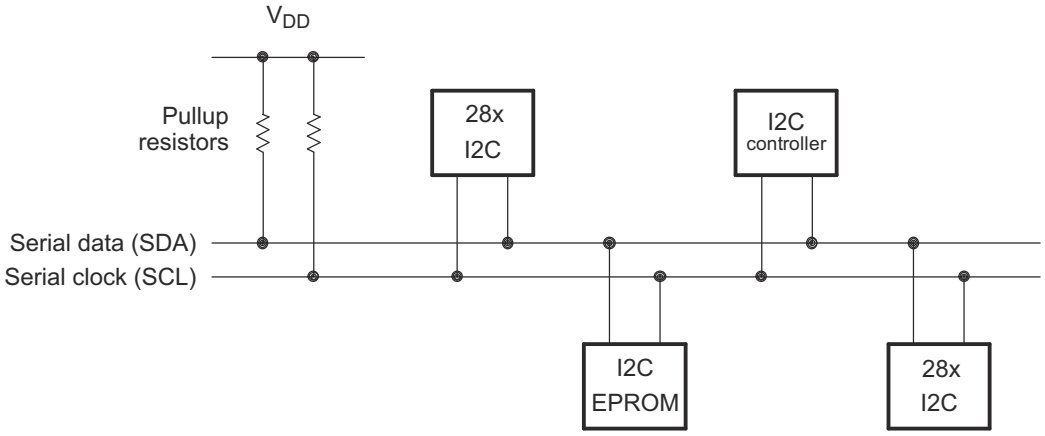


Figure 33-1. Multiple I2C Modules Connected

33.1.1 I2C Related Collateral

Foundational Materials

- C2000 Academy - I2C
- I2C Hardware Overview (Video)
- I2C Protocol Overview (Video)
- *Understanding the I2C Bus Application Report*

Getting Started Materials

- *Configuring the TMS320F280x DSP as an I2C Processor Application Report*
- I2C Buffers Overview (Video)
- *I2C Dynamic Addressing Application Report*
- I2C translators overview (Video)
- *Interfacing EEPROM Using C2000 I2C Module Application Report*
- *Why, When, and How to use I2C Buffers Application Report*

Expert Materials

- *I2C Bus Pull-Up Resistor Calculation Application Report*
- *Maximum Clock Frequency of I2C Bus Using Repeaters Application Report*

33.1.2 Features

The I2C module has the following features:

- Compliance with the NXP Semiconductors I2C bus specification (version 2.1):
  - Support for 8-bit format transfers
  - 7-bit and 10-bit addressing modes
  - General call
  - START byte mode
  - Support for multiple master-transmitters and slave-receivers
  - Support for multiple slave-transmitters and master-receivers
  - Combined master transmit/receive and receive/transmit mode
  - Data transfer rate from 10 kbps up to 400 kbps (Fast-mode)
- Receive FIFO and Transmitter FIFO (16-deep x 8-bit FIFO)
- Supports two ePIE interrupts:
  - I2Cx Interrupt – Any of the following events can be configured to generate an I2Cx interrupt:
    - Transmit-data ready
    - Receive-data ready
    - Register-access ready
    - No-acknowledgment received
    - Arbitration lost
    - Stop condition detected
    - Addressed as slave
  - I2Cx_FIFO interrupts:
    - Transmit FIFO interrupt
    - Receive FIFO interrupt
- Module enable and disable capability
- Free data format mode

33.1.3 Features Not Supported

The I2C module does not support:

- High-speed mode (Hs-mode)
- CBUS-compatibility mode

33.1.4 Functional Overview

Each device connected to an I2C bus is recognized by a unique address. Each device can operate as either a transmitter or a receiver, depending on the function of the device. A device connected to the I2C bus can also be considered as the master or the slave when performing data transfers. A master device is the device that initiates a data transfer on the bus and generates the clock signals to permit that transfer. During this transfer, any device addressed by this master is considered a slave. The I2C module supports the multi-master mode, in which one or more devices capable of controlling an I2C bus can be connected to the same I2C bus.

For data communication, the I2C module has a serial data pin (SDA) and a serial clock pin (SCL), as shown in Figure 33-2. These two pins carry information between the C28x device and other devices connected to the I2C bus. The SDA and SCL pins are both bidirectional and each must be connected to a positive supply voltage using a pull-up resistor. When the bus is free, both pins are high. The driver of these two pins has an open-drain configuration to perform the required wired-AND function.

There are two major transfer techniques:

- Standard Mode: Send exactly n data values, where n is a value you program in an I2C module register. See the I2CCNT register in Section 33.7 for more information.
- Repeat Mode: Keep sending data values until you use software to initiate a STOP condition or a new START condition. See the I2CMDR register in Section 33.7 for RM bit information.

The I2C module consists of the following primary blocks:

- A serial interface: one data pin (SDA) and one clock pin (SCL)
- Data registers and FIFOs to temporarily hold receive data and transmit data traveling between the SDA pin and the CPU
- Control and status registers
- A peripheral bus interface to enable the CPU to access the I2C module registers and FIFOs.
- A clock synchronizer to synchronize the I2C input clock (from the device clock generator) and the clock on the SCL pin, and to synchronize data transfers with masters of different clock speeds
- A prescaler to divide down the input clock that is driven to the I2C module
- A noise filter on each of the two pins, SDA and SCL
- An arbitrator to handle arbitration between the I2C module (when the I2C module is a master) and another master
- Interrupt generation logic, so that an interrupt can be sent to the CPU
- FIFO interrupt generation logic, so that FIFO access can be synchronized to data reception and data transmission in the I2C module

Figure 33-2 shows the four registers used for transmission and reception in non-FIFO mode. The CPU writes data for transmission to I2CDXR and reads received data from I2CDRR. When the I2C module is configured as a transmitter, data written to I2CDXR is copied to I2CXSR and shifted out on the SDA pin one bit at a time. When the I2C module is configured as a receiver, received data is shifted into I2CRSR and then copied to I2CDRR.

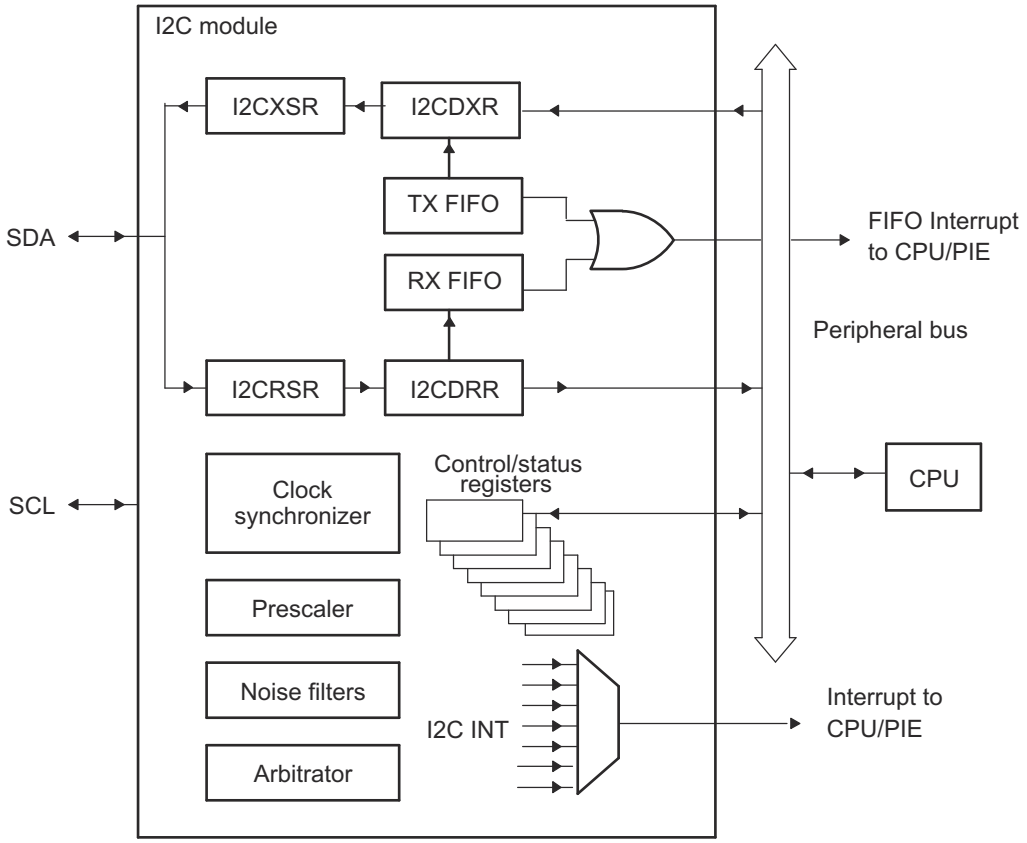


Figure 33-2. I2C Module Conceptual Block Diagram

33.1.5 Clock Generation

The I2C module clock determines the frequency at which the I2C module operates. A programmable prescaler in the I2C module divides down the SYSCLK to produce the I2C module clock and this I2C module clock is divided further to produce the I2C master clock on the SCL pin. Figure 33-3 shows the clock generation diagram for I2C module.

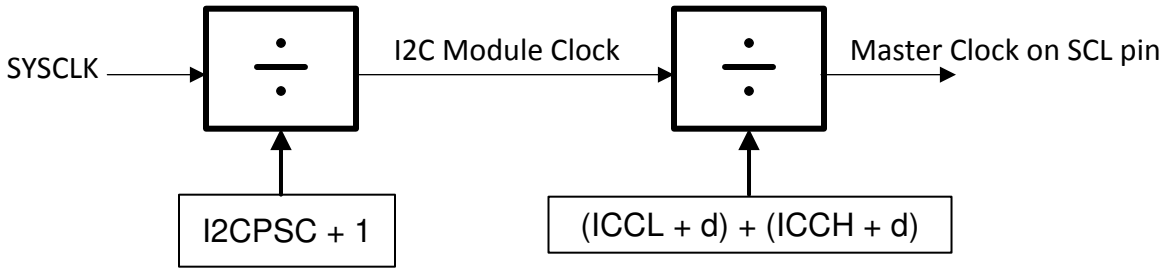


Figure 33-3. Clocking Diagram for the I2C Module

Note

To meet all of the I2C protocol timing specifications, the I2C module clock must be between 7-12 MHz.

To specify the divide-down value, initialize the IPSC field of the prescaler register, I2CPSC. The resulting frequency is:

$$\text{I2C Module Clock (Fmod)} = \frac{\text{SYSCLK}}{(\text{I2CPSC} + 1)} \quad (22)$$

The prescaler must be initialized only while the I2C module is in the reset state (IRS = 0 in I2CMDR). The prescaled frequency takes effect only when IRS is changed to 1. Changing the IPSC value while IRS = 1 has no effect.

The master clock appears on the SCL pin when the I2C module is configured to be a master on the I2C bus. This clock controls the timing of communication between the I2C module and a slave. As shown in Figure 33-3, a second clock divider in the I2C module divides down the module clock to produce the master clock. The clock divider uses the ICCL value of I2CCLKL to divide down the low portion of the module clock signal and uses the ICCH value of I2CCLKH to divide down the high portion of the module clock signal. See Section 33.1.6 for the master clock frequency equation.

33.1.6 I2C Clock Divider Registers (I2CCLKL and I2CCLKH)

As explained in Section 33.1.5, when the I2C module is a master, the I2C module clock is divided down further to use as the master clock on the SCL pin. As shown in Figure 33-4, the shape of the master clock depends on two divide-down values:

- ICCL in I2CCLKL. For each master clock cycle, ICCL determines the amount of time the signal is low.
- ICCH in I2CCKLH. For each master clock cycle, ICCH determines the amount of time the signal is high.

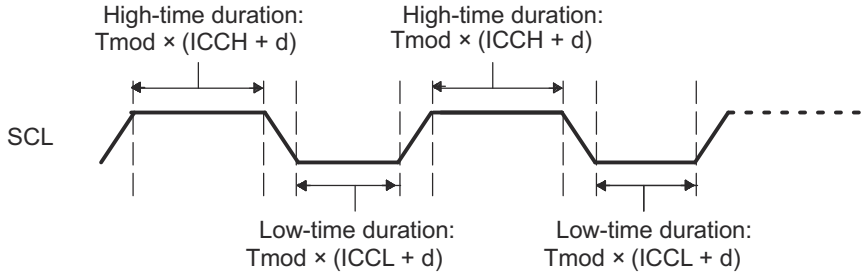


Figure 33-4. Roles of the Clock Divide-Down Values (ICCL and ICCH)

33.1.6.1 Formula for the Master Clock Period

The master clock period (Tmst) is a multiple of the period of the I2C Module Clock (Tmod):

$$\text{Master Clock period (Tmst)} = \frac{[\,(\text{ICCH} + d) + (\text{ICCL} + d)\,]}{\text{I2C Module Clock (Fmod)}} \quad (23)$$

where d depends on the divide-down value IPSC, as shown in Table 33-1. IPSC is described in the I2CPSC register.

Table 33-1. Dependency of Delay d on the Divide-Down Value IPSC

IPSC	d
0	7
1	6
Greater than 1	5

33.2 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

Some IO functionality is defined by GPIO register settings independent of this peripheral. For input signals, the GPIO input qualification must be set to asynchronous mode by setting the appropriate GPxQSELn register bits to 11b. The internal pullups can be configured in the GPyPUD register.

See the *General-Purpose Input/Output (GPIO)* chapter for more details on GPIO mux and settings.

Note

The GPIO configuration register GPyODR must be set to normal mode when the I2C is used. The open-drain operation for I2C is managed by the I2C module

33.3 I2C Module Operational Details

This section provides an overview of the I2C bus protocol and how it is implemented.

33.3.1 Input and Output Voltage Levels

One clock pulse is generated by the master device for each data bit transferred. Due to a variety of different technology devices that can be connected to the I2C bus, the levels of logic 0 (low) and logic 1 (high) are not fixed and depend on the associated level of V_{DD}. For details, see the device data manual.

33.3.2 Data Validity

The data on SDA must be stable during the high period of the clock (see Figure 33-5). The high or low state of the data line, SDA, must change only when the clock signal on SCL is low.

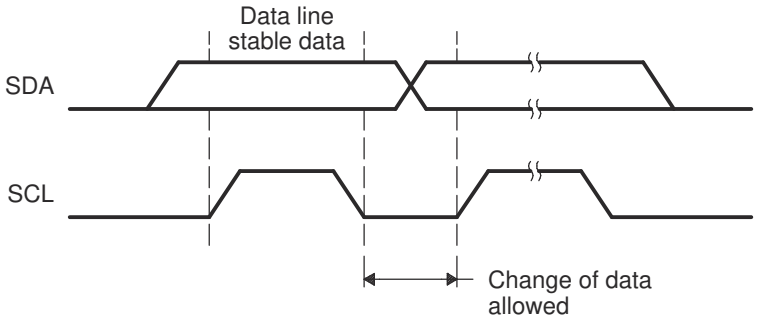


Figure 33-5. Bit Transfer on the I2C bus

33.3.3 Operating Modes

The I2C module has four basic operating modes to support data transfers as a master and as a slave. See Table 33-2 for the names and descriptions of the modes.

If the I2C module is a master, the I2C module begins as a master-transmitter and typically transmits an address for a particular slave. When giving data to the slave, the I2C module must remain a master-transmitter. To receive data from a slave, the I2C module must be changed to the master-receiver mode.

If the I2C module is a slave, the I2C module begins as a slave-receiver and typically sends acknowledgment when the I2C module recognizes the slave address from a master. If the master is sending data to the I2C module, the module must remain a slave-receiver. If the master has requested data from the I2C module, the module must be changed to the slave-transmitter mode.

Table 33-2. Operating Modes of the I2C Module

Operating Mode	Description
Slave-receiver mode	The I2C module is a slave and receives data from a master. All slaves begin in this mode. In this mode, serial data bits received on SDA are shifted in with the clock pulses that are generated by the master. As a slave, the I2C module does not generate the clock signal, but it can hold SCL low while the intervention of the device is required (RSFULL = 1 in I2CSTR) after a byte has been received. See Section 33.3.7 for more details.
Slave-transmitter mode	The I2C module is a slave and transmits data to a master. This mode can be entered only from the slave-receiver mode; the I2C module must first receive a command from the master. When you are using any of the 7-bit/10-bit addressing formats, the I2C module enters its slave-transmitter mode if the slave address byte is the same as its own address (in I2COAR) and the master has transmitted R/ $\overline{W}$ = 1. As a slave-transmitter, the I2C module then shifts the serial data out on SDA with the clock pulses that are generated by the master. While a slave, the I2C module does not generate the clock signal, but it can hold SCL low while the intervention of the device is required (XSMT = 0 in I2CSTR) after a byte has been transmitted. See Section 33.3.7 for more details.
Master-receiver mode	The I2C module is a master and receives data from a slave. This mode can be entered only from the master-transmitter mode; the I2C module must first transmit a command to the slave. When you are using any of the 7-bit/10-bit addressing formats, the I2C module enters its master-receiver mode after transmitting the slave address byte and R/ $\overline{W}$ = 1. Serial data bits on SDA are shifted into the I2C module with the clock pulses generated by the I2C module on SCL. The clock pulses are inhibited and SCL is held low when the intervention of the device is required (RSFULL = 1 in I2CSTR) after a byte has been received.
Master-transmitter mode	The I2C module is a master and transmits control information and data to a slave. All masters begin in this mode. In this mode, data assembled in any of the 7-bit/10-bit addressing formats is shifted out on SDA. The bit shifting is synchronized with the clock pulses generated by the I2C module on SCL. The clock pulses are inhibited and SCL is held low when the intervention of the device is required (XSMT = 0 in I2CSTR) after a byte has been transmitted.

To summarize, SCL is held low in the following conditions:

- When an overrun condition is detected (RSFULL = 1), in Slave-receiver mode.
- When an underflow condition is detected (XSMT = 0), in Slave-transmitter mode.

I2C slave nodes have to accept and provide data when the I2C master node requests it.

- To release SCL in slave-receiver mode, read data from I2CDRR.
- To release SCL in slave-transmitter mode, write data to I2CDXR.
- To force a release without handling the data, reset the module using the I2CMDR.IRS bit.

Table 33-3. Master-Transmitter/Receiver Bus Activity Defined by the RM, STT, and STP Bits of I2CMDR

RM	STT	STP	Bus Activity(1)	Description
0	0	0	None	No activity
0	0	1	P	STOP condition
0	1	0	S-A-D..(n)..D.	START condition, slave address, n data bytes (n = value in I2CCNT)
0	1	1	S-A-D..(n)..D-P	START condition, slave address, n data bytes, STOP condition (n = value in I2CCNT)
1	0	0	None	No activity
1	0	1	P	STOP condition
1	1	0	S-A-D-D-D.	Repeat mode transfer: START condition, slave address, continuous data transfers until STOP condition or next START condition
1	1	1	None	Reserved bit combination (No activity)

(1) S = START condition; A = Address; D = Data byte; P = STOP condition;

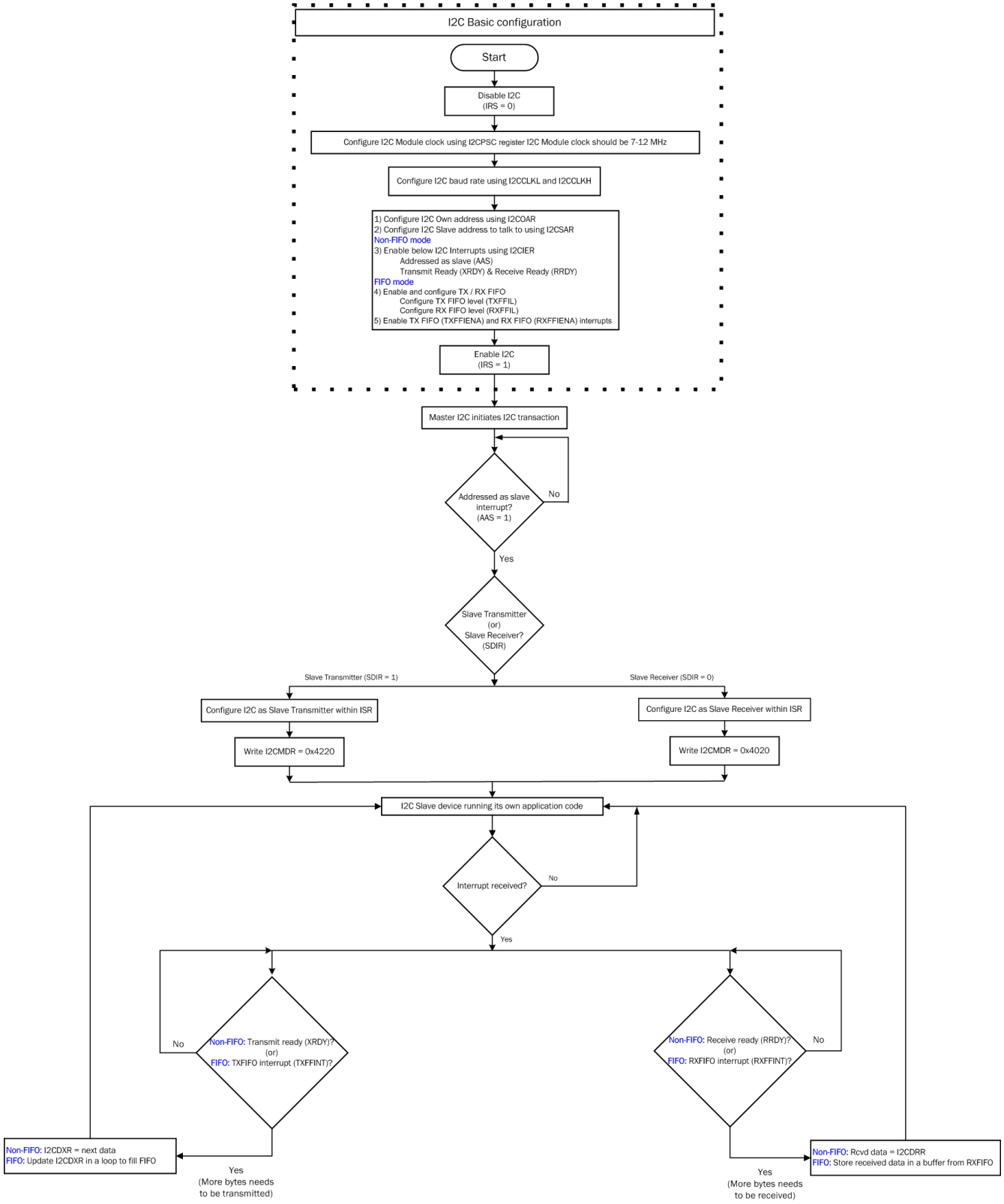


Figure 33-6. I2C Slave TX / RX Flowchart

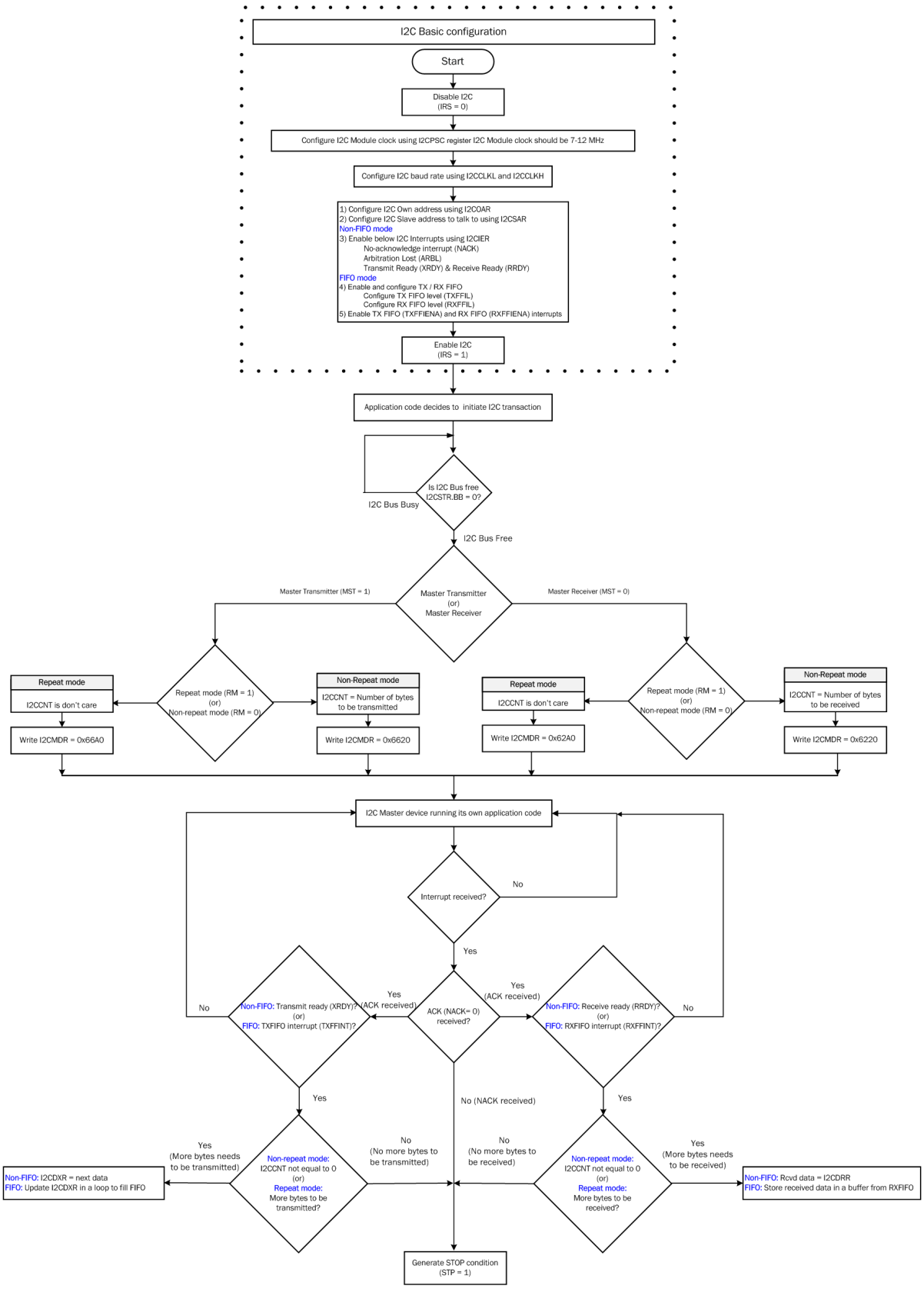


Figure 33-7. I2C Master TX / RX Flowchart

33.3.4 I2C Module START and STOP Conditions

START and STOP conditions can be generated by the I2C module when the module is configured to be a master on the I2C bus. As shown in Figure 33-8:

- The START condition is defined as a high-to-low transition on the SDA line while SCL is high. A master drives this condition to indicate the start of a data transfer.
- The STOP condition is defined as a low-to-high transition on the SDA line while SCL is high. A master drives this condition to indicate the end of a data transfer.

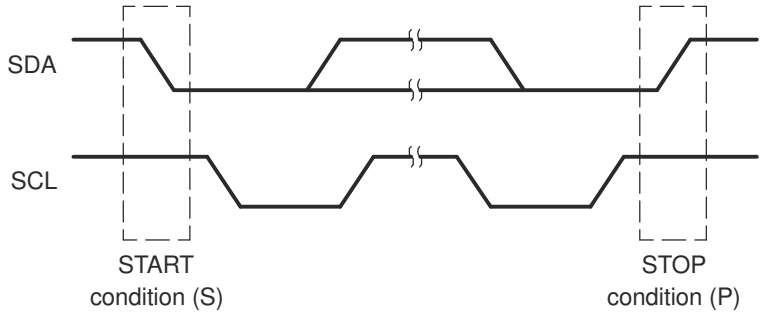


Figure 33-8. I2C Module START and STOP Conditions

After a START condition and before a subsequent STOP condition, the I2C bus is considered busy, and the bus busy (BB) bit of I2CSTR is 1. Between a STOP condition and the next START condition, the bus is considered free, and BB is 0.

For the I2C module to start a data transfer with a START condition, the master mode bit (MST) and the START condition bit (STT) in I2CMDR must both be 1. For the I2C module to end a data transfer with a STOP condition, the STOP condition bit (STP) must be set to 1. When the BB bit is set to 1 and the STT bit is set to 1, a repeated START condition is generated. For a description of I2CMDR and the bits (including MST, STT, and STP), see Section 33.7.

The I2C peripheral cannot detect a START or STOP condition while in reset (IRS = 0). The BB bit remains in the cleared state (BB = 0) while the I2C peripheral is in reset (IRS = 0). When the I2C peripheral is taken out of reset (IRS set to 1), the BB bit does not correctly reflect the I2C bus status until a START or STOP condition is detected.

Follow these steps before initiating the first data transfer with I2C:

1. After taking the I2C peripheral out of reset by setting the IRS bit to 1, wait a period larger than the total time taken for the longest data transfer in the application. By waiting for a period of time after I2C comes out of reset, make sure that at least one START or STOP condition has occurred on the I2C bus and has been captured by the BB bit. After this period, the BB bit correctly reflects the state of the I2C bus.
2. Check the BB bit and verify that BB = 0 (bus not busy) before proceeding.
3. Begin data transfers.

Not resetting the I2C peripheral in between transfers makes sure that the BB bit reflects the actual bus status. If users must reset the I2C peripheral in between transfers, repeat steps 1 through 3 every time the I2C peripheral is taken out of reset.

33.3.5 Non-repeat Mode versus Repeat Mode

Non-repeat mode:

- When I2CMDR.RM = 0, I2C module is configured in non-repeat mode.
- I2CCNT register determines the number of bytes to be transmitted or received.
- If STP = 0 in I2CMDR, the ARDY bit is set when the internal data counter counts down to 0.
- If STP = 1, ARDY bit does not get set and I2C module generates a STOP condition when the internal data counter counts down to 0.

Note

In non-repeat mode (RM = 0), if I2CCNT is set to 0, I2C state machine expects to transmit or receive 65536 bytes and not 0 bytes.

Repeat mode:

- When I2CMDR.RM = 1, I2C module is configured in repeat mode.
- I2CCNT register contents do not determine the number of bytes to be transmitted or received.
- Number of bytes to be transmitted or received can be controlled by software.
- ARDY bit gets set at end of transmission and reception of each byte.

Note

Once you start I2C transaction in non-repeat mode or repeat mode, you cannot switch into another mode until the I2C transaction is completed with a STOP condition.

33.3.6 Serial Data Formats

Figure 33-9 shows an example of a data transfer on the I2C bus. The I2C module supports 1 to 8-bit data values. In Figure 33-9, 8-bit data is transferred. Each bit put on the SDA line equates to 1 pulse on the SCL line, and the values are always transferred with the most significant bit (MSB) first. The number of data values that can be transmitted or received is unrestricted. The serial data format used in Figure 33-9 is the 7-bit addressing format. The I2C module supports the formats shown in Figure 33-10 through Figure 33-12 and described in the paragraphs that follow the figures.

Note

In Figure 33-9 through Figure 33-12, n = the number of data bits (from 1 to 8) specified by the bit count (BC) field of I2CMDR.

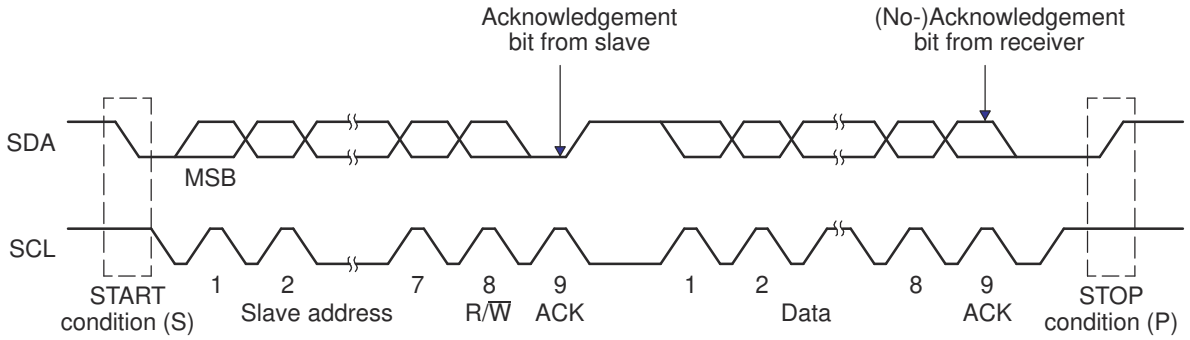


Figure 33-9. I2C Module Data Transfer (7-Bit Addressing with 8-bit Data Configuration Shown)

33.3.6.1 7-Bit Addressing Format

The 7-bit addressing format is the default format after reset. Disabling expanded address (I2CMDR.XA = 0) and free data format (I2CMDR.FDF = 0) enables 7-bit addressing format.

In this format (see Figure 33-10), the first byte after a START condition (S) consists of a 7-bit slave address followed by a R/$\overline{W}$ bit. R/$\overline{W}$ determines the direction of the data:

- R/$\overline{W}$ = 0: The I2C master writes (transmits) data to the addressed slave. This can be achieved by setting I2CMDR.TRX = 1 (Transmitter mode)
- R/$\overline{W}$ = 1: The I2C master reads (receives) data from the slave. This can be achieved by setting I2CMDR.TRX = 0 (Receiver mode)

An extra clock cycle dedicated for acknowledgment (ACK) is inserted after each byte. If the ACK bit is inserted by the slave after the first byte from the master, it is followed by n bits of data from the transmitter (master or slave, depending on the R/$\overline{W}$ bit). n is a number from 1 to 8 determined by the bit count (BC) field of I2CMDR. After the data bits have been transferred, the receiver inserts an ACK bit.

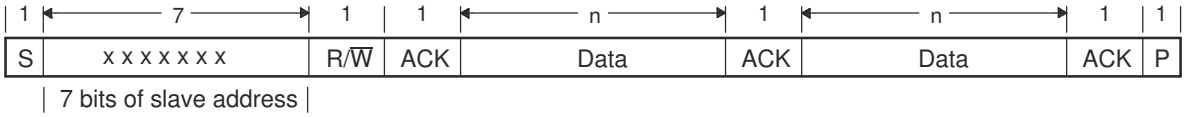


Figure 33-10. I2C Module 7-Bit Addressing Format (FDF = 0, XA = 0 in I2CMDR)

33.3.6.2 10-Bit Addressing Format

The 10-bit addressing format can be enabled by setting expanded address (I2CMDR.XA = 1) and disabling free data format (I2CMDR.FDF = 0).

The 10-bit addressing format (see Figure 33-11) is similar to the 7-bit addressing format, but the master sends the slave address in two separate byte transfers. The first byte consists of 11110b, the two MSBs of the 10-bit slave address, and R/$\overline{W}$. The second byte is the remaining 8 bits of the 10-bit slave address. The slave must send acknowledgment after each of the two byte transfers. Once the master has written the second byte to the slave, the master can either write data or use a repeated START condition to change the data direction. For more details about using 10-bit addressing, see the NXP Semiconductors I2C bus specification.

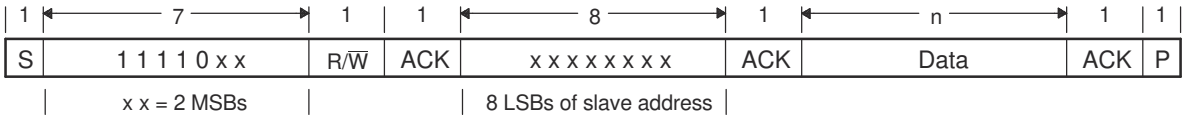


Figure 33-11. I2C Module 10-Bit Addressing Format (FDF = 0, XA = 1 in I2CMDR)

33.3.6.3 Free Data Format

The free data format can be enabled by setting I2CMDR. FDF = 1.

In this format (see Figure 33-12), the first byte after a START condition (S) is a data byte. An ACK bit is inserted after each data byte, which can be from 1 to 8 bits, depending on the BC field of I2CMDR. No address or data-direction bit is sent. Therefore, the transmitter and the receiver must both support the free data format, and the direction of the data must be constant throughout the transfer.

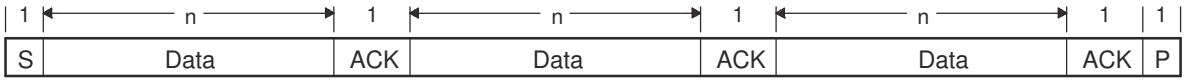


Figure 33-12. I2C Module Free Data Format (FDF = 1 in I2CMDR)

Note

The free data format is not supported in the digital loopback mode (I2CMDR.DLB = 1).

Table 33-4. How the MST and FDF Bits of I2CMDR Affect the Role of the TRX Bit of I2CMDR

MST	FDF	I2C Module State	Function of TRX
0	0	In slave mode but not free data format mode	TRX is a don't care. Depending on the command from the master, the I2C module responds as a receiver or a transmitter.
0	1	In slave mode and free data format mode	The free data format mode requires that the I2C module remains the transmitter or the receiver throughout the transfer. TRX identifies the role of the I2C module: TRX = 1: The I2C module is a transmitter. TRX = 0: The I2C module is a receiver.
1	0	In master mode but not free data format mode	TRX = 1: The I2C module is a transmitter. TRX = 0: The I2C module is a receiver.
1	1	In master mode and free data format mode	TRX = 0: The I2C module is a receiver. TRX = 1: The I2C module is a transmitter.

33.3.6.4 Using a Repeated START Condition

I2C master can communicate with multiple slave addresses without having to give up control of the I2C bus by driving a STOP condition. This can be achieved by driving another START condition at the end of each data type. The repeated START condition can be used with the 7-bit addressing and 10-bit addressing. Figure 33-13 shows a repeated START condition in the 7-bit addressing format.

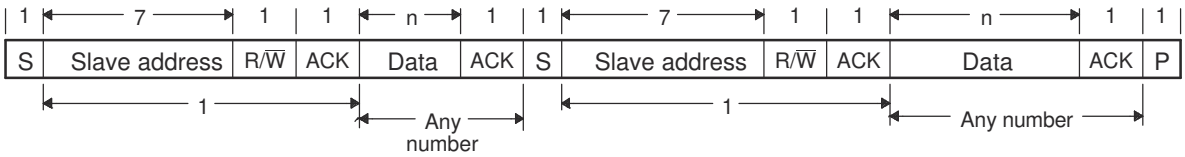


Figure 33-13. Repeated START Condition (in This Case, 7-Bit Addressing Format)

Note

In Figure 33-13, n = the number of data bits (from 1 to 8) specified by the bit count (BC) field of I2CMDR.

33.3.7 Clock Synchronization

Under normal conditions, only one master device generates the clock signal, SCL. During the arbitration procedure, however, there are two or more masters and the clock must be synchronized so that the data output can be compared. Figure 33-14 illustrates the clock synchronization. The wired-AND property of SCL means that a device that first generates a low period on SCL overrules the other devices. At this high-to-low transition, the clock generators of the other devices are forced to start their own low period. The SCL is held low by the device with the longest low period. The other devices that finish their low periods must wait for SCL to be released, before starting their high periods. A synchronized signal on SCL is obtained, where the slowest device determines the length of the low period and the fastest device determines the length of the high period.

If a device pulls down the clock line for a longer time, the result is that all clock generators must enter the wait state. In this way, a slave slows down a fast master and the slow device creates enough time to store a received byte or to prepare a byte to be transmitted.

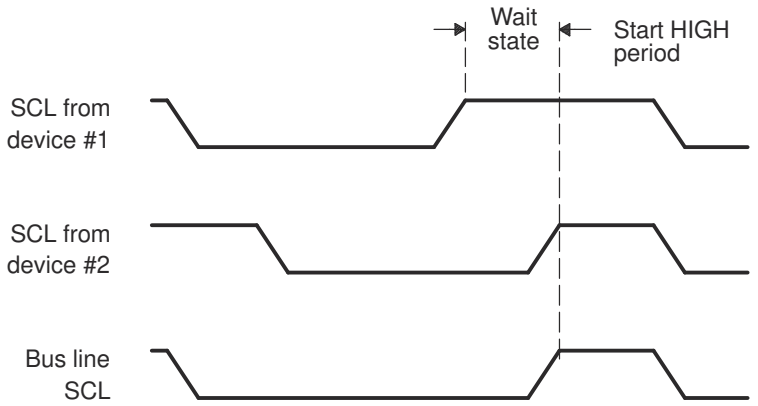


Figure 33-14. Synchronization of Two I2C Clock Generators During Arbitration

33.3.8 Arbitration

If two or more master-transmitters attempt to start a transmission on the same bus at approximately the same time, an arbitration procedure is invoked. The arbitration procedure uses the data presented on the serial data bus (SDA) by the competing transmitters. Figure 33-15 illustrates the arbitration procedure between two devices. The first master-transmitter that releases the SDA line high is overruled by another master-transmitter that drives the SDA low. The arbitration procedure gives priority to the device that transmits the serial data stream with the lowest binary value. If two or more devices send identical first bytes, arbitration continues on the subsequent bytes.

If the I2C module is the losing master, the I2c module switches to the slave-receiver mode, sets the arbitration lost (ARBL) flag, and generates the arbitration-lost interrupt request.

If during a serial transfer the arbitration procedure is still in progress when a repeated START condition or a STOP condition is transmitted to SDA, the master-transmitters involved must send the repeated START condition or the STOP condition at the same position in the format frame. Arbitration is not allowed between:

- A repeated START condition and a data bit
- A STOP condition and a data bit
- A repeated START condition and a STOP condition

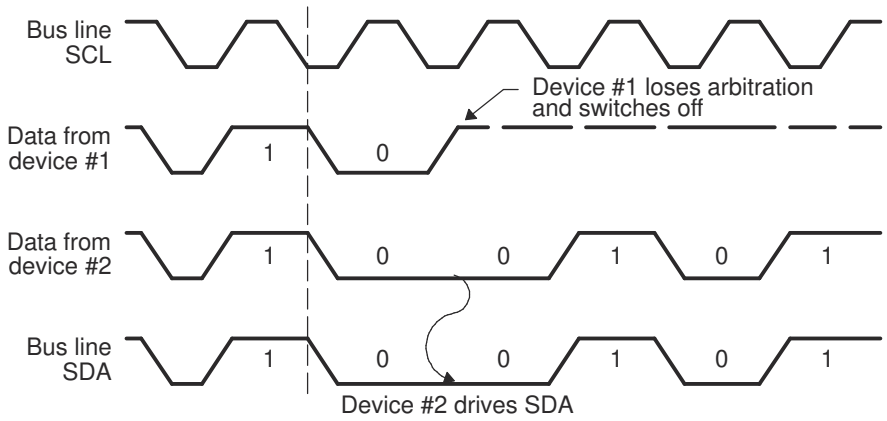


Figure 33-15. Arbitration Procedure Between Two Master-Transmitters

33.3.9 Digital Loopback Mode

The I2C module support a self-test mode called digital loopback, which is enabled by setting the DLB bit in the I2CMDR register. In this mode, data transmitted out of the I2CDXR register is received in the I2CDRR register. The data follows an internal path, and takes n cycles to reach I2CDRR, where:

n = 8 * (SYSCLK) / (I2C module clock (Fmod))

The transmit clock and the receive clock are the same. The address seen on the external SDA pin is the address in the I2COAR register. Figure 33-16 shows the signal routing in digital loopback mode.

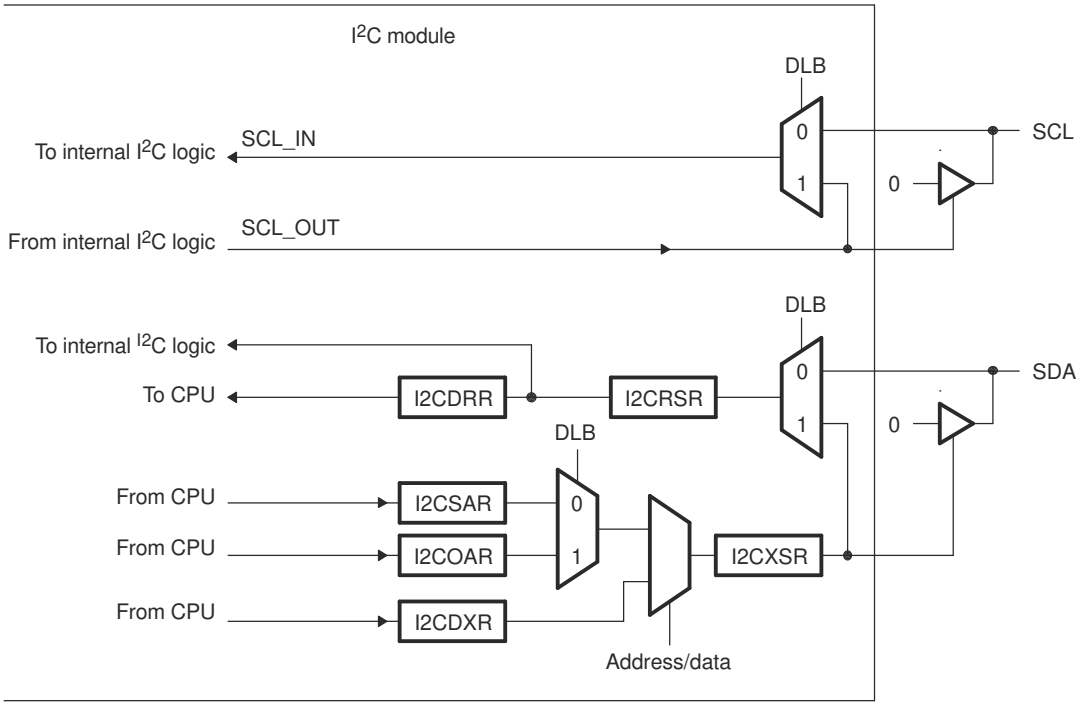


Figure 33-16. Pin Diagram Showing the Effects of the Digital Loopback Mode (DLB) Bit

Note

The free data format (I2CMDR.FDF = 1) is not supported in digital loopback mode.

33.3.10 NACK Bit Generation

When the I2C module is a receiver (master or slave), it can acknowledge or ignore bits sent by the transmitter. To ignore any new bits, the I2C module must send a no-acknowledge (NACK) bit during the acknowledge cycle on the bus. Table 33-5 summarizes the various ways you can allow the I2C module to send a NACK bit.

Note

When a NACK occurs, the following will occur:

1. The STP in I2CMDR is cleared
2. SCL is held low
3. The NACK in I2CSTR is set

Table 33-5. Ways to Generate a NACK Bit

I2C Module Condition	NACK Bit Generation Options
Slave-receiver modes	Allow an overrun condition (RSFULL = 1 in I2CSTR) Reset the module (IRS = 0 in I2CMDR) Set the NACKMOD bit of I2CMDR before the rising edge of the last data bit you intend to receive
Master-receiver mode AND Repeat mode (RM = 1 in I2CMDR)	Generate a STOP condition (STP = 1 in I2CMDR) Reset the module (IRS = 0 in I2CMDR) Set the NACKMOD bit of I2CMDR before the rising edge of the last data bit you intend to receive
Master-receiver mode AND Nonrepeat mode (RM = 0 in I2CMDR)	If STP = 1 in I2CMDR, allow the internal data counter to count down to 0 and thus force a STOP condition If STP = 0, make STP = 1 to generate a STOP condition Reset the module (IRS = 0 in I2CMDR). = 1 to generate a STOP condition Set the NACKMOD bit of I2CMDR before the rising edge of the last data bit you intend to receive

33.4 Interrupt Requests Generated by the I2C Module

Each I2C module can generate two CPU interrupts.

1. Basic I2C interrupt: Possible basic I2C interrupt sources that can trigger this interrupt are described in Section 33.4.1.
2. I2C FIFO interrupt: Possible I2C FIFO interrupt sources that can trigger this interrupt are described in Section 33.4.2

33.4.1 Basic I2C Interrupt Requests

The I2C module generates the interrupt requests described in Table 33-6. As shown in Figure 33-17, all requests are multiplexed through an arbiter to a single I2C interrupt request to the CPU. Each interrupt request has a flag bit in the status register (I2CSTR) and an enable bit in the interrupt enable register (I2CIER). When one of the specified events occurs, the flag bit is set. If the corresponding enable bit is 0, the interrupt request is blocked. If the enable bit is 1, the request is forwarded to the CPU as an I2C interrupt.

The I2C interrupt is one of the maskable interrupts of the CPU. As with any maskable interrupt request, if the request is properly enabled in the CPU, the CPU executes the corresponding interrupt service routine (I2CINT1A_ISR). The I2CINT1A_ISR for the I2C interrupt can determine the interrupt source by reading the interrupt source register, I2CISRC. Then the I2CINT1A_ISR can branch to the appropriate subroutine.

After the CPU reads I2CISRC, the following events occur:

1. The flag for the source interrupt is cleared in I2CSTR. Exception: The ARDY, RRDY, and XRDY bits in I2CSTR are not cleared when I2CISRC is read. To clear one of these bits, write a 1 to the bit.
2. The arbiter determines which of the remaining interrupt requests has the highest priority, writes the code for that interrupt to I2CISRC, and forwards the interrupt request to the CPU.

Table 33-6. Descriptions of the Basic I2C Interrupt Requests

I2C Interrupt Request	Interrupt Source
XRDYINT	Transmit ready condition: The data transmit register (I2CDXR) is ready to accept new data because the previous data has been copied from I2CDXR to the transmit shift register (I2CXSR). As an alternative to using XRDYINT, the CPU can poll the XRDY bit of the status register, I2CSTR. XRDYINT must not be used when in FIFO mode. Use the FIFO interrupts instead.
RRDYINT	Receive ready condition: The data receive register (I2CDRR) is ready to be read because data has been copied from the receive shift register (I2CRSR) to I2CDRR. As an alternative to using RRDYINT, the CPU can poll the RRDY bit of I2CSTR. RRDYINT must not be used when in FIFO mode. Use the FIFO interrupts instead.
ARDYINT	Register-access ready condition: The I2C module registers are ready to be accessed because the previously programmed address, data, and command values have been used. The specific events that generate ARDYINT are the same events that set the ARDY bit of I2CSTR. As an alternative to using ARDYINT, the CPU can poll the ARDY bit.
NACKINT	No-acknowledgment condition: The I2C module is configured as a master-transmitter and did not received acknowledgment from the slave-receiver. As an alternative to using NACKINT, the CPU can poll the NACK bit of I2CSTR.
ARBLINT	Arbitration-lost condition: The I2C module has lost an arbitration contest with another master-transmitter. As an alternative to using ARBLINT, the CPU can poll the ARBL bit of I2CSTR.
SCDINT	Stop condition detected: A STOP condition was detected on the I2C bus. As an alternative to using SCDINT, the CPU can poll the SCD bit of the status register, I2CSTR.
AASINT	Addressed as slave condition: The I2C has been addressed as a slave device by another master on the I2C bus. As an alternative to using AASINT, the CPU can poll the AAS bit of the status register, I2CSTR.

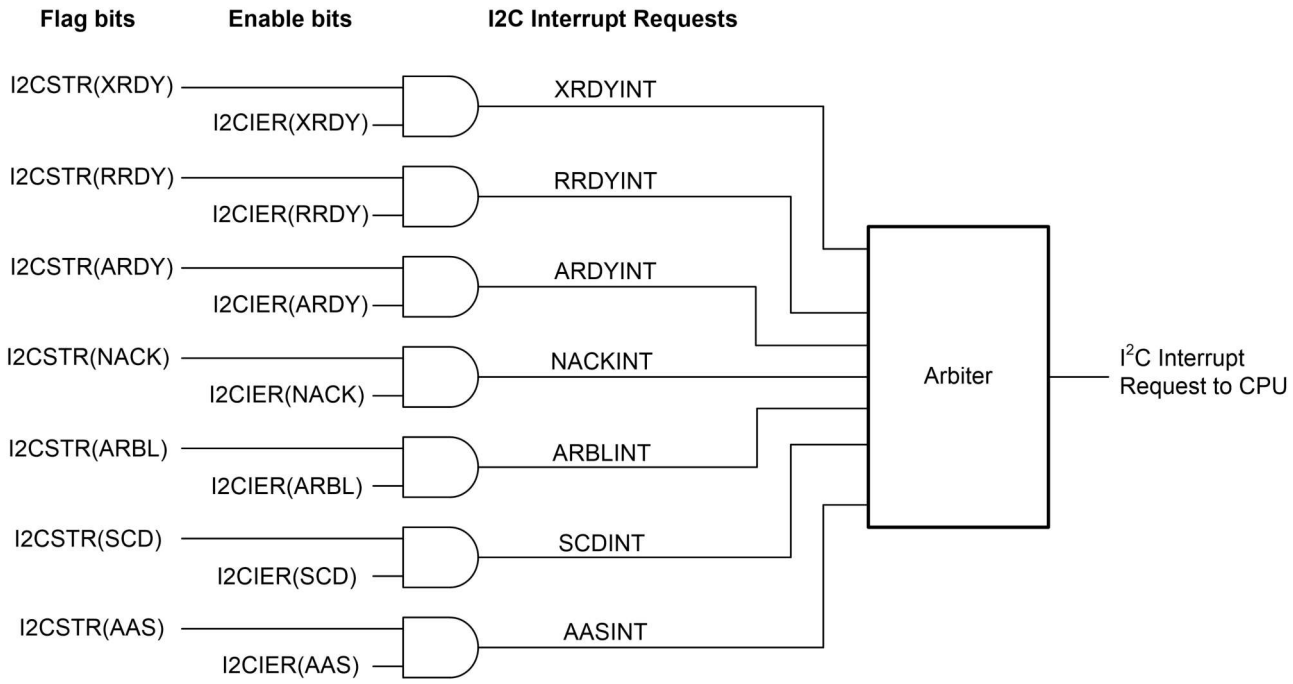


Figure 33-17. Enable Paths of the I2C Interrupt Requests

The priorities of the basic I2C interrupt requests are listed in order of highest priority to lowest priority:

1. ARBLINT
2. NACKINT
3. ARDYINT
4. RRDYINT
5. XRDYINT
6. SCDINT
7. AASINT

The normal transmit interrupt timing makes it possible for stale data to remain in the transmit buffer if a transaction is aborted in the middle of a byte. To avoid this, set the FCM bit in the I2CEMDR register. When this bit is set, the transmit data ready interrupt is generated only when data is required for a bus transaction. In master mode, the interrupt is first generated when the ACK of the address byte is received. In slave mode, the interrupt is first generated when the address is matched. Further interrupts are generated when the data is ACKed. In this mode, XRDY is asserted at the same time as the transmit ready interrupt.

The I2C module has a backwards compatibility bit (BC) in the I2CEMDR register. The timing diagram in Figure 33-18 demonstrates the effect the backwards compatibility bit has on I2C module registers and interrupts when configured as a slave-transmitter.

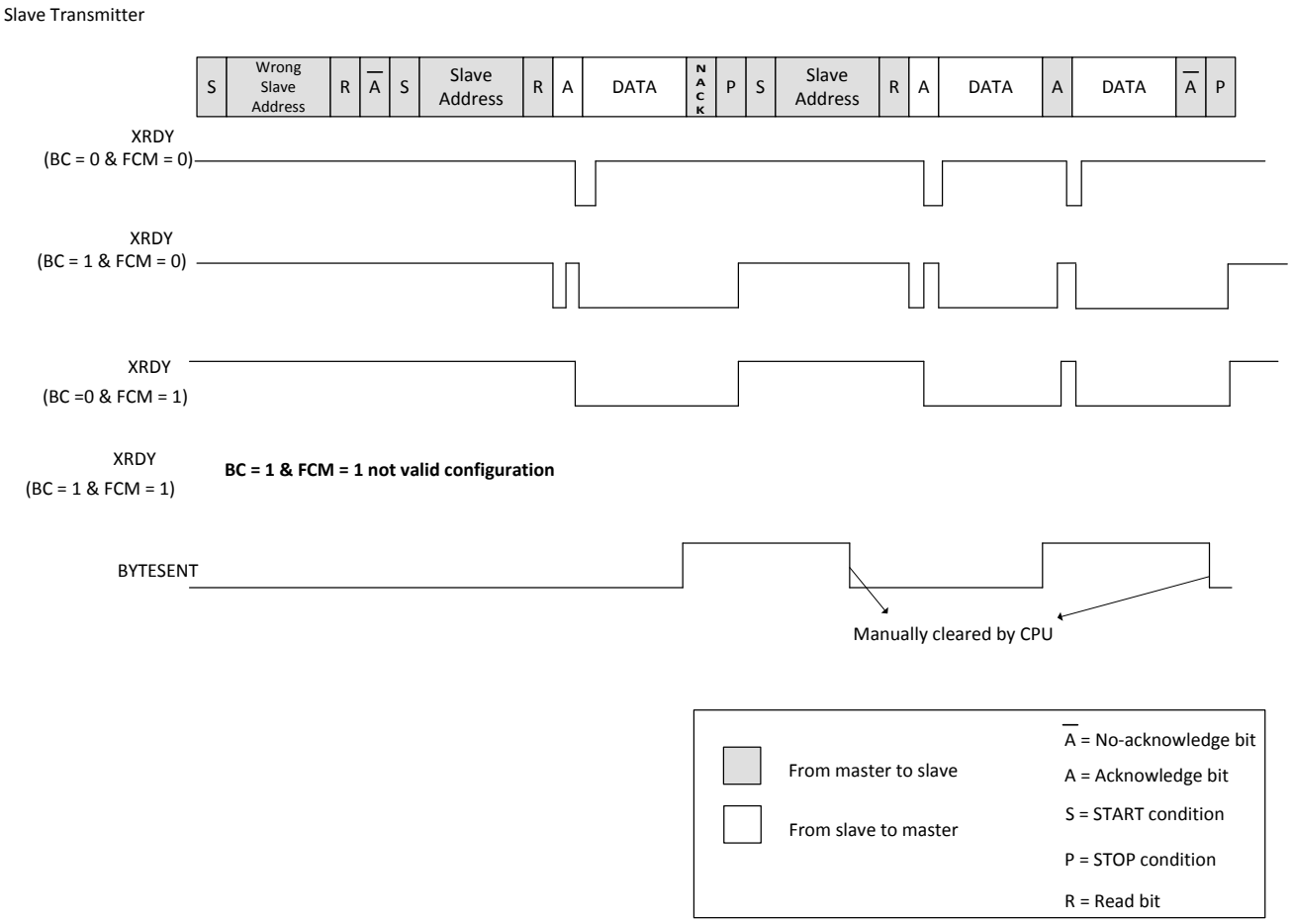


Figure 33-18. Backwards Compatibility Mode and Forward Compatibility Bit, Slave Transmitter

33.4.2 I2C FIFO Interrupts

In addition to the seven basic I2C interrupts, the transmit and receive FIFOs each contain the ability to generate an interrupt (I2CINT2A). The transmit FIFO can be configured to generate an interrupt after transmitting a defined number of bytes, up to 16. The receive FIFO can be configured to generate an interrupt after receiving a defined number of bytes, up to 16. These two interrupt sources are ORed together into a single maskable CPU interrupt. Figure 33-19 shows the structure of I2C FIFO interrupt. The interrupt service routine can then read the FIFO interrupt status flags to determine from which source the interrupt came. See the I2C transmit FIFO register (I2CFFTX) and the I2C receive FIFO register (I2CFFRX) descriptions.

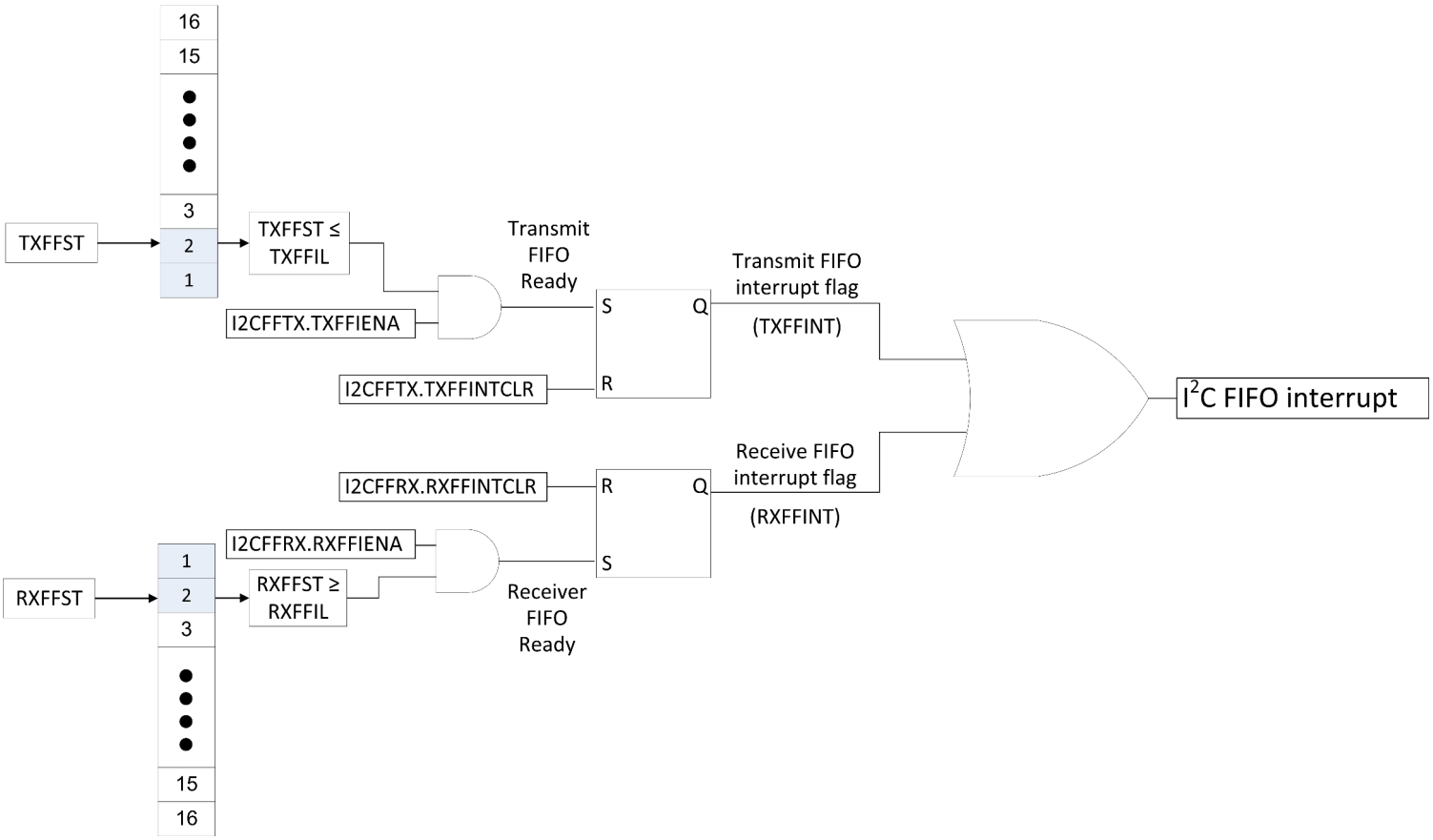


Figure 33-19. I2C FIFO Interrupt

33.5 Resetting or Disabling the I2C Module

You can reset or disable the I2C module in two ways:

- Write 0 to the I2C reset bit (IRS) in the I2C mode register (I2CMDR). All status bits (in I2CSTR) are forced to their default values, and the I2C module remains disabled until IRS is changed to 1. The SDA and SCL pins are in the high-impedance state.
- Initiate a device reset by driving the $\overline{\text{XRS}}$ pin low. The entire device is reset and is held in the reset state until you drive the pin high. When the $\overline{\text{XRS}}$ pin is released, all I2C module registers are reset to their default values. The IRS bit is forced to 0, which resets the I2C module. The I2C module stays in the reset state until you write 1 to IRS.

The IRS must be 0 while you configure or reconfigure the I2C module. Forcing IRS to 0 can be used to save power and to clear error conditions.

33.6 Software

33.6.1 I2C Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/i2c

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

33.6.1.1 C28x-I2C Library source file for FIFO interrupts

FILE: i2cLib_FIFO_controller_interrupt.c

33.6.1.2 C28x-I2C Library source file for FIFO using polling

FILE: i2cLib_FIFO_polling.c

33.6.1.3 C28x-I2C Library source file for FIFO interrupts

FILE: i2cLib_FIFO_controller_target_interrupt.c

33.6.1.4 I2C Loopback with Slave Receive Interrupt - CM

FILE: i2c_ex1_slave_receive_int.c

This program shows how to configure a receive interrupt on the slave module. This includes setting up the I2C0 module for loopback mode as well as configuring the master and slave modules. Loopback mode internally connects the master and slave data and clock lines together. The address of the slave module is set to a value so it can receive data from the master.

This is a 7-bit slave module address sent in the following format: [A6:A5:A4:A3:A2:A1:A0:RS]

A zero in the R/S position of the first byte means that the master transmits (sends) data to the selected slave, and a one in this position means that the master receives data from the slave.

External Connections

- None

Watch Variables

- *ui32DataTx* - Data to send
- *ui32DataRx* - Received data
- *result* - Status of the I2C communication

33.6.1.5 I2C Digital Loopback with FIFO Interrupts

FILE: i2c_ex1_loopback.c

This program uses the internal loopback test mode of the I2C module. Both the TX and RX I2C FIFOs and their interrupts are used. The pinmux and I2C initialization is done through the sysconfig file.

A stream of data is sent and then compared to the received stream. The sent data looks like this:
0000 0001
0001 0002
0002 0003
....
00FE 00FF
00FF 0000
etc..
This pattern is repeated forever.

External Connections

- None

Watch Variables

- *sData* - Data to send

- *rData* - Received data
- *rDataPoint* - Used to keep track of the last position in the receive stream for error checking

33.6.1.6 I2C EEPROM

FILE: i2c_ex2_eeprom.c

This program will write 1-14 words to EEPROM and read them back. The data written and the EEPROM address written to are contained in the message structure, i2cMsgOut. The data read back will be contained in the message structure i2cMsgIn.

External Connections
- Connect external I2C EEPROM at address 0x50
- Connect DEVICE_GPIO_PIN_SDAA on to external EEPROM SDA (serial data) pin
- Connect DEVICE_GPIO_PIN_SCLA on to external EEPROM SCL (serial clock) pin

Watch Variables
- *i2cMsgOut* - Message containing data to write to EEPROM
- *i2cMsgIn* - Message containing data read from EEPROM

33.6.1.7 I2C Digital External Loopback with FIFO Interrupts

FILE: i2c_ex3_external_loopback.c

This program uses the I2CA and I2CB modules for achieving external loopback. The I2CA TX FIFO and the I2CB RX FIFO are used along with their interrupts.

A stream of data is sent on I2CA and then compared to the received stream on I2CB. The sent data looks like this:
0000 0001
0001 0002
0002 0003
....
00FE 00FF
00FF 0000
etc..
This pattern is repeated forever.

External Connections
- Connect SCLA(DEVICE_GPIO_PIN_SCLA) to SCLB (DEVICE_GPIO_PIN_SCLB)
- and SDAA(DEVICE_GPIO_PIN_SDAA) to SDAB (DEVICE_GPIO_PIN_SDAB)
- Connect DEVICE_GPIO_PIN_LED1 to an LED used to depict data transfers.

Watch Variables
- *sData* - Data to send
- *rData* - Received data
- *rDataPoint* - Used to keep track of the last position in the receive stream for error checking

33.6.1.8 I2C EEPROM

FILE: i2c_ex4_eeprom_polling.c

This program will shows how to perform different EEPROM write and read commands using I2C polling method EEPROM used for this example is AT24C256

External Connections

- Connect external I2C EEPROM at address 0x50 Signal | I2CA | EEPROM

SCL | DEVICE_GPIO_PIN_SCLA | SCL SDA | DEVICE_GPIO_PIN_SDAA | SDA

Make sure to connect GND pins if EEPROM and C2000 device are in different board.

33.6.1.9 I2C controller target communication using FIFO interrupts

FILE: i2c_ex5_controller_target_interrupt.c

This program shows how to use I2CA and I2CB modules in both controller and target configuration This example uses I2C FIFO interrupts and doesn't using polling

Example1: I2CA as controller Transmitter and I2CB working target Receiver Example2: I2CA as controller Receiver and I2CB working target Transmitter Example3: I2CB as controller Transmitter and I2CA working target Receiver Example4: I2CB as controller Receiver and I2CA working target Transmitter

\b External \b Connections on launchpad should be made as shown below \n Signal | I2CA | I2CB

SCL | DEVICE_GPIO_PIN_SCLA | DEVICE_GPIO_PIN_SCLB

SDA | DEVICE_GPIO_PIN_SDAA | DEVICE_GPIO_PIN_SDAB

Watch Variables in memory window
- *I2CA_TXdata*
- *I2CA_RXdata*
- *I2CB_TXdata*
- *I2CB_RXdata* stream for error checking

33.6.1.10 I2C EEPROM

FILE: i2c_ex6_eeprom_interrupt.c

This program will shows how to perform different EEPROM write and read commands using I2C interrupts EEPROM used for this example is AT24C256

External Connections

- Connect external I2C EEPROM at address 0x50 Signal | I2CA | EEPROM

SCL | DEVICE_GPIO_PIN_SCLA | SCL SDA | DEVICE_GPIO_PIN_SDAA | SDA

Make sure to connect GND pins if EEPROM and C2000 device are in different board.

//Example 1: EEPROM Byte Write //Example 2: EEPROM Byte Read //Example 3: EEPROM word (16-bit) write //Example 4: EEPROM word (16-bit) read //Example 5: EEPROM Page write //Example 6: EEPROM word Paged read

Watch Variables
- *TX_MsgBuffer* - Message buffer which stores the data to be transmitted
- *RX_MsgBuffer* - Message buffer which stores the data to be received

33.7 I2C Registers

This section describes the C28x I2C Module Registers.

33.7.1 I2C Base Address Table (C28)

Table 33-7. I2C Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
I2caRegs	I2C_REGS	I2CA_BASE	0x0000_7300	YES	YES	-	-	YES
I2cbRegs	I2C_REGS	I2CB_BASE	0x0000_7340	YES	YES	-	-	YES

33.7.2 I2C_REGS Registers

Table 33-8 lists the memory-mapped registers for the I2C_REGS registers. All register offset addresses not listed in Table 33-8 should be considered as reserved locations and the register contents should not be modified.

Table 33-8. I2C_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	I2COAR	I2C Own address		Go
1h	I2CIER	I2C Interrupt Enable		Go
2h	I2CSTR	I2C Status		Go
3h	I2CCLKL	I2C Clock low-time divider		Go
4h	I2CCLKH	I2C Clock high-time divider		Go
5h	I2CCNT	I2C Data count		Go
6h	I2CDRR	I2C Data receive		Go
7h	I2CSAR	I2C Slave address		Go
8h	I2CDXR	I2C Data Transmit		Go
9h	I2CMDR	I2C Mode		Go
Ah	I2CISRC	I2C Interrupt Source		Go
Bh	I2CEMDR	I2C Extended Mode		Go
Ch	I2CPSC	I2C Prescaler		Go
20h	I2CFFTX	I2C FIFO Transmit		Go
21h	I2CFFRX	I2C FIFO Receive		Go

Complex bit access types are encoded to fit into small table cells. Table 33-9 shows the codes that are used for access types in this section.

Table 33-9. I2C_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 33-9. I2C_REGS Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

33.7.2.1 I2COAR Register (Offset = 0h) [Reset = 0h]

I2COAR is shown in Figure 33-20 and described in Table 33-10.

Return to the Summary Table.

The I2C own address register (I2COAR) is a 16-bit register. The I2C module uses this register to specify its own slave address, which distinguishes it from other slaves connected to the I2C-bus. If the 7-bit addressing mode is selected (XA = 0 in I2CMDR), only bits 6-0 are used
write 0s to bits 9-7.

Figure 33-20. I2COAR Register

15	14	13	12	11	10	9	8
RESERVED						OAR	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
OAR							
R/W-0h							

Table 33-10. I2COAR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	OAR	R/W	0h	In 7-bit addressing mode (XA = 0 in I2CMDR): 00h-7Fh Bits 6-0 provide the 7-bit slave address of the I2C module. Write 0s to bits 9-7. In 10-bit addressing mode (XA = 1 in I2CMDR): 000h-3FFh Bits 9-0 provide the 10-bit slave address of the I2C module. Reset type: SYSRSn

33.7.2.2 I2CIER Register (Offset = 1h) [Reset = 0h]

I2CIER is shown in Figure 33-21 and described in Table 33-11.

Return to the Summary Table.

I2CIER is used by the CPU to individually enable or disable I2C interrupt requests.

Figure 33-21. I2CIER Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	AAS	SCD	XRDY	RRDY	ARDY	NACK	ARBL
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 33-11. I2CIER Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6	AAS	R/W	0h	Addressed as slave interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt request disabled 1h (R/W) = Interrupt request enabled
5	SCD	R/W	0h	Stop condition detected interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt request disabled 1h (R/W) = Interrupt request enabled
4	XRDY	R/W	0h	Transmit-data-ready interrupt enable bit. This bit should not be set when using FIFO mode. Reset type: SYSRSn 0h (R/W) = Interrupt request disabled 1h (R/W) = Interrupt request enabled
3	RRDY	R/W	0h	Receive-data-ready interrupt enable bit. This bit should not be set when using FIFO mode. Reset type: SYSRSn 0h (R/W) = Interrupt request disabled 1h (R/W) = Interrupt request enabled
2	ARDY	R/W	0h	Register-access-ready interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt request disabled 1h (R/W) = Interrupt request enabled
1	NACK	R/W	0h	No-acknowledgment interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt request disabled 1h (R/W) = Interrupt request enabled
0	ARBL	R/W	0h	Arbitration-lost interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt request disabled 1h (R/W) = Interrupt request enabled

33.7.2.3 I2CSTR Register (Offset = 2h) [Reset = 410h]

I2CSTR is shown in Figure 33-22 and described in Table 33-12.

Return to the Summary Table.

The I2C status register (I2CSTR) is a 16-bit register used to determine which interrupt has occurred and to read status information.

Figure 33-22. I2CSTR Register

15	14	13	12	11	10	9	8
RESERVED	SDIR	NACKSNT	BB	RSFULL	XSMT	AAS	AD0
R-0h	R/W1C-0h	R/W1C-0h	R-0h	R-0h	R-1h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	BYTESENT	SCD	XRDY	RRDY	ARDY	NACK	ARBL
R-0h	R/W1C-0h	R/W1C-0h	R-1h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

Table 33-12. I2CSTR Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	SDIR	R/W1C	0h	Slave direction bit Reset type: SYSRSn 0h (R/W) = I2C is not addressed as a slave transmitter. SDIR is cleared by one of the following events: - It is manually cleared. To clear this bit, write a 1 to it. - Digital loopback mode is enabled. - A START or STOP condition occurs on the I2C bus. 1h (R/W) = I2C is addressed as a slave transmitter.
13	NACKSNT	R/W1C	0h	NACK sent bit. This bit is used when the I2C module is in the receiver mode. One instance in which NACKSNT is affected is when the NACK mode is used (see the description for NACKMOD in Reset type: SYSRSn 0h (R/W) = NACK not sent. NACKSNT bit is cleared by any one of the following events: - It is manually cleared. To clear this bit, write a 1 to it. - The I2C module is reset (either when 0 is written to the IRS bit of I2CMDR or when the whole device is reset). 1h (R/W) = NACK sent: A no-acknowledge bit was sent during the acknowledge cycle on the I2C-bus.
12	BB	R	0h	Bus busy bit. BB indicates whether the I2C-bus is busy or is free for another data transfer. See the paragraph following the table for more information Reset type: SYSRSn 0h (R/W) = Bus free. BB is cleared by any one of the following events: - The I2C module receives or transmits a STOP bit (bus free). - The I2C module is reset. 1h (R/W) = Bus busy: The I2C module has received or transmitted a START bit on the bus.

Table 33-12. I2CSTR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	RSFULL	R	0h	Receive shift register full bit. RSFULL indicates an overrun condition during reception. Overrun occurs when new data is received into the shift register (I2CRSR) and the old data has not been read from the receive register (I2CDRR). As new bits arrive from the SDA pin, they overwrite the bits in I2CRSR. The new data will not be copied to ICDRR until the previous data is read. Reset type: SYSRSn 0h (R/W) = No overrun detected. RSFULL is cleared by any one of the following events: - I2CDRR is read by the CPU. Emulator reads of the I2CDRR do not affect this bit. - The I2C module is reset. 1h (R/W) = Overrun detected
10	XSMT	R	1h	Transmit shift register empty bit. XSMT = 0 indicates that the transmitter has experienced underflow. Underflow occurs when the transmit shift register (I2CXSR) is empty but the data transmit register (I2CDXR) has not been loaded since the last I2CDXR-to-I2CXSR transfer. The next I2CDXR-to-I2CXSR transfer will not occur until new data is in I2CDXR. If new data is not transferred in time, the previous data may be re-transmitted on the SDA pin. Reset type: SYSRSn 0h (R/W) = Underflow detected (empty) 1h (R/W) = No underflow detected (not empty). XSMT is set by one of the following events: - Data is written to I2CDXR. - The I2C module is reset
9	AAS	R	0h	Addressed-as-slave bit Reset type: SYSRSn 0h (R/W) = In the 7-bit addressing mode, the AAS bit is cleared when receiving a NACK, a STOP condition, or a repeated START condition. In the 10-bit addressing mode, the AAS bit is cleared when receiving a NACK, a STOP condition, or by a slave address different from the I2C peripheral's own slave address. 1h (R/W) = The I2C module has recognized its own slave address or an address of all zeros (general call).
8	AD0	R	0h	Address 0 bits Reset type: SYSRSn 0h (R/W) = AD0 has been cleared by a START or STOP condition. 1h (R/W) = An address of all zeros (general call) is detected.
7	RESERVED	R	0h	Reserved
6	BYTESENT	R/W1C	0h	Byte Transmit over indication. BYTESENT is set when the master/slave has successfully sent the byte on SCL/SDA lines. This is diagnostic register which needs to be explicitly cleared by Software. In case not cleared the stale status would keep reflecting as no automated clear incorporated to avoid corner conditions. Reset type: SYSRSn 0h (R/W) = The I2C module has not finished transmitting the next data byte. BYTESENT is cleared by any one of the following events: - It is manually cleared. To clear this bit, write a 1 to it. - The I2C module is reset. 1h (R/W) = The I2C module has completed the transmission of a byte.

Table 33-12. I2CSTR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	SCD	R/W1C	0h	Stop condition detected bit. SCD is set when the I2C sends or receives a STOP condition. The I2C module delays clearing of the I2CMDR[STP] bit until the SCD bit is set. Reset type: SYSRSn 0h (R/W) = STOP condition not detected since SCD was last cleared. SCD is cleared by any one of the following events: - I2CISRC is read by the CPU when it contains the value 110b (stop condition detected). Emulator reads of the I2CISRC do not affect this bit. - SCD is manually cleared. To clear this bit, write a 1 to it. - The I2C module is reset. 1h (R/W) = A STOP condition has been detected on the I2C bus.
4	XRDY	R	1h	Transmit-data-ready interrupt flag bit. When not in FIFO mode, XRDY indicates that the data transmit register (I2CDXR) is ready to accept new data. FCM=0 : When the previous data has been copied from I2CDXR to the transmit shift register (I2CXSR). The CPU can poll XRDY or use the XRDY interrupt request When in FIFO mode, use TXFFINT instead. FCM=1: XRDY is asserted only when next data is required it gets deasserted with write to I2CDXR. Both Polling and interrupt based data transfers are allowed in the FCM mode. Reset type: SYSRSn 0h (R/W) = I2CDXR not ready. XRDY is cleared when data is written to I2CDXR. 1h (R/W) = I2CDXR ready: Data has been copied from I2CDXR to I2CXSR. XRDY is also forced to 1 when the I2C module is reset.
3	RRDY	R/W1C	0h	Receive-data-ready interrupt flag bit. When not in FIFO mode, RRDY indicates that the data receive register (I2CDRR) is ready to be read because data has been copied from the receive shift register (I2CRSR) to I2CDRR. The CPU can poll RRDY or use the RRDY interrupt request When in FIFO mode, use RXFFINT instead. Reset type: SYSRSn 0h (R/W) = I2CDRR not ready. RRDY is cleared by any one of the following events: - I2CDRR is read by the CPU. Emulator reads of the I2CDRR do not affect this bit. - RRDY is manually cleared. To clear this bit, write a 1 to it. - The I2C module is reset. 1h (R/W) = I2CDRR ready: Data has been copied from I2CRSR to I2CDRR.
2	ARDY	R/W1C	0h	Register-access-ready interrupt flag bit (only Applicable when the I2C module is in the master mode). ARDY indicates that the I2C module registers are ready to be accessed because the previously programmed address, data, and command values have been used. The CPU can poll ARDY or use the ARDY interrupt request Reset type: SYSRSn 0h (R/W) = The registers are not ready to be accessed. ARDY is cleared by any one of the following events: - The I2C module starts using the current register contents. - ARDY is manually cleared. To clear this bit, write a 1 to it. - The I2C module is reset. 1h (R/W) = The registers are ready to be accessed. In the nonrepeat mode (RM = 0 in I2CMDR): If STP = 0 in I2CMDR, the ARDY bit is set when the internal data counter counts down to 0. If STP = 1, ARDY is not affected (instead, the I2C module generates a STOP condition when the counter reaches 0). In the repeat mode (RM = 1): ARDY is set at the end of each byte transmitted from I2CDXR.

Table 33-12. I2CSTR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	NACK	R/W1C	0h	No-acknowledgment interrupt flag bit. NACK applies when the I2C module is a master transmitter. NACK indicates whether the I2C module has detected an acknowledge bit (ACK) or a noacknowledge bit (NACK) from the slave receiver. The CPU can poll NACK or use the NACK interrupt request. Reset type: SYSRSn 0h (R/W) = ACK received/NACK not received. This bit is cleared by any one of the following events: - An acknowledge bit (ACK) has been sent by the slave receiver. - NACK is manually cleared. To clear this bit, write a 1 to it. - The CPU reads the interrupt source register (I2CISRC) and the register contains the code for a NACK interrupt. Emulator reads of the I2CISRC do not affect this bit. - The I2C module is reset. 1h (R/W) = NACK bit received. The hardware detects that a no-acknowledge (NACK) bit has been received. Note: While the I2C module performs a general call transfer, NACK is 1, even if one or more slaves send acknowledgment.
0	ARBL	R/W1C	0h	Arbitration-lost interrupt flag bit (only applicable when the I2C module is a master-transmitter). ARBL primarily indicates when the I2C module has lost an arbitration contest with another master/transmitter. The CPU can poll ARBL or use the ARBL interrupt request. Reset type: SYSRSn 0h (R/W) = Arbitration not lost. AL is cleared by any one of the following events: - AL is manually cleared. To clear this bit, write a 1 to it. - The CPU reads the interrupt source register (I2CISRC) and the register contains the code for an AL interrupt. Emulator reads of the I2CISRC do not affect this bit. - The I2C module is reset. 1h (R/W) = Arbitration lost. AL is set by any one of the following events: - The I2C module senses that it has lost an arbitration with two or more competing transmitters that started a transmission almost simultaneously. - The I2C module attempts to start a transfer while the BB (bus busy) bit is set to 1. When AL becomes 1, the MST and STP bits of I2CMDR are cleared, and the I2C module becomes a slave-receiver.

33.7.2.4 I2CCLKL Register (Offset = 3h) [Reset = 0h]

I2CCLKL is shown in Figure 33-23 and described in Table 33-13.

Return to the Summary Table.

I2C Clock low-time divider

Figure 33-23. I2CCLKL Register

15	14	13	12	11	10	9	8
I2CCLKL							
R/W-0h							
7	6	5	4	3	2	1	0
I2CCLKL							
R/W-0h							

Table 33-13. I2CCLKL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	I2CCLKL	R/W	0h	Clock low-time divide-down value. To produce the low time duration of the master clock, the period of the module clock is multiplied by (ICCL + d). d is an adjustment factor based on the prescaler. See the Clock Divider Registers section of the Introduction for details. Note: These bits must be set to a non-zero value for proper I2C clock generation. Reset type: SYSRSn

33.7.2.5 I2CCLKH Register (Offset = 4h) [Reset = 0h]

I2CCLKH is shown in Figure 33-24 and described in Table 33-14.

Return to the Summary Table.

I2C Clock high-time divider

Figure 33-24. I2CCLKH Register

15	14	13	12	11	10	9	8
I2CCLKH							
R/W-0h							
7	6	5	4	3	2	1	0
I2CCLKH							
R/W-0h							

Table 33-14. I2CCLKH Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	I2CCLKH	R/W	0h	Clock high-time divide-down value. To produce the high time duration of the master clock, the period of the module clock is multiplied by (ICCL + d). d is an adjustment factor based on the prescaler. See the Clock Divider Registers section of the Introduction for details. Note: These bits must be set to a non-zero value for proper I2C clock generation. Reset type: SYSRSn

33.7.2.6 I2CCNT Register (Offset = 5h) [Reset = 0h]

I2CCNT is shown in Figure 33-25 and described in Table 33-15.

Return to the Summary Table.

I2CCNT is a 16-bit register used to indicate how many data bytes to transfer when the I2C module is configured as a transmitter, or to receive when configured as a master receiver. In the repeat mode (RM = 1), I2CCNT is not used.
The value written to I2CCNT is copied to an internal data counter. The internal data counter is decremented by 1 for each byte transferred (I2CCNT remains unchanged). If a STOP condition is requested in the master mode (STP = 1 in I2CMDR), the I2C module terminates the transfer with a STOP condition when the countdown is complete (that is, when the last byte has been transferred).

Figure 33-25. I2CCNT Register

15	14	13	12	11	10	9	8
I2CCNT							
R/W-0h							
7	6	5	4	3	2	1	0
I2CCNT							
R/W-0h							

Table 33-15. I2CCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	I2CCNT	R/W	0h	Data count value. I2CCNT indicates the number of data bytes to transfer or receive. If a STOP condition is specified (STP=1) then I2CCNT will decrease after each byte is sent until it reaches zero, which in turn will generate a STOP condition. The value in I2CCNT is a don't care when the RM bit in I2CMDR is set to 1. Reset type: SYSRSn 0h (R/W) = data count value is 65536 1h (R/W) = data count value is 1 2h (R/W) = data count value is 2 FFFFh (R/W) = data count value is 65535

33.7.2.7 I2CDRR Register (Offset = 6h) [Reset = 0h]

I2CDRR is shown in Figure 33-26 and described in Table 33-16.

Return to the Summary Table.

I2CDRR is a 16-bit register used by the CPU to read received data. The I2C module can receive a data byte with 1 to 8 bits. The number of bits is selected with the bit count (BC) bits in I2CMDR. One bit at a time is shifted in from the SDA pin to the receive shift register (I2CRSR). When a complete data byte has been received, the I2C module copies the data byte from I2CRSR to I2CDRR. The CPU cannot access I2CRSR directly.
If a data byte with fewer than 8 bits is in I2CDRR, the data value is right-justified, and the other bits of I2CDRR(7-0) are undefined. For example, if BC = 011 (3-bit data size), the receive data is in I2CDRR(2-0), and the content of I2CDRR(7-3) is undefined.
When in the receive FIFO mode, the I2CDRR register acts as the receive FIFO buffer.

Figure 33-26. I2CDRR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
DATA							
R-0h							

Table 33-16. I2CDRR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	DATA	R	0h	Receive data Reset type: SYSRSn

33.7.2.8 I2CSAR Register (Offset = 7h) [Reset = 3FFh]

I2CSAR is shown in Figure 33-27 and described in Table 33-17.

Return to the Summary Table.

The I2C slave address register (I2CSAR) is a 16-bit register for storing the next slave address that will be transmitted by the I2C module when it is a master. The SAR field of I2CSAR contains a 7-bit or 10-bit slave address. When the I2C module is not using the free data format (FDF = 0 in I2CMDR), it uses this address to initiate data transfers with a slave, or slaves. When the address is nonzero, the address is for a particular slave. When the address is 0, the address is a general call to all slaves. If the 7-bit addressing mode is selected (XA = 0 in I2CMDR), only bits 6-0 of I2CSAR are used
write 0s to bits 9-7.

Figure 33-27. I2CSAR Register

15	14	13	12	11	10	9	8
RESERVED						SAR	
R-0h						R/W-3FFh	
7	6	5	4	3	2	1	0
SAR							
R/W-3FFh							

Table 33-17. I2CSAR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	SAR	R/W	3FFh	In 7-bit addressing mode (XA = 0 in I2CMDR): 00h-7Fh Bits 6-0 provide the 7-bit slave address that the I2C module transmits when it is in the master-transmitter mode. Write 0s to bits 9-7. In 10-bit addressing mode (XA = 1 in I2CMDR): 000h-3FFh Bits 9-0 provide the 10-bit slave address that the I2C module transmits when it is in the master transmitter mode. Reset type: SYSRSn

33.7.2.9 I2CDXR Register (Offset = 8h) [Reset = 0h]

I2CDXR is shown in Figure 33-28 and described in Table 33-18.

Return to the Summary Table.

The CPU writes transmit data to I2CDXR. This 16-bit register accepts a data byte with 1 to 8 bits. Before writing to I2CDXR, specify how many bits are in a data byte by loading the appropriate value into the bit count (BC) bits of I2CMDR. When writing a data byte with fewer than 8 bits, make sure the value is right-aligned in I2CDXR. After a data byte is written to I2CDXR, the I2C module copies the data byte to the transmit shift register (I2CXSR). The CPU cannot access I2CXSR directly. From I2CXSR, the I2C module shifts the data byte out on the SDA pin, one bit at a time.
When in the transmit FIFO mode, the I2CDXR register acts as the transmit FIFO buffer.

Figure 33-28. I2CDXR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
DATA							
R/W-0h							

Table 33-18. I2CDXR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	DATA	R/W	0h	Transmit data Reset type: SYSRSn

33.7.2.10 I2CMDR Register (Offset = 9h) [Reset = 0h]

I2CMDR is shown in Figure 33-29 and described in Table 33-19.

Return to the Summary Table.

The I2C mode register (I2CMDR) is a 16-bit register that contains the control bits of the I2C module.

Figure 33-29. I2CMDR Register

15	14	13	12	11	10	9	8
NACKMOD	FREE	STT	RESERVED	STP	MST	TRX	XA
R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RM	DLB	IRS	STB	FDF	BC		
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h		

Table 33-19. I2CMDR Register Field Descriptions

Bit	Field	Type	Reset	Description
15	NACKMOD	R/W	0h	NACK mode bit. This bit is only applicable when the I2C module is acting as a receiver. Reset type: SYSRSn 0h (R/W) = In the slave-receiver mode: The I2C module sends an acknowledge (ACK) bit to the transmitter during each acknowledge cycle on the bus. The I2C module only sends a no-acknowledge (NACK) bit if you set the NACKMOD bit. In the master-receiver mode: The I2C module sends an ACK bit during each acknowledge cycle until the internal data counter counts down to 0. At that point, the I2C module sends a NACK bit to the transmitter. To have a NACK bit sent earlier, you must set the NACKMOD bit 1h (R/W) = In either slave-receiver or master-receiver mode: The I2C module sends a NACK bit to the transmitter during the next acknowledge cycle on the bus. Once the NACK bit has been sent, NACKMOD is cleared. Important: To send a NACK bit in the next acknowledge cycle, you must set NACKMOD before the rising edge of the last data bit.
14	FREE	R/W	0h	This bit controls the action taken by the I2C module when a debugger breakpoint is encountered. Reset type: SYSRSn 0h (R/W) = When I2C module is master: If SCL is low when the breakpoint occurs, the I2C module stops immediately and keeps driving SCL low, whether the I2C module is the transmitter or the receiver. If SCL is high, the I2C module waits until SCL becomes low and then stops. When I2C module is slave: A breakpoint forces the I2C module to stop when the current transmission/reception is complete. 1h (R/W) = The I2C module runs free that is, it continues to operate when a breakpoint occurs.
13	STT	R/W	0h	START condition bit (only applicable when the I2C module is a master). The RM, STT, and STP bits determine when the I2C module starts and stops data transmissions (see Table 9-6). Note that the STT and STP bits can be used to terminate the repeat mode, and that this bit is not writable when IRS = 0. Reset type: SYSRSn 0h (R/W) = In the master mode, STT is automatically cleared after the START condition has been generated. 1h (R/W) = In the master mode, setting STT to 1 causes the I2C module to generate a START condition on the I2C-bus
12	RESERVED	R	0h	Reserved

Table 33-19. I2CMDR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	STP	R/W	0h	STOP condition bit (only applicable when the I2C module is a master). In the master mode, the RM,STT, and STP bits determine when the I2C module starts and stops data transmissions. Note that the STT and STP bits can be used to terminate the repeat mode, and that this bit is not writable when IRS=0. When in non-repeat mode, at least one byte must be transferred before a stop condition can be generated. The I2C module delays clearing of this bit until after the I2CSTR[SCD] bit is set. To avoid disrupting the I2C state machine, the user must wait until this bit is clear before initiating a new message. Reset type: SYSRSn 0h (R/W) = STP is automatically cleared after the STOP condition has been generated 1h (R/W) = STP has been set by the device to generate a STOP condition when the internal data counter of the I2C module counts down to 0.
10	MST	R/W	0h	Master mode bit. MST determines whether the I2C module is in the slave mode or the master mode. MST is automatically changed from 1 to 0 when the I2C master generates a STOP condition Reset type: SYSRSn 0h (R/W) = Slave mode. The I2C module is a slave and receives the serial clock from the master. 1h (R/W) = Master mode. The I2C module is a master and generates the serial clock on the SCL pin.
9	TRX	R/W	0h	Transmitter mode bit. When relevant, TRX selects whether the I2C module is in the transmitter mode or the receiver mode. Reset type: SYSRSn 0h (R/W) = Receiver mode. The I2C module is a receiver and receives data on the SDA pin. 1h (R/W) = Transmitter mode. The I2C module is a transmitter and transmits data on the SDA pin.
8	XA	R/W	0h	Expanded address enable bit. Reset type: SYSRSn 0h (R/W) = 7-bit addressing mode (normal address mode). The I2C module transmits 7-bit slave addresses (from bits 6-0 of I2CSAR), and its own slave address has 7 bits (bits 6-0 of I2COAR). 1h (R/W) = 10-bit addressing mode (expanded address mode). The I2C module transmits 10-bit slave addresses (from bits 9-0 of I2CSAR), and its own slave address has 10 bits (bits 9-0 of I2COAR).
7	RM	R/W	0h	Repeat mode bit (only applicable when the I2C module is a master-transmitter). The RM, STT, and STP bits determine when the I2C module starts and stops data transmissions Reset type: SYSRSn 0h (R/W) = Nonrepeat mode. The value in the data count register (I2CCNT) determines how many bytes are received/transmitted by the I2C module. 1h (R/W) = Repeat mode. A data byte is transmitted each time the I2CDXR register is written to (or until the transmit FIFO is empty when in FIFO mode) until the STP bit is manually set. The value of I2CCNT is ignored. The ARDY bit/interrupt can be used to determine when the I2CDXR (or FIFO) is ready for more data, or when the data has all been sent and the CPU is allowed to write to the STP bit.

Table 33-19. I2CMDR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	DLB	R/W	0h	Digital loopback mode bit. Reset type: SYSRSn 0h (R/W) = Digital loopback mode is disabled. 1h (R/W) = Digital loopback mode is enabled. For proper operation in this mode, the MST bit must be 1. In the digital loopback mode, data transmitted out of I2CDXR is received in I2CDRR after n device cycles by an internal path, where: n = ((I2C input clock frequency/module clock frequency) x 8) The transmit clock is also the receive clock. The address transmitted on the SDA pin is the address in I2COAR. Note: The free data format (FDF = 1) is not supported in the digital loopback mode.
5	IRS	R/W	0h	I2C module reset bit. Reset type: SYSRSn 0h (R/W) = The I2C module is in reset/disabled. When this bit is cleared to 0, all status bits (in I2CSTR) are set to their default values. 1h (R/W) = The I2C module is enabled. This has the effect of releasing the I2C bus if the I2C peripheral is holding it.
4	STB	R/W	0h	START byte mode bit. This bit is only applicable when the I2C module is a master. As described in version 2.1 of the Philips Semiconductors I2C-bus specification, the START byte can be used to help a slave that needs extra time to detect a START condition. When the I2C module is a slave, it ignores a START byte from a master, regardless of the value of the STB bit. Reset type: SYSRSn 0h (R/W) = The I2C module is not in the START byte mode. 1h (R/W) = The I2C module is in the START byte mode. When you set the START condition bit (STT), the I2C module begins the transfer with more than just a START condition. Specifically, it generates: 1. A START condition 2. A START byte (0000 0001b) 3. A dummy acknowledge clock pulse 4. A repeated START condition Then, as normal, the I2C module sends the slave address that is in I2CSAR.
3	FDF	R/W	0h	Free data format mode bit. Reset type: SYSRSn 0h (R/W) = Free data format mode is disabled. Transfers use the 7-/10-bit addressing format selected by the XA bit. 1h (R/W) = Free data format mode is enabled. Transfers have the free data (no address) format described in Section 9.2.5. The free data format is not supported in the digital loopback mode (DLB=1).
2-0	BC	R/W	0h	Bit count bits. BC defines the number of bits (1 to 8) in the next data byte that is to be received or transmitted by the I2C module. The number of bits selected with BC must match the data size of the other device. Notice that when BC = 000b, a data byte has 8 bits. BC does not affect address bytes, which always have 8 bits. Note: If the bit count is less than 8, receive data is right-justified in I2CDRR(7-0), and the other bits of I2CDRR(7-0) are undefined. Also, transmit data written to I2CDXR must be right-justified Reset type: SYSRSn 0h (R/W) = 8 bits per data byte 1h (R/W) = 1 bit per data byte 2h (R/W) = 2 bits per data byte 3h (R/W) = 3 bits per data byte 4h (R/W) = 4 bits per data byte 5h (R/W) = 5 bits per data byte 6h (R/W) = 6 bits per data byte 7h (R/W) = 7 bits per data byte

33.7.2.11 I2CISRC Register (Offset = Ah) [Reset = 0h]

I2CISRC is shown in Figure 33-30 and described in Table 33-20.

Return to the Summary Table.

The I2C interrupt source register (I2CISRC) is a 16-bit register used by the CPU to determine which event generated the I2C interrupt.

Figure 33-30. I2CISRC Register

15	14	13	12	11	10	9	8
RESERVED				WRITE_ZEROS			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
RESERVED						INTCODE	
R-0h						R-0h	

Table 33-20. I2CISRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-8	WRITE_ZEROS	R/W	0h	TI internal testing bits These reserved bit locations should always be written as zeros. Reset type: SYSRSn
7-3	RESERVED	R	0h	Reserved
2-0	INTCODE	R	0h	Interrupt code bits. The binary code in INTCODE indicates the event that generated an I2C interrupt. A CPU read will clear this field. If another lower priority interrupt is pending and enabled, the value corresponding to that interrupt will then be loaded. Otherwise, the value will stay cleared. The interrupt events below are listed in descending order of priority. That is INTCODE 1 (Arbitration lost) has the highest priority and INTCODE 7 (Addressed as slave) has the lowest priority. In the case of an arbitration lost, a no-acknowledgment condition detected, or a stop condition detected, a CPU read will also clear the associated interrupt flag bit in the I2CSTR register. Emulator reads will not affect the state of this field or of the status bits in the I2CSTR register. Reset type: SYSRSn 0h (R/W) = None 1h (R/W) = Arbitration lost 2h (R/W) = No-acknowledgment condition detected 3h (R/W) = Registers ready to be accessed 4h (R/W) = Receive data ready 5h (R/W) = Transmit data ready 6h (R/W) = Stop condition detected 7h (R/W) = Addressed as slave

33.7.2.12 I2CEMDR Register (Offset = Bh) [Reset = 1h]

I2CEMDR is shown in Figure 33-31 and described in Table 33-21.

Return to the Summary Table.

I2C Extended Mode

Figure 33-31. I2CEMDR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						FCM	BC
R-0h						R/W-0h	R/W-1h

Table 33-21. I2CEMDR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R	0h	Reserved
1	FCM	R/W	0h	Forward Compatibility mode. This bit when programmed brings the functionality of Tx request only when Tx data required regardless of data status in Tx buffer for non-FIFO mode. This register affects the XRDY behavior hence needs to be set after releasing the IRS (I2CMDR[5]). Reset type: SYSRSn 0h (R/W) = Legacy functionality of requesting Tx data upon buffer copy to shift register or upon start condition is active. Stale data is reused after illegal start, ARB Lost, NACK conditions. 1h (R/W) = New functionality of requesting data only upon ACK (address/data) is active.
0	BC	R/W	1h	Backwards compatibility mode. This bit affects the timing of the transmit status bits (XRDY and XSMT) in the I2CSTR register when in slave transmitter mode. Reset type: SYSRSn 0h (R/W) = See the "Backwards Compatibility Mode Bit, Slave Transmitter" Figure for details. 1h (R/W) = See the "Backwards Compatibility Mode Bit, Slave Transmitter" Figure for details.

33.7.2.13 I2CPSC Register (Offset = Ch) [Reset = 0h]

I2CPSC is shown in Figure 33-32 and described in Table 33-22.

Return to the Summary Table.

The I2C prescaler register (I2CPSC) is a 16-bit register (see Figure 14-21) used for dividing down the I2C input clock to obtain the desired module clock for the operation of the I2C module. See the device-specific data manual for the supported range of values for the module clock frequency.
IPSC must be initialized while the I2C module is in reset (IRS = 0 in I2CMDR). The prescaled frequency takes effect only when IRS is changed to 1. Changing the IPSC value while IRS = 1 has no effect.

Figure 33-32. I2CPSC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
IPSC							
R/W-0h							

Table 33-22. I2CPSC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	IPSC	R/W	0h	I2C prescaler divide-down value. IPSC determines how much the CPU clock is divided to create the module clock of the I2C module: module clock frequency = I2C input clock frequency/(IPSC + 1) Note: IPSC must be initialized while the I2C module is in reset (IRS = 0 in I2CMDR). Reset type: SYSRSn

33.7.2.14 I2CFFTX Register (Offset = 20h) [Reset = 0h]

I2CFFTX is shown in Figure 33-33 and described in Table 33-23.

Return to the Summary Table.

The I2C transmit FIFO register (I2CFFTX) is a 16-bit register that contains the I2C FIFO mode enable bit as well as the control and status bits for the transmit FIFO mode of operation on the I2C peripheral.

Figure 33-33. I2CFFTX Register

15	14	13	12	11	10	9	8
RESERVED	I2CFFEN	TXFFRST	TXFFST				
R-0h	R/W-0h	R/W-0h	R-0h				
7	6	5	4	3	2	1	0
TXFFINT	TXFFINTCLR	TXFFIENA	TXFFIL				
R-0h	R-0/W1S-0h	R/W-0h	R/W-0h				

Table 33-23. I2CFFTX Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	I2CFFEN	R/W	0h	I2C FIFO mode enable bit. This bit must be enabled for either the transmit or the receive FIFO to operate correctly. Reset type: SYSRSn 0h (R/W) = Disable the I2C FIFO mode. 1h (R/W) = Enable the I2C FIFO mode.
13	TXFFRST	R/W	0h	Transmit FIFO Reset Reset type: SYSRSn 0h (R/W) = Reset the transmit FIFO pointer to 0000 and hold the transmit FIFO in the reset state. 1h (R/W) = Enable the transmit FIFO operation.
12-8	TXFFST	R	0h	Contains the status of the transmit FIFO: xxxxx Transmit FIFO contains xxxxx bytes. 00000 Transmit FIFO is empty. Note: Since these bits are reset to zero, the transmit FIFO interrupt flag will be set when the transmit FIFO operation is enabled and the I2C is taken out of reset. This will generate a transmit FIFO interrupt if enabled. To avoid any detrimental effects from this, write a one to the TXFFINTCLR once the transmit FIFO operation is enabled and the I2C is taken out of reset. Reset type: SYSRSn
7	TXFFINT	R	0h	Transmit FIFO interrupt flag. This bit cleared by a CPU write of a 1 to the TXFFINTCLR bit. If the TXFFIENA bit is set, this bit will generate an interrupt when it is set. Reset type: SYSRSn 0h (R/W) = Transmit FIFO interrupt condition has not occurred. 1h (R/W) = Transmit FIFO interrupt condition has occurred.
6	TXFFINTCLR	R-0/W1S	0h	Transmit FIFO Interrupt Flag Clear Reset type: SYSRSn 0h (R/W) = Writes of zeros have no effect. Reads return a 0. 1h (R/W) = Writing a 1 to this bit clears the TXFFINT flag.
5	TXFFIENA	R/W	0h	Transmit FIFO Interrupt Enable Reset type: SYSRSn 0h (R/W) = Disabled. TXFFINT flag does not generate an interrupt when set. 1h (R/W) = Enabled. TXFFINT flag does generate an interrupt when set.

Table 33-23. I2CFFTX Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	TXFFIL	R/W	0h	Transmit FIFO interrupt level. These bits set the status level that will set the transmit interrupt flag. When the TXFFST4-0 bits reach a value equal to or less than these bits, the TXFFINT flag will be set. This will generate an interrupt if the TXFFIENA bit is set. Because the I2C on this device has a 16-level transmit FIFO, these bits cannot be configured for an interrupt of more than 16 FIFO levels. Reset type: SYSRSn

33.7.2.15 I2CFFRX Register (Offset = 21h) [Reset = 0h]

I2CFFRX is shown in Figure 33-34 and described in Table 33-24.

Return to the Summary Table.

The I2C receive FIFO register (I2CFFRX) is a 16-bit register that contains the control and status bits for the receive FIFO mode of operation on the I2C peripheral.

Figure 33-34. I2CFFRX Register

15	14	13	12	11	10	9	8
RESERVED		RXFFRST	RXFFST				
R-0h		R/W-0h	R-0h				
7	6	5	4	3	2	1	0
RXFFINT	RXFFINTCLR	RXFFIENA	RXFFIL				
R-0h	R-0/W1S-0h	R/W-0h	R/W-0h				

Table 33-24. I2CFFRX Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	RESERVED	R	0h	Reserved
13	RXFFRST	R/W	0h	I2C receive FIFO reset bit Reset type: SYSRSn 0h (R/W) = Reset the receive FIFO pointer to 0000 and hold the receive FIFO in the reset state. 1h (R/W) = Enable the receive FIFO operation.
12-8	RXFFST	R	0h	Contains the status of the receive FIFO: xxxxx Receive FIFO contains xxxxx bytes 00000 Receive FIFO is empty. Reset type: SYSRSn
7	RXFFINT	R	0h	Receive FIFO interrupt flag. This bit cleared by a CPU write of a 1 to the RXFFINTCLR bit. If the RXFFIENA bit is set, this bit will generate an interrupt when it is set Reset type: SYSRSn 0h (R/W) = Receive FIFO interrupt condition has not occurred. 1h (R/W) = Receive FIFO interrupt condition has occurred.
6	RXFFINTCLR	R-0/W1S	0h	Receive FIFO interrupt flag clear bit. Reset type: SYSRSn 0h (R/W) = Writes of zeros have no effect. Reads return a zero. 1h (R/W) = Writing a 1 to this bit clears the RXFFINT flag.
5	RXFFIENA	R/W	0h	Receive FIFO interrupt enable bit. Reset type: SYSRSn 0h (R/W) = Disabled. RXFFINT flag does not generate an interrupt when set. 1h (R/W) = Enabled. RXFFINT flag does generate an interrupt when set.
4-0	RXFFIL	R/W	0h	Receive FIFO interrupt level. These bits set the status level that will set the receive interrupt flag. When the RXFFST4-0 bits reach a value equal to or greater than these bits, the RXFFINT flag is set. This will generate an interrupt if the RXFFIENA bit is set. Note: Since these bits are reset to zero, the receive FIFO interrupt flag will be set if the receive FIFO operation is enabled and the I2C is taken out of reset. This will generate a receive FIFO interrupt if enabled. To avoid this, modify these bits on the same instruction as or prior to setting the RXFFRST bit. Because the I2C on this device has a 16-level receive FIFO, these bits cannot be configured for an interrupt of more than 16 FIFO levels. Reset type: SYSRSn

33.7.3 I2C Registers to Driverlib Functions

Table 33-25. I2C Registers to Driverlib Functions

File	Driverlib Function
OAR	
i2c.h	I2C_setOwnAddress
IER	
i2c.c	I2C_enableInterrupt
i2c.c	I2C_disableInterrupt
STR	
i2c.c	I2C_getInterruptStatus
i2c.c	I2C_clearInterruptStatus
i2c.h	I2C_isBusBusy
i2c.h	I2C_getStatus
i2c.h	I2C_clearStatus
CLKL	
i2c.c	I2C_initController
CLKH	
i2c.c	I2C_initController
CNT	
i2c.h	I2C_setDataCount
DRR	
i2c.h	I2C_getData
TAR	
i2c.h	I2C_setTargetAddress
DXR	
i2c.h	I2C_putData
MDR	
i2c.h	I2C_enableModule
i2c.h	I2C_disableModule
i2c.h	I2C_setConfig
i2c.h	I2C_setBitCount
i2c.h	I2C_sendStartCondition
i2c.h	I2C_sendStopCondition
i2c.h	I2C_sendNACK
i2c.h	I2C_getStopConditionStatus
i2c.h	I2C_setAddressMode
i2c.h	I2C_setEmulationMode
i2c.h	I2C_enableLoopback
i2c.h	I2C_disableLoopback
ISRC	
i2c.h	I2C_getInterruptSource
EMDR	
i2c.h	I2C_setExtendedMode
PSC	
i2c.c	I2C_initController
i2c.c	I2C_configureModuleFrequency
i2c.h	I2C_getPreScaler

Table 33-25. I2C Registers to Driverlib Functions (continued)

File	Driverlib Function
FFTX	
i2c.c	I2C_enableInterrupt
i2c.c	I2C_disableInterrupt
i2c.c	I2C_getInterruptStatus
i2c.c	I2C_clearInterruptStatus
i2c.h	I2C_enableFIFO
i2c.h	I2C_disableFIFO
i2c.h	I2C_setFIFOInterruptLevel
i2c.h	I2C_getFIFOInterruptLevel
i2c.h	I2C_getTxFIFOStatus
FFRX	
i2c.c	I2C_enableInterrupt
i2c.c	I2C_disableInterrupt
i2c.c	I2C_getInterruptStatus
i2c.c	I2C_clearInterruptStatus
i2c.h	I2C_enableFIFO
i2c.h	I2C_disableFIFO
i2c.h	I2C_setFIFOInterruptLevel
i2c.h	I2C_getFIFOInterruptLevel
i2c.h	I2C_getRxFIFOStatus

Chapter 34

Multichannel Buffered Serial Port (McBSP)

This chapter describes the multichannel buffered serial port (McBSP).

34.1 Introduction

This device provides up to two high-speed multichannel buffered serial ports (McBSPs) that allow direct interface to codecs and other devices in a system. The McBSP consists of a data-flow path and a control path connected to external devices by six pins as shown in Figure 34-1.

Data is communicated to devices interfaced with the McBSP via the data transmit (DX) pin for transmission and via the data receive (DR) pin for reception. Control information in the form of clocking and frame synchronization is communicated via the following pins: CLKX (transmit clock), CLKR (receive clock), FSX (transmit frame synchronization), and FSR (receive frame synchronization).

The CPU and the DMA controller communicate with the McBSP through 16-bit wide registers accessible via the internal peripheral bus. The CPU or the DMA controller writes the data to be transmitted to the data transmit registers (DXR1, DXR2). Data written to the DXRs is shifted out to the DX pin via the transmit shift registers (XSR1, XSR2). Similarly, receive data on the DR pin is shifted into the receive shift registers (RSR1, RSR2) and copied into the receive buffer registers (RBR1, RBR2). The contents of the receive buffer registers (RBRs) is then copied to the data receive registers (DRRs), which can be read by the CPU or the DMA controller. This allows simultaneous movement of internal and external data communications.

If the serial word length is 8 bits, 12 bits, or 16 bits, the DRR2, RBR2, RSR2, DXR2, and XSR2 registers are not used (written, read, or shifted) For larger word lengths, these registers are needed to hold the most significant bits.

The frame and clock loop-back is implemented at chip level to enable CLKX and FSX to drive CLKR and FSR. If the loop-back is enabled, the CLKR and FSR get their signals from the CLKX and FSX pads; instead of the CLKR and FSR pins.

34.1.1 MCBSP Related Collateral

Foundational Materials

- C2000 Academy - McBSP
- *KeyStone Architecture Multichannel Buffered Serial Port (McBSP)*

34.1.2 Features of the McBSPs

The McBSPs feature:

- Full-duplex communication
- Double-buffered transmission and triple-buffered reception, allowing a continuous data stream
- Independent clocking and framing for reception and transmission
- The capability to send interrupts to the CPU and to send DMA events to the DMA controller
- 128 channels for transmission and reception
- Multichannel selection modes that enable or disable block transfers in each of the channels
- Direct interface to industry-standard codecs, analog interface chips (AICs), and other serially connected A/D and D/A devices
- Support for external generation of clock signals and frame-synchronization signals
- A programmable sample rate generator for internal generation and control of clock signals and frame-synchronization signals
- Programmable polarity for frame-synchronization pulses and clock signals
- Direct interface to:
  - T1/E1 framers
  - IOM-2 compliant devices
  - AC97-compliant devices (the necessary multiphase frame capability is provided)
  - I2S compliant devices
  - SPI devices

- A wide selection of data sizes: 8, 12, 16, 20, 24, and 32 bits

Note

A value of the chosen data size is referred to as a *serial word* or *word* throughout the McBSP chapter. Elsewhere, *word* is used to describe a 16-bit value.

- µ-law and A-law companding
- The option of transmitting/receiving 8-bit data with the LSB first
- Status bits for flagging exception/error conditions
- ABIS mode is not supported

34.1.3 McBSP Pins/Signals

Table 34-1 describes the McBSP interface pins and some internal signals.

Table 34-1. McBSP Interface Pins/Signals

McBSP-A Pin	McBSP-B Pin	Type	Description
MCLKRA	MCLKRB	I/O	Supplies or reflects the receive clock; supplies the input clock of the sample rate generator
MCLKXA	MCLKXB	I/O	Supplies or reflects the transmit clock; supplies the input clock of the sample rate generator
MDRA	MDRB	I	Serial data receive pin
MDXA	MDXB	O	Serial data transmit pin
MFSRA	MFSRB	I/O	Supplies or reflects the receive frame-sync signal; controls sample rate generator synchronization when GSYNC = 1 (see Section 34.4.3)
MFSXA	MFSXB	I/O	Supplies or reflects the transmit frame-sync signal
CPU Interrupt Signals			
MRINT			Receive interrupt to CPU
MXINT			Transmit interrupt to CPU
DMA Events			
REVT			Receive synchronization event to DMA
XEVT			Transmit synchronization event to DMA

34.1.3.1 McBSP Generic Block Diagram

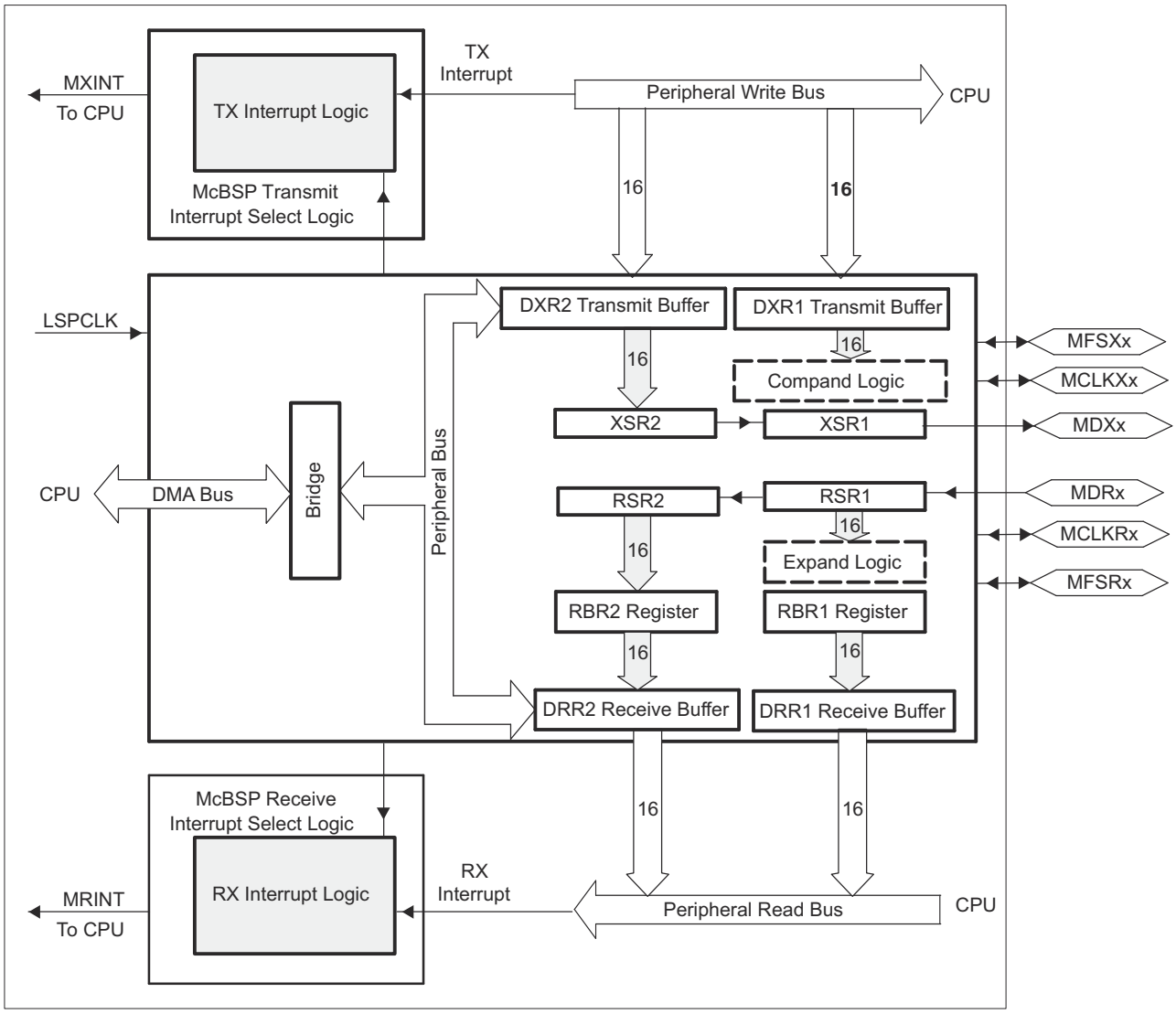
The McBSP consists of a data-flow path and a control path connected to external devices by six pins as shown in Figure 34-1. The figure and the text in this section use generic pin names.

34.2 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

Some IO functionality is defined by GPIO register settings independent of this peripheral. For input signals, the GPIO input qualification must be set to asynchronous mode by setting the appropriate GPxQSELn register bits to 11b. The internal pullups can be configured in the GPyPUD register.

See the *General-Purpose Input/Output (GPIO)* chapter for more details on GPIO mux and settings.



A. Not available in all devices. See the device-specific data sheet.

Figure 34-1. Conceptual Block Diagram of the McBSP

34.3 McBSP Operation

This section addresses the following topics:

- Data transfer process
- Companding (compressing and expanding) data
- Clocking and framing data
- Frame phases
- McBSP reception
- McBSP transmission
- Interrupts and DMA events generated by McBSPs

34.3.1 Data Transfer Process of McBSPs

Figure 34-2 shows a diagram of the McBSP data transfer paths. The McBSP receive operation is triple-buffered, and transmit operation is double-buffered. The use of registers varies, depending on whether the defined length of each serial word is 16 bits.

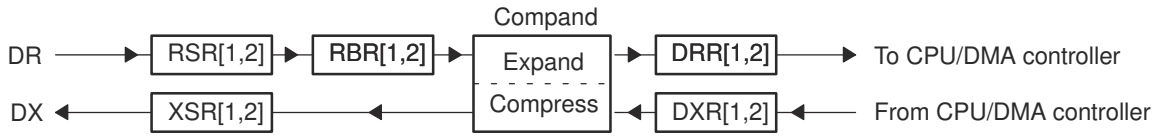


Figure 34-2. McBSP Data Transfer Paths

34.3.1.1 Data Transfer Process for Word Length of 8, 12, or 16 Bits

If the word length is 16 bits or smaller, only one 16-bit register is needed at each stage of the data transfer paths. The registers DRR2, RBR2, RSR2, DXR2, and XSR2 are not used (written, read, or shifted).

Receive data arrives on the DR pin and is shifted into receive shift register 1 (RSR1). Once a full word is received, the content of RSR1 is copied to the receive buffer register 1 (RBR1), that is, if RBR1 is not full with previous data. RBR1 is then copied to the data receive register 1 (DRR1), unless the previous content of DRR1 has not been read by the CPU or the DMA controller. If the companding feature of the McBSP is implemented, the required word length is 8 bits and receive data is expanded into the appropriate format before being passed from RBR1 to DRR1. For more details about reception, see Section 34.3.5.

Transmit data is written by the CPU or the DMA controller to the data transmit register 1 (DXR1). If there is no previous data in the transmit shift register (XSR1), the value in DXR1 is copied to XSR1; otherwise, DXR1 is copied to XSR1 when the last bit of the previous data is shifted out on the DX pin. If selected, the companding module compresses 16-bit data into the appropriate 8-bit format before passing it to XSR1. After transmit frame synchronization, the transmitter begins shifting bits from XSR1 to the DX pin. For more details about transmission, see Section 34.3.6.

34.3.1.2 Data Transfer Process for Word Length of 20, 24, or 32 Bits

If the word length is larger than 16 bits, two 16-bit registers are needed at each stage of the data transfer paths. The registers DRR2, RBR2, RSR2, DXR2, and XSR2 are needed to hold the most significant bits.

Receive data arrives on the DR pin and is shifted first into RSR2 and then into RSR1. Once the full word is received, the contents of RSR2 and RSR1 are copied to RBR2 and RBR1, respectively, if RBR1 is not full. Then the contents of RBR2 and RBR1 are copied to DRR2 and DRR1, respectively, unless the previous content of DRR1 has not been read by the CPU or the DMA controller. The CPU or the DMA controller must read data from DRR2 first and then from DRR1. When DRR1 is read, the next RBR-to-DRR copy occurs. For more details about reception, see Section 34.3.5.

For transmission, the CPU or the DMA controller must write data to DXR2 first and then to DXR1. When new data arrives in DXR1, if there is no previous data in XSR1, the contents of DXR2 and DXR1 are copied to XSR2 and XSR1, respectively; otherwise, the contents of the DXRs are copied to the XSRs when the last bit of the previous data is shifted out on the DX pin. After transmit frame synchronization, the transmitter begins shifting bits from the XSRs to the DX pin. For more details about transmission, see Section 34.3.6.

34.3.2 Companding (Compressing and Expanding) Data

Companding (COMpressing and exPANDing) hardware allows compression and expansion of data in either μ-law or A-law format. The companding standard employed in the United States and Japan is μ-law. The European companding standard is referred to as A-law. The specifications for μ-law and A-law log PCM are part of the CCITT G.711 recommendation.

A-law and μ-law allow 13 bits and 14 bits of dynamic range, respectively. Any values outside this range are set to the most positive or most negative value. Thus, for companding to work best, the data transferred to and from the McBSP via the CPU or DMA controller must be at least 16 bits wide.

The μ-law and A-law formats both encode data into 8-bit code words. Companded data is always 8 bits wide; the appropriate word length bits (RWDLEN1, RWDLEN2, XWDLEN1, XWDLEN2) must therefore be set to 0, indicating an 8-bit wide serial data stream. If companding is enabled and either of the frame phases does not have an 8-bit word length, companding continues as if the word length is 8 bits.

Figure 34-3 illustrates the companding processes. When companding is chosen for the transmitter, compression occurs during the process of copying data from DXR1 to XSR1. The transmit data is encoded according to the specified companding law (A-law or μ-law). When companding is chosen for the receiver, expansion occurs during the process of copying data from RBR1 to DRR1. The receive data is decoded to twos complement format.

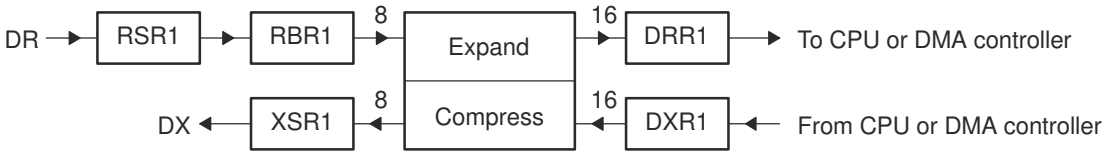


Figure 34-3. Companding Processes

34.3.2.1 Companding Formats

For reception, the 8-bit compressed data in RBR1 is expanded to left-justified 16-bit data in DRR1. The receive sign-extension and justification mode specified in RJUST is ignored when companding is used.

For transmission using μ-law compression, the 14 data bits must be left-justified in DXR1 with the remaining two low-order bits filled with 0s as shown in Figure 34-4.

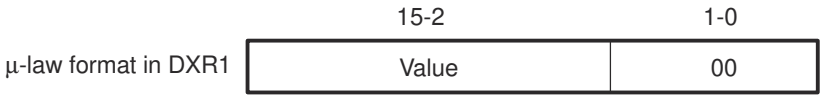


Figure 34-4. μ-Law Transmit Data Companding Format

For transmission using A-law compression, the 13 data bits must be left-justified in DXR1, with the remaining three low-order bits filled with 0s as shown in Figure 34-5.

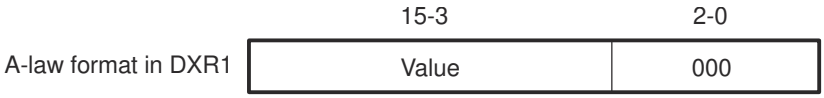


Figure 34-5. A-Law Transmit Data Companding Format

34.3.2.2 Capability to Compand Internal Data

If the McBSP is unused (the serial port transmit and receive sections are reset), the companding hardware can compand internal data. This can be used to:

- Convert linear to the appropriate μ-law or A-law format
- Convert μ-law or A-law to the linear format
- Observe the quantization effects in companding by transmitting linear data and compressing and re-expanding this data. This is useful only if both XCOMPAND and RCOMPAND enable the same companding format.

Figure 34-6 shows two methods by which the McBSP can compand internal data. Data paths for these two methods are used to indicate:

- When both the transmit and receive sections of the serial port are reset, DRR1 and DXR1 are connected internally through the companding logic. Values from DXR1 are compressed, as selected by XCOMPAND, and then expanded, as selected by RCOMPAND. RRDY and XRDY bits are not set. However, data is available in DRR1 within four CPU clocks after being written to DXR1.

  The advantage of this method is the speed. The disadvantage is that there is no synchronization available to the CPU and DMA to control the flow. DRR1 and DXR1 are internally connected if the (X/R)COMPAND bits are set to 10b or 11b (compand using μ-law or A-law).
- The McBSP is enabled in digital loopback mode with companding appropriately enabled by RCOMPAND and XCOMPAND. Receive and transmit interrupts (RINT when RINTM = 0 and XINT when XINTM = 0) or

synchronization events (REVT and XEVT) allow synchronization of the CPU or DMA to these conversions, respectively. Here, the time for this companding depends on the serial bit rate selected.

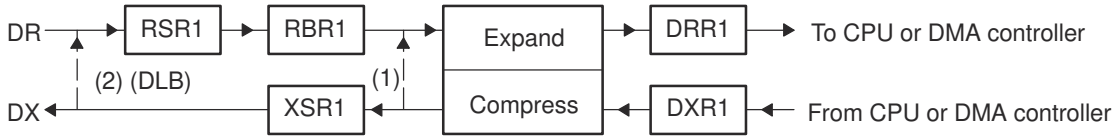


Figure 34-6. Two Methods by Which the McBSP Can Compand Internal Data

34.3.2.3 Reversing Bit Order: Option to Transfer LSB First

Generally, the McBSP transmits or receives all data with the most significant bit (MSB) first. However, certain 8-bit data protocols (that do not use companded data) require the least significant bit (LSB) to be transferred first. If you set XCOMPAND = 01b in XCR2, the bit ordering of 8-bit words is reversed (LSB first) before being sent from the serial port. If you set RCOMPAND = 01b in RCR2, the bit ordering of 8-bit words is reversed during reception. Similar to companding, this feature is enabled only if the appropriate word length bits are set to 0, indicating that 8-bit words are to be transferred serially. If either phase of the frame does not have an 8-bit word length, the McBSP assumes the word length is eight bits, and LSB-first ordering is done.

34.3.3 Clocking and Framing Data

This section explains basic concepts and terminology important for understanding how McBSP data transfers are timed and delimited.

34.3.3.1 Clocking

Note

The McBSP cannot operate at a frequency faster than ½ the LSPCLK frequency. When driving CLKX or CLKR at the pin, choose an appropriate input clock frequency. When using the internal sample rate generator for CLKX and/or CLKR, choose an appropriate input clock frequency and divide down value (CLKDV) (that is, be certain that CLKX or CLKR ≤ LSPCLK/2).

Data is shifted one bit at a time from the DR pin to the RSRs or from the XSRs to the DX pin. The time for each bit transfer is controlled by the rising or falling edge of a clock signal.

The receive clock signal (CLKR) controls bit transfers from the DR pin to the RSRs. The transmit clock signal (CLKX) controls bit transfers from the XSRs to the DX pin. CLKR or CLKX can be derived from a pin at the boundary of the McBSP or derived from inside the McBSP. The polarities of CLKR and CLKX are programmable.

Figure 34-7 shows how the clock signal controls the timing of each bit transfer on the pin.

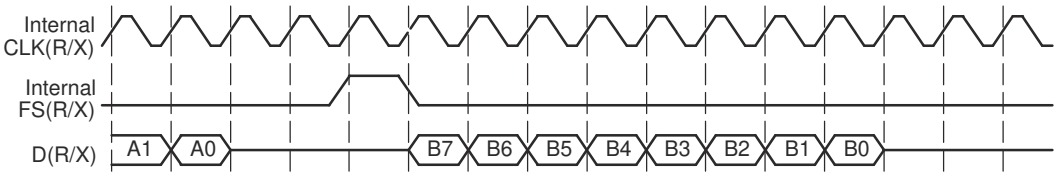


Figure 34-7. Example - Clock Signal Control of Bit Transfer Timing

34.3.3.2 Serial Words

Bits traveling between a shift register (RSR or XSR) and a data pin (DR or DX) are transferred in a group called a serial word. You can define how many bits are in a word.

Bits coming in on the DR pin are held in RSR until RSR holds a full serial word. Only then is the word passed to RBR (and ultimately to the DRR).

During transmission, XSR does not accept new data from DXR until a full serial word has been passed from XSR to the DX pin.

In the exmaple in Figure 34-7, an 8-bit word size was defined (see bits 7 through 0 of word B being transferred).

34.3.3.3 Frames and Frame Synchronization

One or more words are transferred in a group called a frame. You can define how many words are in a frame.

All of the words in a frame are sent in a continuous stream. However, there can be pauses between frame transfers. The McBSP uses frame-synchronization signals to determine when each frame is received/transmitted. When a pulse occurs on a frame-synchronization signal, the McBSP begins receiving/transmitting a frame of data. When the next pulse occurs, the McBSP receives/transmits the next frame, and so on.

Pulses on the receive frame-synchronization (FSR) signal initiate frame transfers on the DR data pin. Pulses on the transmit frame-sync (FSX) signal initiate frame transfers on DX. FSR or FSX can be derived from a pin at the boundary of the McBSP or derived from inside the McBSP.

In Figure 34-7, a one-word frame is transferred when a frame-synchronization pulse occurs.

In McBSP operation, the inactive-to-active transition of the frame-synchronization signal indicates the start of the next frame. For this reason, the frame-synchronization signal may be high for an arbitrary number of clock cycles. Only after the signal is recognized to have gone inactive, and then active again, does the next frame synchronization occur.

34.3.3.4 Generating Transmit and Receive Interrupts

The McBSP can send receive and transmit interrupts to the CPU to indicate specific events in the McBSP. To facilitate detection of frame synchronization, these interrupts can be sent in response to frame-synchronization pulses. Set the appropriate interrupt mode bits to 10b (for reception, RINTM = 10b; for transmission, XINTM = 10b).

34.3.3.4.1 Detecting Frame-Synchronization Pulses, Even in Reset State

Unlike other serial port interrupt modes, this mode can operate while the associated portion of the serial port is in reset (such as activating RINT when the receiver is in reset). In this case, FSRM/FSXM and FSRP/FSXP still select the appropriate source and polarity of frame synchronization. Thus, even when the serial port is in the reset state, these signals are synchronized to the CPU clock and then sent to the CPU in the form of RINT and XINT at the point at which they feed the receiver and transmitter of the serial port. Consequently, a new frame-synchronization pulse can be detected, and after this occurs the CPU can take the serial port out of reset safely.

34.3.3.5 Ignoring Frame-Synchronization Pulses

The McBSP can be configured to ignore transmit and/or receive frame-synchronization pulses. To have the receiver or transmitter recognize frame-synchronization pulses, clear the appropriate frame-synchronization ignore bit (RFIG = 0 for the receiver, XFIG = 0 for the transmitter). To have the receiver or transmitter ignore frame-synchronization pulses until the desired frame length or number of words is reached, set the appropriate frame-synchronization ignore bit (RFIG = 1 for the receiver, XFIG = 1 for the transmitter). For more details on unexpected frame-synchronization pulses, see one of the following topics:

- *Unexpected Receive Frame-Synchronization Pulse* (see Section 34.5.3)
- *Unexpected Transmit Frame-Synchronization Pulse* (see Section 34.5.6)

You can also use the frame-synchronization ignore function for data packing (for more details, see Section 34.11.2).

34.3.3.6 Frame Frequency

The frame frequency is determined by the period between frame-synchronization pulses and is defined by the following equation:

$$\text{Frame Frequency} = \frac{\text{Clock Frequency}}{\text{Number of Clock Cycles Between Frame- Sync Pulses}}$$

The frame frequency can be increased by decreasing the time between frame-synchronization pulses (limited only by the number of bits per frame). As the frame transmit frequency increases, the inactivity period between the data packets for adjacent transfers decreases to zero.

34.3.3.7 Maximum Frame Frequency

The minimum number of clock cycles between frame synchronization pulses is equal to the number of bits transferred per frame. The maximum frame frequency is defined by the following equation:

$$\text{Maximum Frame Frequency} = \frac{\text{Clock Frequency}}{\text{Number of Bits Per Frame}}$$

Figure 34-8 shows the McBSP operating at maximum packet frequency. At maximum packet frequency, the data bits in consecutive packets are transmitted contiguously with no inactivity between bits.

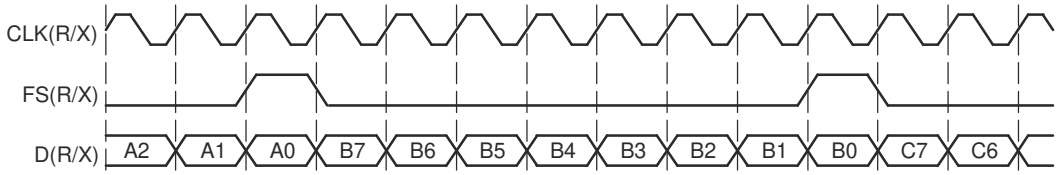


Figure 34-8. McBSP Operating at Maximum Packet Frequency

If there is a 1-bit data delay as shown in Figure 34-8, the frame-synchronization pulse overlaps the last bit transmitted in the previous frame. Effectively, this permits a continuous stream of data, making frame-synchronization pulses redundant. Theoretically, only an initial frame-synchronization pulse is required to initiate a multipacket transfer.

The McBSP supports operation of the serial port in this fashion by ignoring the successive frame-synchronization pulses. Data is clocked into the receiver or clocked out of the transmitter during every clock cycle.

Note

For XDATDLY = 0 (0-bit data delay), the first bit of data is transmitted asynchronously to the internal transmit clock signal (CLKX). For more details, see Section 34.9.13.

34.3.4 Frame Phases

The McBSP allows you to configure each frame to contain one or two phases. The number of words and the number of bits per word can be specified differently for each of the two phases of a frame, allowing greater flexibility in structuring data transfers. For example, you might define a frame as consisting of one phase containing two words of 16 bits each, followed by a second phase consisting of 10 words of 8 bits each. This configuration permits you to compose frames for custom applications or, in general, to maximize the efficiency of data transfers.

34.3.4.1 Number of Phases, Words, and Bits Per Frame

Table 34-2 shows which bit-fields in the receive control registers (RCR1 and RCR2) and in the transmit control registers (XCR1 and XCR2) determine the number of phases per frame, the number of words per frame, and number of bits per word for each phase, for the receiver and transmitter. The maximum number of words per frame is 128 for a single-phase frame and 256 for a dual-phase frame. The number of bits per word can be 8, 12, 16, 20, 24, or 32 bits.

Table 34-2. Register Bits That Determine the Number of Phases, Words, and Bits

Operation	Number of Phases	Words per Frame Set With	Bits per Word Set With
Reception	1 (RPHASE = 0)	RFRLEN1	RWDLEN1
Reception	2 (RPHASE = 1)	RFRLEN1 and RFRLEN2	RWDLEN1 for phase 1
			RWDLEN2 for phase 2
Transmission	1 (XPHASE = 0)	XFRLEN1	XWDLEN1
Transmission	2 (XPHASE = 1)	XFRLEN1 and XFRLEN2	XWDLEN1 for phase 1
			XWDLEN2 for phase 2

34.3.4.2 Single-Phase Frame Example

Figure 34-9 shows an example of a single-phase data frame containing one 8-bit word. Because the transfer is configured for one data bit delay, the data on the DX and DR pins are available one clock cycle after FS(R/X) goes active. The figure makes the following assumptions:

- (R/X)PHASE = 0: Single-phase frame
- (R/X)FRLEN1 = 0b: 1 word per frame
- (R/X)WDLEN1 = 000b: 8-bit word length
- (R/X)FRLEN2 and (R/X)WDLEN2 are ignored
- CLK(X/R)P = 0: Receive data clocked on falling edge; transmit data clocked on rising edge
- FS(R/X)P = 0: Active-high frame-synchronization signals
- (R/X)DATDLY = 01b: 1-bit data delay

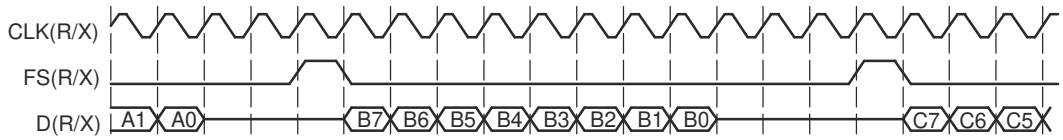
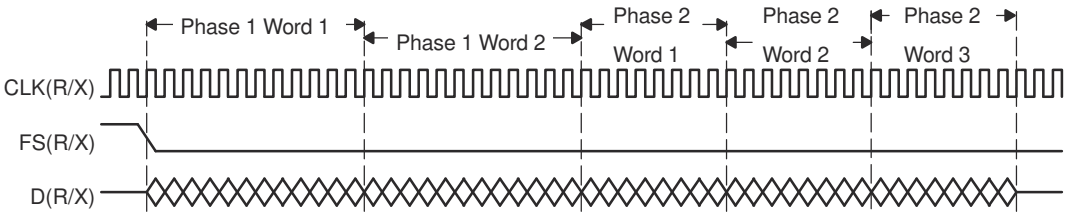


Figure 34-9. Single-Phase Frame for a McBSP Data Transfer

34.3.4.3 Dual-Phase Frame Example

Figure 34-10 shows an example of a frame where the first phase consists of two words of 12 bits each, followed by a second phase of three words of 8 bits each. The entire bit stream in the frame is contiguous. There are no gaps either between words or between phases.



A. XRDY gets asserted once per phase. So, if there are 2 phases, XRDY gets asserted twice (once per phase).

Figure 34-10. Dual-Phase Frame for a McBSP Data Transfer

34.3.4.4 Implementing the AC97 Standard With a Dual-Phase Frame

Figure 34-11 shows an example of the Audio Codec '97 (AC97) standard, which uses the dual-phase frame feature. Notice that words, not individual bits, are shown on the D(R/X) signal. The first phase (P1) consists of a single 16-bit word. The second phase (P2) consists of twelve 20-bit words. The phase configurations are listed after the figure.

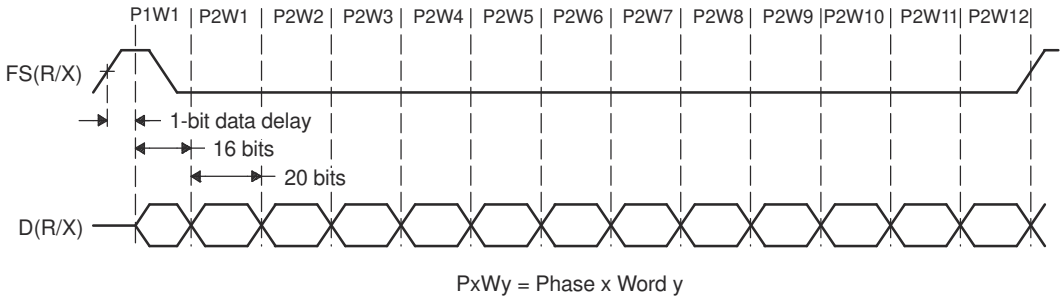


Figure 34-11. Implementing the AC97 Standard With a Dual-Phase Frame

- (R/X)PHASE = 1: Dual-phase frame
- (R/X)FRLEN1 = 0000000b: 1 word in phase 1
- (R/X)WDLEN1 = 010b: 16 bits per word in phase 1
- (R/X)FRLEN2 = 0001011b: 12 words in phase 2
- (R/X)WDLEN2 = 011b: 20 bits per word in phase 2
- CLKRP/CLKXP= 0: Receive data sampled on falling edge of internal CLKR / transmit data clocked on rising edge of internal CLKX
- FSRP/FSXP = 0: Active-high frame-sync signal
- (R/X)DATDLY = 01b: Data delay of 1 clock cycle (1-bit data delay)

Figure 34-12 shows the timing of an AC97-standard data transfer near frame synchronization. In this figure, individual bits are shown on D(R/X). Specifically, it shows the last two bits of phase 2 of one frame and the first four bits of phase 1 of the next frame. Regardless of the data delay, data transfers can occur without gaps. The first bit of the second frame (P1W1B15) immediately follows the last bit of the first frame (P2W12B0). Because a 1-bit data delay has been chosen, the transition on the frame-sync signal can occur when P2W12B0 is transferred.

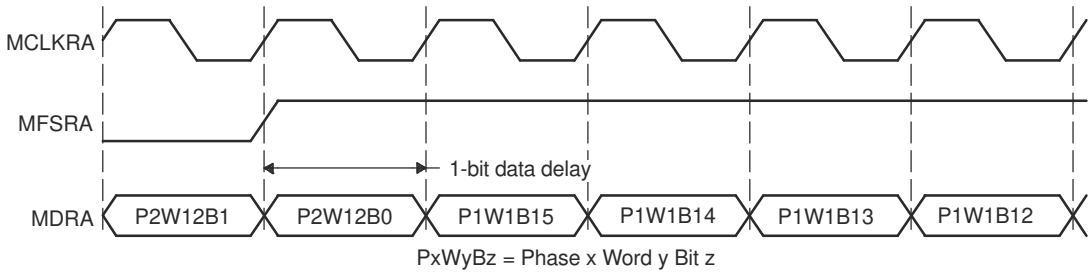
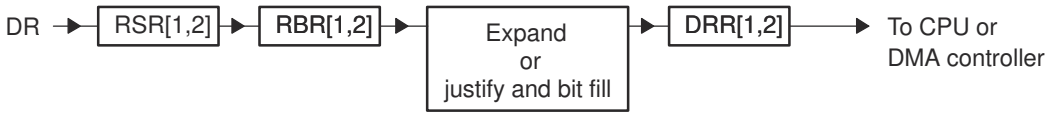


Figure 34-12. Timing of an AC97-Standard Data Transfer Near Frame Synchronization

34.3.5 McBSP Reception

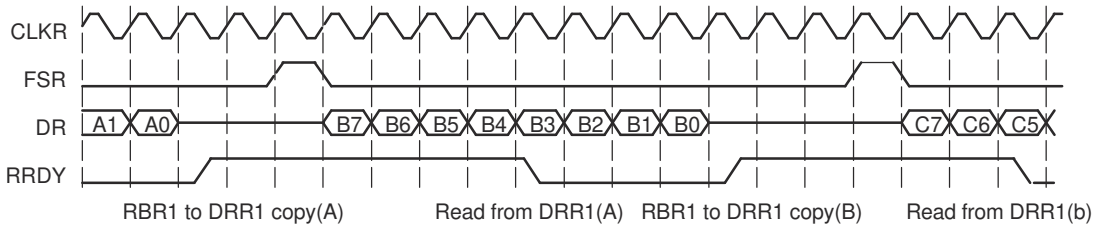
This section explains the fundamental process of reception in the McBSP. For details about how to program the McBSP receiver, see Section 34.8.

Figure 34-13 and Figure 34-14 show how reception occurs in the McBSP. Figure 34-13 shows the physical path for the data. Figure 34-14 is a timing diagram showing signal activity for one possible reception scenario. A description of the process follows the figures.



A. RSR[1,2]: Receive shift registers 1 and 2
B. RBR[1,2]: Receive buffer registers 1 and 2
C. DRR[1,2]: Data receive registers 1 and 2

Figure 34-13. McBSP Reception Physical Data Path



A. CLKR: Internal receive clock
B. FSR: Internal receive frame-synchronization signal
C. DR: Data on DR pin
D. RRDY: Status of receiver ready bit (high is 1)

Figure 34-14. McBSP Reception Signal Activity

The following process describes how data travels from the DR pin to the CPU or to the DMA controller:

1. The McBSP waits for a receive frame-synchronization pulse on internal FSR.
2. When the pulse arrives, the McBSP inserts the appropriate data delay that is selected with the RDATDLY bits of RCR2.
   In the preceding timing diagram, a 1-bit data delay is selected.
3. The McBSP accepts data bits on the DR pin and shifts them into the receive shift registers.
   If the word length is 16 bits or smaller, only RSR1 is used. If the word length is larger than 16 bits, RSR2 and RSR1 are used and RSR2 contains the most-significant bits. For details on choosing a word length, see Section 34.8.8.
4. When a full word is received, the McBSP copies the contents of the receive shift registers to the receive buffer registers, provided that RBR1 is not full with previous data.
   If the word length is 16 bits or smaller, only RBR1 is used. If the word length is larger than 16 bits, RBR2 and RBR1 are used and RBR2 contains the most-significant bits.
5. The McBSP copies the contents of the receive buffer registers into the data receive registers, provided that DRR1 is not full with previous data. When DRR1 receives new data, the receiver ready bit (RRDY) is set in SPCR1. This indicates that received data is ready to be read by the CPU or the DMA controller.
   If the word length is 16 bits or smaller, only DRR1 is used. If the word length is larger than 16 bits, DRR2 and DRR1 are used and DRR2 contains the most-significant bits.
   If companding is used during the copy (RCOMPAND = 10b or 11b in RCR2), the 8-bit compressed data in RBR1 is expanded to a left-justified 16-bit value in DRR1. If companding is disabled, the data copied from RBR[1,2] to DRR[1,2] is justified and bit filled according to the RJUST bits.
6. The CPU or the DMA controller reads the data from the data receive registers. When DRR1 is read, RRDY is cleared and the next RBR-to-DRR copy is initiated.

Note

If both DRRs are required (word length larger than 16 bits), the CPU or the DMA controller must read from DRR2 first and then from DRR1. As soon as DRR1 is read, the next RBR-to-DRR copy is initiated. If DRR2 is not read first, the data in DRR2 is lost.

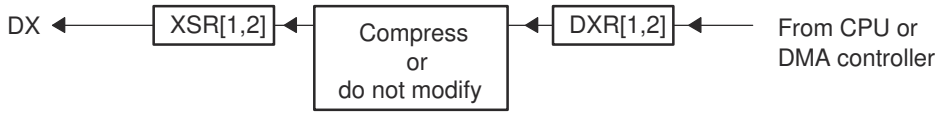
When activity is not properly timed, errors can occur. See the following topics for more details:

- Section 34.5.2
- Section 34.5.3

34.3.6 McBSP Transmission

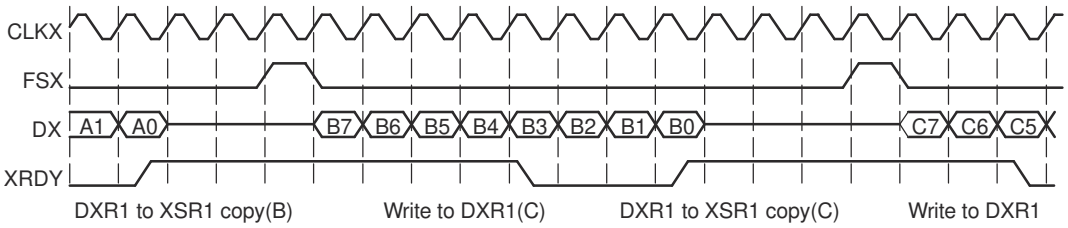
This section explains the fundamental process of transmission in the McBSP. For details about how to program the McBSP transmitter, see Section 34.9.

Figure 34-15 and Figure 34-16 show how transmission occurs in the McBSP. Figure 34-15 shows the physical path for the data. Figure 34-16 is a timing diagram showing signal activity for one possible transmission scenario. A description of the process follows the figures.



A. XSR[1,2]: Transmit shift registers 1 and 2
B. DXR[1,2]: Data transmit registers 1 and 2

Figure 34-15. McBSP Transmission Physical Data Path



A. CLKX: Internal transmit clock
B. FSX: Internal transmit frame-synchronization signal
C. DX: Data on DX pin
D. XRDY: Status of transmitter ready bit (high is 1)

Figure 34-16. McBSP Transmission Signal Activity

1. The CPU or the DMA controller writes data to the data transmit registers. When DXR1 is loaded, the transmitter ready bit (XRDY) is cleared in SPCR2 to indicate that the transmitter is not ready for new data. If the word length is 16 bits or smaller, only DXR1 is used. If the word length is larger than 16 bits, DXR2 and DXR1 are used and DXR2 contains the most-significant bits. For details on choosing a word length, see Section 34.9.9.

   Note

   If both DXRs are needed (word length larger than 16 bits), the CPU or the DMA controller must load DXR2 first and then load DXR1. As soon as DXR1 is loaded, the contents of both DXRs are copied to the transmit shift registers (XSRs), as described in the next step. If DXR2 is not loaded first, the previous content of DXR2 is passed to the XSR2.

2. When new data arrives in DXR1, the McBSP copies the content of the data transmit registers to the transmit shift registers. In addition, the transmit ready bit (XRDY) is set. This indicates that the transmitter is ready to accept new data from the CPU or the DMA controller.

If the word length is 16 bits or smaller, only XSR1 is used. If the word length is larger than 16 bits, XSR2 and XSR1 are used and XSR2 contains the most-significant bits.
If companding is used during the transfer (XCOMPAND = 10b or 11b in XCR2), the McBSP compresses the 16-bit data in DXR1 to 8-bit data in the μ-law or A-law format in XSR1. If companding is disabled, the McBSP passes data from the DXRs to the XSRs without modification.

3. The McBSP waits for a transmit frame-synchronization pulse on internal FSX.
4. When the pulse arrives, the McBSP inserts the appropriate data delay that is selected with the XDATDLY bits of XCR2.
   In the preceding timing diagram (Figure 34-16), a 1-bit data delay is selected.
5. The McBSP shifts data bits from the transmit shift registers to the DX pin.

When activity is not properly timed, errors can occur. See the following topics for more details:

- Section 34.5.4
- Section 34.5.5
- Section 34.5.6

34.3.7 Interrupts and DMA Events Generated by a McBSP

The McBSP sends notification of important events to the CPU and DMA via the internal signals shown in Table 34-3.

Table 34-3. Interrupts and DMA Events Generated by a McBSP

Internal Signal	Description
RINT	Receive interrupt The McBSP sends a receive interrupt request to the CPU based upon a selected condition in the receiver of the McBSP (a condition selected by the RINTM bits of SPCR1).
XINT	Transmit interrupt The McBSP sends a transmit interrupt request to the CPU based upon a selected condition in the transmitter of the McBSP (a condition selected by the XINTM bits of SPCR2).
REVT	Receive synchronization event An REVT signal is sent to the DMA when data has been received in the data receive registers (DRRs).
XEVT	Transmit synchronization event An XEVT signal is sent to the DMA when the data transmit registers (DXRs) are ready to accept the next serial word for transmission.

34.4 McBSP Sample Rate Generator

Each McBSP contains a sample rate generator (SRG) that can be programmed to generate an internal data clock (CLKG) and an internal frame-synchronization signal (FSG). CLKG can be used for bit shifting on the data receive (DR) pin and/or the data transmit (DX) pin. FSG can be used to initiate frame transfers on DR and/or DX. Figure 34-17 is a conceptual block diagram of the sample rate generator.

34.4.1 Block Diagram

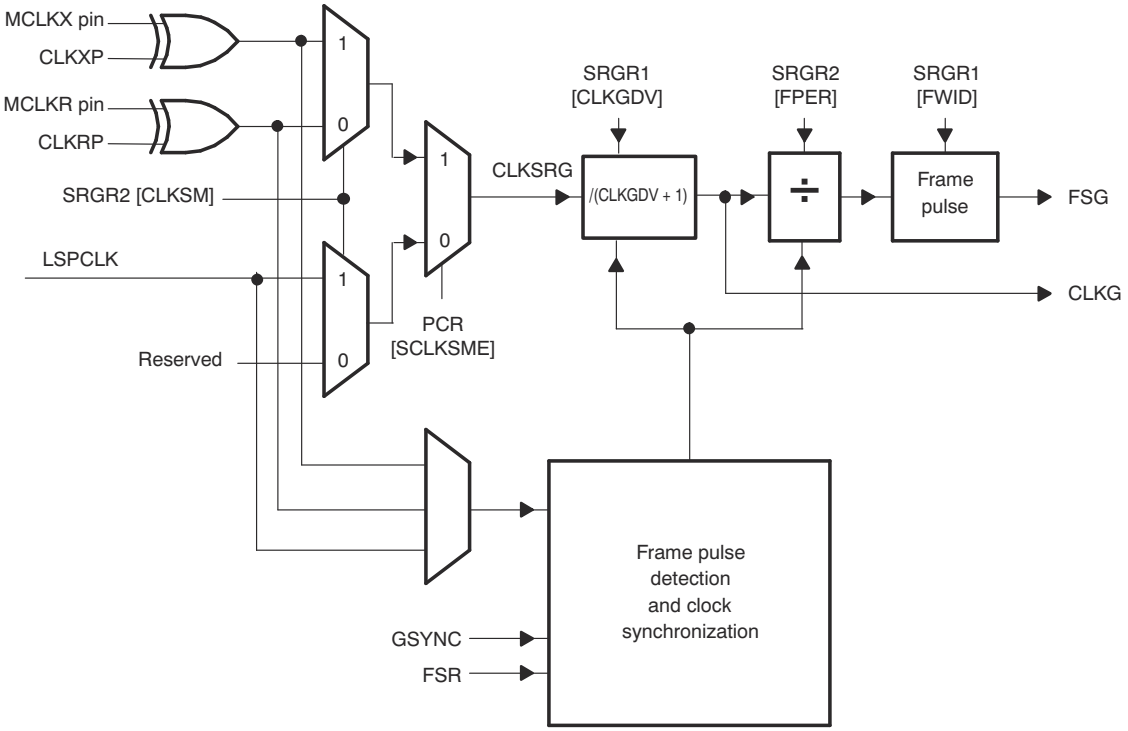


Figure 34-17. Conceptual Block Diagram of the Sample Rate Generator

The source clock for the sample rate generator (labeled CLKSRG in the diagram) can be supplied by the LSPCLK, or by an external pin (MCLKX or MCLKR). The source is selected with the SCLKME bit of PCR and the CLKSM bit of SRGR2. If a pin is used, the polarity of the incoming signal can be inverted with the appropriate polarity bit (CLKXP of PCR or CLKRP of PCR).

The sample rate generator has a three-stage clock divider that gives CLKG and FSG programmability. The three stages provide:

- Clock divide-down. The source clock is divided according to the CLKGDV bits of SRGR1 to produce CLKG.
- Frame period divide-down. CLKG is divided according to the FPER bits of SRGR2 to control the period from the start of a frame-pulse to the start of the next pulse.
- Frame-synchronization pulse-width countdown. CLKG cycles are counted according to the FWID bits of SRGR1 to control the width of each frame-synchronization pulse.

Note

The McBSP cannot operate at a frequency faster than ½ the source clock frequency. You must choose an input clock frequency and a CLKGDV value such that CLKG is less than or equal to ½ the source clock frequency.

In addition to the three-stage clock divider, the sample rate generator has a frame-synchronization pulse detection and clock synchronization module that allows synchronization of the clock divide down with an incoming frame-synchronization pulse on the FSR pin. This feature is enabled or disabled with the GSYNC bit of SRGR2.

For details on getting the sample rate generator ready for operation, see Section 34.4.4.

34.4.1.1 Clock Generation in the Sample Rate Generator

The sample rate generator can produce a clock signal (CLKG) for use by the receiver, the transmitter, or both. Use of the sample rate generator to drive clocking is controlled by the clock mode bits (CLKRM and CLKXM) in the pin control register (PCR). When a clock mode bit is set to 1 (CLKRM = 1 for reception, CLKXM = 1 for transmission), the corresponding data clock (CLKR for reception, CLKX for transmission) is driven by the internal sample rate generator output clock (CLKG).

The effects of CLKRM = 1 and CLKXM = 1 on the McBSP are partially affected by the use of the digital loopback mode and the clock stop (SPI) mode, respectively, as described in Table 34-4. The digital loopback mode (described in Section 34.8.4) is selected with the DLB bit of SPCR1. The clock stop mode (described in Section 34.7.2) is selected with the CLKSTP bits of SPCR1.

When using the sample rate generator as a clock source, make sure the sample rate generator is enabled (GRST = 1).

Table 34-4. Effects of DLB and CLKSTP on Clock Modes

Mode Bit Settings		Effect
CLKRM = 1	DLB = 0 (Digital loopback mode disabled)	CLKR is an output pin driven by the sample rate generator output clock (CLKG).
	DLB = 1 (Digital loopback mode enabled)	CLKR is an output pin driven by internal CLKX. The source for CLKX depends on the CLKXM bit.
CLKXM = 1	CLKSTP = 00b or 01b (Clock stop (SPI) mode disabled)	CLKX is an output pin driven by the sample rate generator output clock (CLKG).
	CLKSTP = 10b or 11b (Clock stop (SPI) mode enabled)	The McBSP is a master in an SPI system. Internal CLKX drives internal CLKR and the shift clocks of any SPI-compliant slave devices in the system. CLKX is driven by the internal sample rate generator.

34.4.1.2 Choosing an Input Clock

The sample rate generator must be driven by an input clock signal from one of the three sources selectable with the SCLKME bit of PCR and the CLKSM bit of SRGR2 (see Table 34-5). When CLKSM = 1, the minimum divide down value in CLKGDV bits is 1. CLKGDV is described in Section 34.4.1.4.

Table 34-5. Choosing an Input Clock for the Sample Rate Generator with the SCLKME and CLKSM Bits

SCLKME	CLKSM	Input Clock for Sample Rate Generator
0	0	Reserved
0	1	LSPCLK
1	0	Signal on MCLKR pin
1	1	Signal on MCLKX pin

34.4.1.3 Choosing a Polarity for the Input Clock

As shown in Figure 34-18, when the input clock is received from a pin, you can choose the polarity of the input clock. The rising edge of CLKSRG generates CLKG and FSG, but you can determine which edge of the input clock causes a rising edge on CLKSRG. The polarity options and their effects are described in Table 34-6.

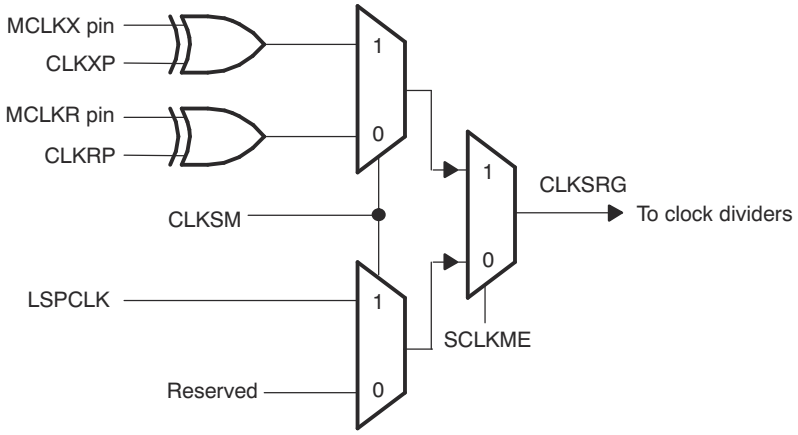


Figure 34-18. Possible Inputs to the Sample Rate Generator and the Polarity Bits

Table 34-6. Polarity Options for the Input to the Sample Rate Generator

Input Clock	Polarity Option	Effect
LSPCLK	Always positive polarity	Rising edge of CPU clock generates transitions on CLKG and FSG.
Signal on MCLKR pin	CLKRP = 0 in PCR	Falling edge on MCLKR pin generates transitions on CLKG and FSG.
	CLKRP = 1 in PCR	Rising edge on MCLKR pin generates transitions on CLKG and FSG.
Signal on MCLKX pin	CLKXP = 0 in PCR	Rising edge on MCLKX pin generates transitions on CLKG and FSG.
	CLKXP = 1 in PCR	Falling edge on MCLKX pin generates transitions on CLKG and FSG.

34.4.1.4 Choosing a Frequency for the Output Clock (CLKG)

The input clock (LSPCLK or external clock) can be divided down by a programmable value to drive CLKG. Regardless of the source to the sample rate generator, the rising edge of CLKSRG (see Section 34.4.1.2) generates CLKG and FSG.

The first divider stage of the sample rate generator creates the output clock from the input clock. This divider stage uses a counter that is preloaded with the divide down value in the CLKGDV bits of SRGR1. The output of this stage is the data clock (CLKG). CLKG has the frequency represented by the following equation:

$$\text{CLKG frequency} = \frac{\text{Input clock frequency}}{(\text{CLKGDV} + 1)}$$

34.4.1.4.1 CLKG Frequency

Thus, the input clock frequency is divided by a value between 1 and 256. When CLKGDV is odd or equal to 0, the CLKG duty cycle is 50%. When CLKGDV is an even value, 2p, representing an odd divide down, the high-state duration is p+1 cycles and the low-state duration is p cycles.

34.4.1.5 Keeping CLKG Synchronized to External MCLKR

When the MCLKR pin is used to drive the sample rate generator (see Section 34.4.1.2), the GSYNC bit in SRGR2 and the FSR pin can be used to configure the timing of the output clock (CLKG) relative to the input clock. Note that this feature is available only when the MCLKR pin is used to feed the external clock.

GSYNC = 1 ensures that the McBSP and an external device are dividing down the input clock with the same phase relationship. If GSYNC = 1, an inactive-to-active transition on the FSR pin triggers a resynchronization of CLKG and generation of FSG.

For more details about synchronization, see Section 34.4.3.

34.4.2 Frame Synchronization Generation in the Sample Rate Generator

The sample rate generator can produce a frame-synchronization signal (FSG) for use by the receiver, the transmitter, or both.

If you want the receiver to use FSG for frame synchronization, make sure FSRM = 1. (When FSRM = 0, receive frame synchronization is supplied via the FSR pin.)

If you want the transmitter to use FSG for frame synchronization, you must set:

- FSXM = 1 in PCR: This indicates that transmit frame synchronization is supplied by the McBSP itself rather than from the FSX pin.
- FSGM = 1 in SRGR2: This indicates that when FSXM = 1, transmit frame synchronization is supplied by the sample rate generator. (When FSGM = 0 and FSXM = 1, the transmitter uses frame-synchronization pulses generated every time data is transferred from DXR[1,2] to XSR[1,2].)

In either case, the sample rate generator must be enabled (GRST = 1) and the frame-synchronization logic in the sample rate generator must be enabled (FRST = 1).

34.4.2.1 Choosing the Width of the Frame-Synchronization Pulse on FSG

Each pulse on FSG has a programmable width. You program the FWID bits of SRGR1, and the resulting pulse width is (FWID + 1) CLKG cycles, where CLKG is the output clock of the sample rate generator.

34.4.2.2 Controlling the Period Between the Starting Edges of Frame-Synchronization Pulses on FSG

You can control the amount of time from the starting edge of one FSG pulse to the starting edge of the next FSG pulse. This period is controlled in one of two ways, depending on the configuration of the sample rate generator:

- If the sample rate generator is using an external input clock and GSYNC = 1 in SRGR2, FSG pulses in response to an inactive-to-active transition on the FSR pin. Thus, the frame-synchronization period is controlled by an external device.
- Otherwise, you program the FPER bits of SRGR2, and the resulting frame-synchronization period is (FPER + 1) CLKG cycles, where CLKG is the output clock of the sample rate generator.

34.4.2.3 Keeping FSG Synchronized to an External Clock

When an external signal is selected to drive the sample rate generator (see Section 34.4.1.2) , the GSYNC bit of SRGR2 and the FSR pin can be used to configure the timing of FSG pulses.

GSYNC = 1 ensures that the McBSP and an external device are dividing down the input clock with the same phase relationship. If GSYNC = 1, an inactive-to-active transition on the FSR pin triggers a resynchronization of CLKG and generation of FSG.

See Section 34.4.3 for more details about synchronization.

34.4.3 Synchronizing Sample Rate Generator Outputs to an External Clock

The sample rate generator can produce a clock signal (CLKG) and a frame-synchronization signal (FSG) based on an input clock signal that is either the CPU clock signal or a signal at the MCLKR or MCLKX pin. When an external clock is selected to drive the sample rate generator, the GSYNC bit of SRGR2 and the FSR pin can be used to control the timing of CLKG and the pulsing of FSG relative to the chosen input clock.

Make GSYNC = 1 when you want the McBSP and an external device to divide down the input clock with the same phase relationship. If GSYNC = 1:

- An inactive-to-active transition on the FSR pin triggers a resynchronization of CLKG and a pulsing of FSG.
- CLKG always begins with a high state after synchronization.
- FSR is always detected at the same edge of the input clock signal that generates CLKG, no matter how long the FSR pulse is.
- The FPER bits of SRGR2 are ignored because the frame-synchronization period on FSG is determined by the arrival of the next frame-synchronization pulse on the FSR pin.

If GSYNC = 0, CLKG runs freely and is not resynchronized, and the frame-synchronization period on FSG is determined by FPER.

34.4.3.1 Operating the Transmitter Synchronously with the Receiver

When GSYNC = 1, the transmitter can operate synchronously with the receiver, provided that:

- FSX is programmed to be driven by FSG (FSGM = 1 in SRGR2 and FSXM = 1 in PCR). If the input FSR has appropriate timing so that it can be sampled by the falling edge of CLKG, it can be used, instead, by setting FSXM = 0 and connecting FSR to FSX externally.
- The sample rate generator clock drives the transmit and receive clocking (CLKRM = CLKXM = 1 in PCR).

34.4.3.2 Synchronization Examples

Figure 34-19 and Figure 34-20 show the clock and frame-synchronization operation with various polarities of CLKR and FSR. These figures assume FWID = 0 in SRGR1, for an FSG pulse that is one CLKG cycle wide. The FPER bits of SRGR2 are not programmed; the period from the start of a frame-synchronization pulse to the start of the next pulse is determined by the arrival of the next inactive-to-active transition on the FSR pin. Each of the figures shows what happens to CLKG when it is initially synchronized and GSYNC = 1, and when it is not initially synchronized and GSYNC = 1. Figure 34-20 has a slower CLKG frequency (it has a larger divide-down value in the CLKGDV bits of SRGR1).

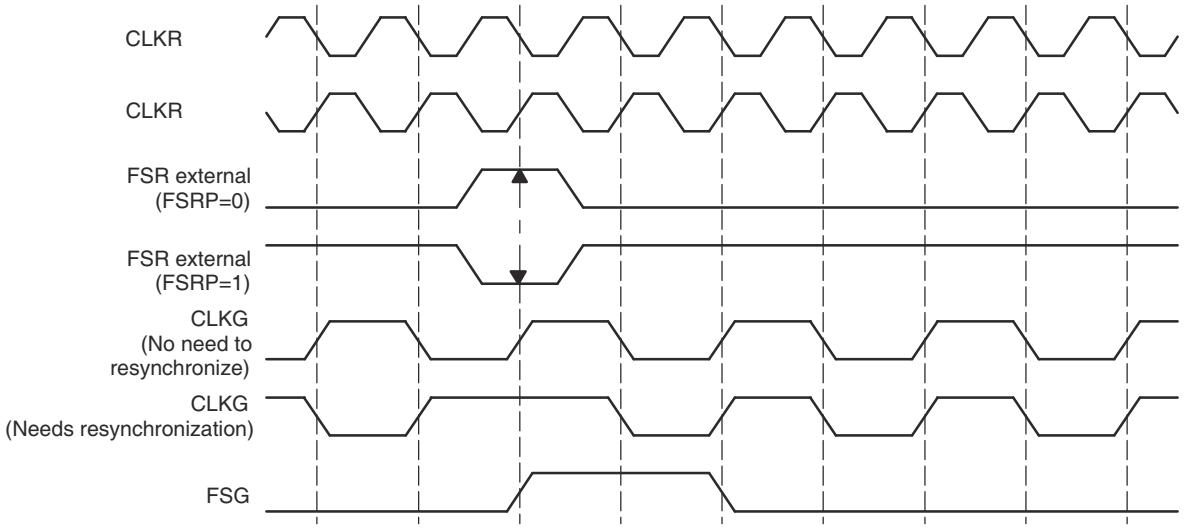


Figure 34-19. CLKG Synchronization and FSG Generation When GSYNC = 1 and CLKGDV = 1

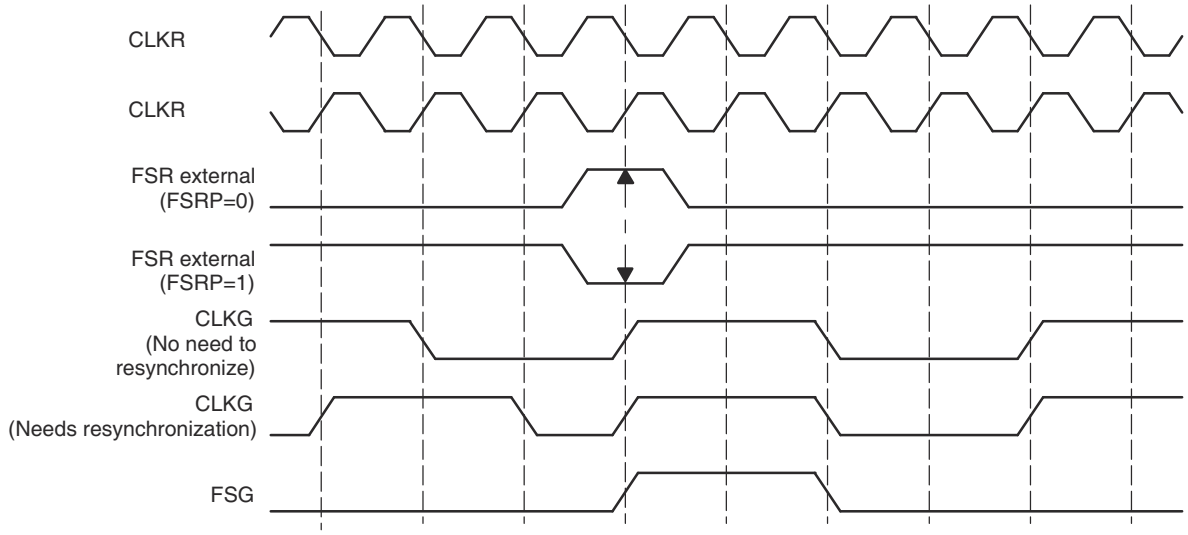


Figure 34-20. CLKG Synchronization and FSG Generation When GSYNC = 1 and CLKGDV = 3

34.4.4 Reset and Initialization Procedure for the Sample Rate Generator

To reset and initialize the sample rate generator:

1. Place the McBSP/sample rate generator in reset.

   During a DSP reset, the sample rate generator, the receiver, and the transmitter reset bits (GRST, RRST, and XRST) are automatically forced to 0. Otherwise, during normal operation, the sample rate generator can be reset by setting GRST = 0 in SPCR2, provided that CLKG and/or FSG is not used by any portion of the McBSP. Depending on your system, you can reset the receiver (RRST = 0 in SPCR1) and reset the transmitter (XRST = 0 in SPCR2).

   If GRST = 0 due to a device reset, CLKG is driven by the CPU clock divided by 2, and FSG is driven inactive-low. If GRST = 0 due to program code, CLKG and FSG are driven low (inactive).
2. Program the registers that affect the sample rate generator.

   Program the sample rate generator registers (SRGR1 and SRGR2) as required for your application. If necessary, other control registers can be loaded with desired values, provided the respective portion of the McBSP (the receiver or transmitter) is in reset.

   After the sample rate generator registers are programmed, wait 2 CLKSRG cycles. This makes sure of proper synchronization internally.
3. Enable the sample rate generator (take the sample rate generator out of reset).

   In SPCR2, make GRST = 1 to enable the sample rate generator.

   After the sample rate generator is enabled, wait two CLKG cycles for the sample rate generator logic to stabilize.

   On the next rising edge of CLKSRG, CLKG transitions to 1 and starts clocking with a frequency equal to the following CLKG frequency equation:

$$\text{CLKG frequency} = \frac{\text{Input clock frequency}}{(\text{CLKGDV} + 1)}$$

   where the input clock is selected with the SCLKME bit of PCR and the CLKSM bit of SRGR2 in one of the configurations shown in Table 34-7.

Table 34-7. Input Clock Selection for Sample Rate Generator

SCLKME	CLKSM	Input Clock for Sample Rate Generator
0	0	Reserved
0	1	LSPCLK
1	0	Signal on MCLKR pin
1	1	Signal on MCLKX pin

4. If necessary, enable the receiver and/or the transmitter.

   If necessary, remove the receiver and/or transmitter from reset by setting RRST and/or XRST = 1.
5. If necessary, enable the frame-synchronization logic of the sample rate generator.

After the required data acquisition setup is done (DXR[1,2] is loaded with data), set GRST = 1 in SPCR2 if an internally generated frame-synchronization pulse is required. FSG is generated with an active-high edge after the programmed number of CLKG clocks (FPER + 1) have elapsed.

34.5 McBSP Exception/Error Conditions

This section describes exception/error conditions and how to handle them.

34.5.1 Types of Errors

There are five serial port events that can constitute a system error:

- Receiver overrun (RFULL = 1)

  This error occurs when DRR1 has not been read since the last RBR-to-DRR copy. Consequently, the receiver does not copy a new word from the RBRs to the DRRs and the RSRs are now full with another new word shifted in from DR. Therefore, RFULL = 1 indicates an error condition wherein any new data that can arrive at this time on DR replaces the contents of the RSRs, and the previous word is lost. The RSRs continue to be overwritten as long as new data arrives on DR and DRR1 is not read. For more details about overrun in the receiver, see Section 34.5.2.
- Unexpected receive frame-synchronization pulse (RSYNCERR = 1)

  This error occurs during reception when RFIG = 0 and an unexpected frame-synchronization pulse occurs. An unexpected frame-synchronization pulse is one that begins the next frame transfer before all the bits of the current frame have been received. Such a pulse causes data reception to abort and restart. If new data has been copied into the RBRs from the RSRs since the last RBR-to-DRR copy, this new data in the RBRs is lost. This is because no RBR-to-DRR copy occurs; the reception has been restarted. For more details about receive frame-synchronization errors, see Section 34.5.3.
- Transmitter data overwrite

  This error occurs when the CPU or DMA controller overwrites data in the DXRs before the data is copied to the XSRs. The overwritten data never reaches the DX pin. For more details about overwrite in the transmitter, see Section 34.5.4.
- Transmitter underflow ($\overline{\text{XEMPTY}}$ = 0)

  If a new frame-synchronization signal arrives before new data is loaded into DXR1, the previous data in the DXRs is sent again. This procedure continues for every new frame-synchronization pulse that arrives until DXR1 is loaded with new data. For more details about underflow in the transmitter, see Section 34.5.5.
- Unexpected transmit frame-synchronization pulse (XSYNCERR = 1)

  This error occurs during transmission when XFIG = 0 and an unexpected frame-synchronization pulse occurs. An unexpected frame-synchronization pulse is one that begins the next frame transfer before all the bits of the current frame have been transferred. Such a pulse causes the current data transmission to abort and restart. If new data has been written to the DXRs since the last DXR-to-XSR copy, the current value in the XSRs is lost. For more details about transmit frame-synchronization errors, see Section 34.5.6.

34.5.2 Overrun in the Receiver

RFULL = 1 in SPCR1 indicates that the receiver has experienced overrun and is in an error condition. RFULL is set when all of the following conditions are met:

1. DRR1 has not been read since the last RBR-to-DRR copy (RRDY = 1).
2. RBR1 is full and an RBR-to-DRR copy has not occurred.
3. RSR1 is full and an RSR1-to-RBR copy has not occurred.

As described in Section 34.3.5, data arriving on DR is continuously shifted into RSR1 (for word length of 16 bits or smaller) or RSR2 and RSR1 (for word length larger than 16 bits). Once a complete word is shifted into the RSR(s), an RSR-to-RBR copy can occur only if the previous data in RBR1 has been copied to DRR1. The RRDY bit is set when new data arrives in DRR1 and is cleared when that data is read from DRR1. Until RRDY = 0, the next RBR-to-DRR copy does not take place, and the data is held in the RSR(s). New data arriving on the DR pin is shifted into RSR(s), and the previous content of the RSR(s) is lost.

You can prevent the loss of data if DRR1 is read no later than 2.5 cycles before the end of the third word is shifted into the RSR1.

Note

If both DRRs are needed (word length larger than 16 bits), the CPU or the DMA controller must read from DRR2 first and then from DRR1. As soon as DRR1 is read, the next RBR-to-DRR copy is initiated. If DRR2 is not read first, the data in DRR2 is lost.

After the receiver starts running from reset, a minimum of three words must be received before RFULL is set. Either of the following events clears the RFULL bit and allows subsequent transfers to be read properly:

- The CPU or DMA controller reads DRR1.
- The receiver is reset individually (RRST = 0) or as part of a device reset.

Another frame-synchronization pulse is required to restart the receiver.

34.5.2.1 Example of Overrun Condition

Figure 34-21 shows the receive overrun condition. Because serial word A is not read from DRR1 before serial word B arrives in RBR1, B is not transferred to DRR1 yet. Another new word ©) arrives and RSR1 is full with this data. DRR1 is finally read, but not earlier than 2.5 cycles before the end of word C. Therefore, new data (D) overwrites word C in RSR1. If DRR1 is not read in time, the next word can overwrite D.

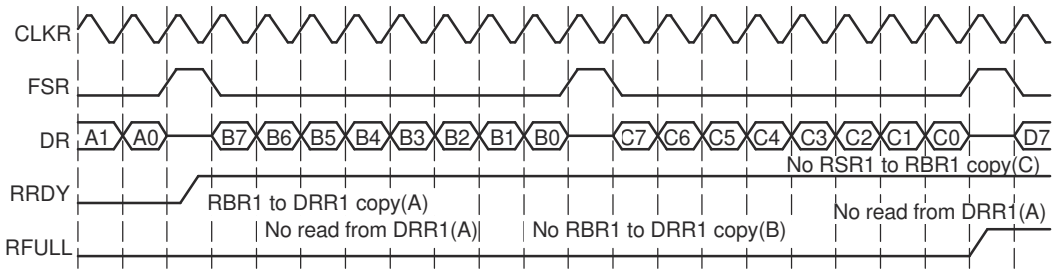


Figure 34-21. Overrun in the McBSP Receiver

34.5.2.2 Example of Preventing Overrun Condition

Figure 34-22 shows the case where RFULL is set, but the overrun condition is prevented by a read from DRR1 at least 2.5 cycles before the next serial word is completely shifted into RSR1. This makes sure that an RBR1-to-DRR1 copy of word B occurs before receiver attempts to transfer word C from RSR1 to RBR1.

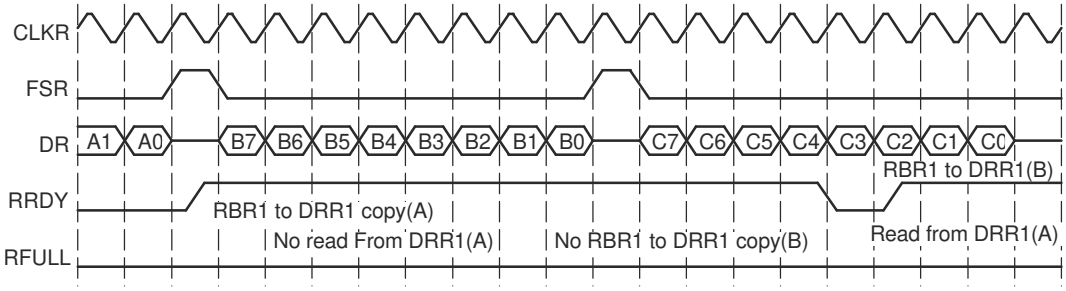


Figure 34-22. Overrun Prevented in the McBSP Receiver

34.5.3 Unexpected Receive Frame-Synchronization Pulse

Section 34.5.3.1 shows how the McBSP responds to any receive frame-synchronization pulses, including an unexpected pulse. Section 34.5.3.2 and Section 34.5.3.3 show an example of a frame-synchronization error and an example of how to prevent such an error, respectively.

34.5.3.1 Possible Responses to Receive Frame-Synchronization Pulses

Figure 34-23 shows the decision tree that the receiver uses to handle all incoming frame-synchronization pulses. The figure assumes that the receiver has been started (RRST = 1 in SPCR1). Case 3 shows where an error occurs.

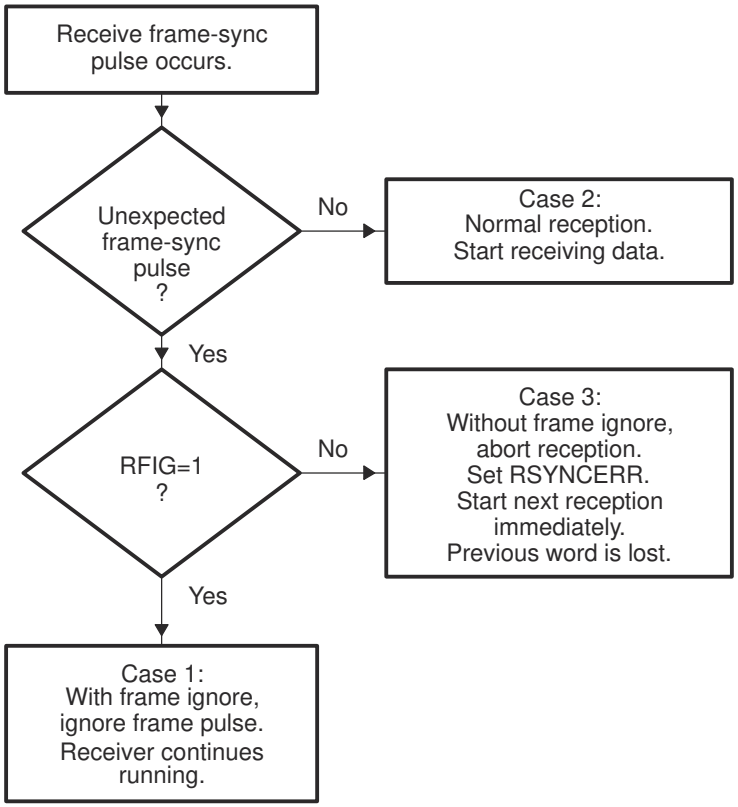


Figure 34-23. Possible Responses to Receive Frame-Synchronization Pulses

Any one of three cases can occur:

- Case 1: Unexpected internal FSR pulses with RFIG = 1 in RCR2. Receive frame-synchronization pulses are ignored, and the reception continues.
- Case 2: Normal serial port reception. Reception continues normally because the frame-synchronization pulse is not unexpected. There are three possible reasons why a receive operation might *not* be in progress when the pulse occurs:
  - The FSR pulse is the first after the receiver is enabled (RRST = 1 in SPCR1).
  - The FSR pulse is the first after DRR[1,2] is read, clearing a receiver full (RFULL = 1 in SPCR1) condition.
  - The serial port is in the interpacket intervals. The programmed data delay for reception (programmed with the RDATDLY bits in RCR2) may start during these interpacket intervals for the first bit of the next word to be received. Thus, at maximum frame frequency, frame synchronization can still be received 0 to 2 clock cycles before the first bit of the synchronized frame.
- Case 3: Unexpected receive frame synchronization with RFIG = 0 (frame-synchronization pulses not ignored). Unexpected frame-synchronization pulses can originate from an external source or from the internal sample rate generator.

  If a frame-synchronization pulse starts the transfer of a new frame before the current frame is fully received, this pulse is treated as an unexpected frame-synchronization pulse, and the receiver sets the receive frame-synchronization error bit (RSYNCERR) in SPCR1. RSYNCERR can be cleared only by a receiver reset or by a write of 0 to this bit.

  If you want the McBSP to notify the CPU of receive frame-synchronization errors, you can set a special receive interrupt mode with the RINTM bits of SPCR1. When RINTM = 11b, the McBSP sends a receive interrupt (RINT) request to the CPU each time that RSYNCERR is set.

34.5.3.2 Example of Unexpected Receive Frame-Synchronization Pulse

Figure 34-30 shows an unexpected receive frame-synchronization pulse during normal operation of the serial port, with time intervals between data packets. When the unexpected frame-synchronization pulse occurs, the RSYNCERR bit is set, the reception of data B is aborted, and the reception of data C begins. In addition, if RINTM = 11b, the McBSP sends a receive interrupt (RINT) request to the CPU.

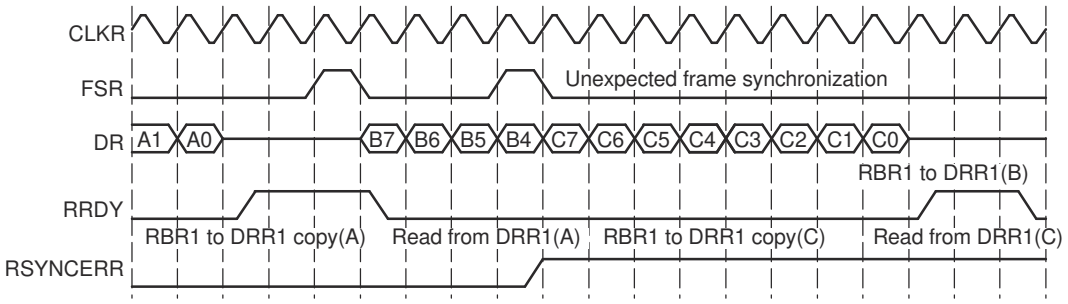


Figure 34-24. An Unexpected Frame-Synchronization Pulse During a McBSP Reception

34.5.3.3 Preventing Unexpected Receive Frame-Synchronization Pulses

Each frame transfer can be delayed by 0, 1, or 2 MCLKR cycles, depending on the value in the RDATDLY bits of RCR2. For each possible data delay, Figure 34-25 shows when a new frame-synchronization pulse on FSR can safely occur relative to the last bit of the current frame.

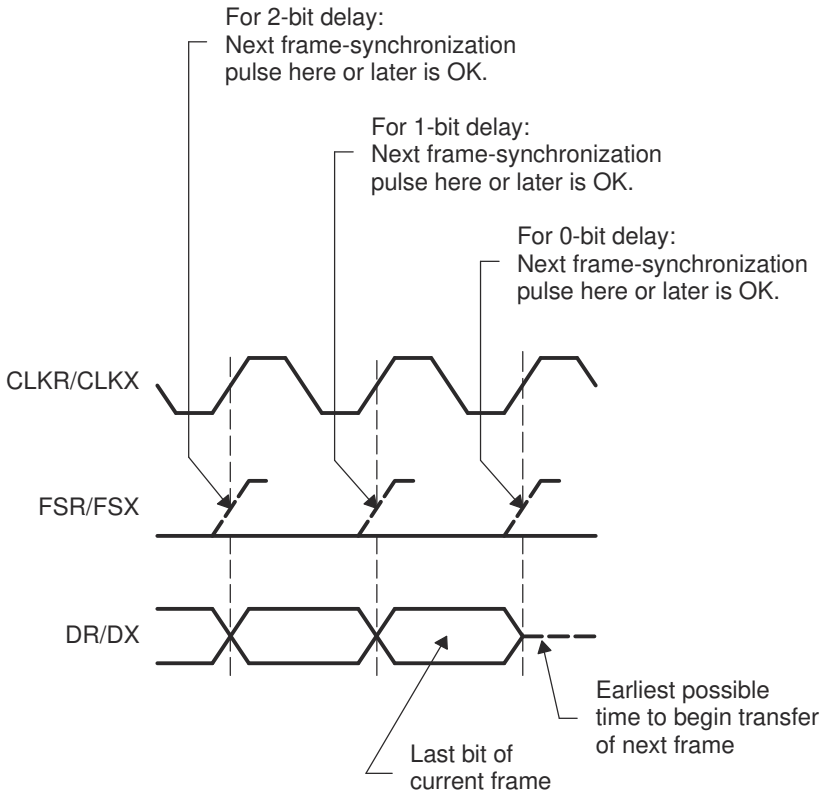


Figure 34-25. Proper Positioning of Frame-Synchronization Pulses

34.5.4 Overwrite in the Transmitter

As described in Section 34.3.6), the transmitter must copy the data previously written to the DXR(s) by the CPU or DMA controller into the XSR(s) and then shift each bit from the XSR(s) to the DX pin. If new data is written to the DXR(s) before the previous data is copied to the XSR(s), the previous data in the DXR(s) is overwritten and thus lost.

34.5.4.1 Example of Overwrite Condition

Figure 34-26 shows what happens if the data in DXR1 is overwritten before being transmitted. Initially, DXR1 is loaded with data C. A subsequent write to DXR1 overwrites C with D before C is copied to XSR1. Thus, C is never transmitted on DX.

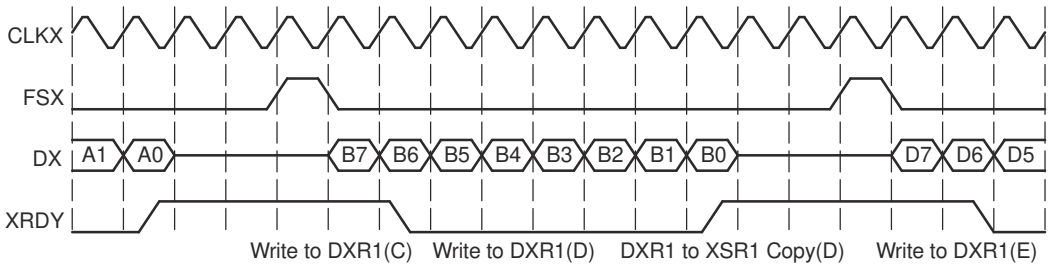


Figure 34-26. Data in the McBSP Transmitter Overwritten and Thus Not Transmitted

34.5.4.2 Preventing Overwrites

You can prevent CPU overwrites by making the CPU:

- Poll for XRDY = 1 in SPCR2 before writing to the DXRs. XRDY is set when data is copied from DXR1 to XSR1 and is cleared when new data is written to DXR1.

- Wait for a transmit interrupt (XINT) before writing to the DXRs. When XINTM = 00b in SPCR2, the transmitter sends XINT to the CPU each time XRDY is set.

You can prevent DMA overwrites by synchronizing DMA transfers to the transmit synchronization event XEVT. The transmitter sends an XEVT signal each time XRDY is set.

34.5.5 Underflow in the Transmitter

The McBSP indicates a transmitter empty (or underflow) condition by clearing the $\overline{\text{XEMPTY}}$ bit in SPCR2. Either of the following events activates $\overline{\text{XEMPTY}}$ ($\overline{\text{XEMPTY}}$ = 0):

- DXR1 has not been loaded since the last DXR-to-XSR copy, and all bits of the data word in the XSR(s) have been shifted out on the DX pin.
- The transmitter is reset (by forcing XRST = 0 in SPCR2, or by a device reset) and is then restarted.

In the underflow condition, the transmitter continues to transmit the old data that is in the DXR(s) for every new transmit frame-synchronization signal, until a new value is loaded into DXR1 by the CPU or the DMA controller.

Note

If both DXRs are needed (word length larger than 16 bits), the CPU or the DMA controller must load DXR2 first and then load DXR1. As soon as DXR1 is loaded, the contents of both DXRs are copied to the transmit shift registers (XSRs). If DXR2 is not loaded first, the previous content of DXR2 is passed to the XSR2.

XEMPTY is deactivated (XEMPTY = 1) when a new word in DXR1 is transferred to XSR1. If FSXM = 1 in PCR and FSGM = 0 in SRGR2, the transmitter generates a single internal FSX pulse in response to a DXR-to-XSR copy. Otherwise, the transmitter waits for the next frame-synchronization pulse before sending out the next frame on DX.

When the transmitter is taken out of reset (XRST = 1), it is in a transmitter ready (XRDY = 1 in SPCR2) and transmitter empty (XEMPTY = 0) state. If DXR1 is loaded by the CPU or the DMA controller before internal FSX goes active high, a valid DXR-to-XSR transfer occurs. This allows for the first word of the first frame to be valid even before the transmit frame-synchronization pulse is generated or detected. Alternatively, if a transmit frame-synchronization pulse is detected before DXR1 is loaded, zeros are output on DX.

34.5.5.1 Example of the Underflow Condition

Figure 34-27 shows an underflow condition. After B is transmitted, DXR1 is not reloaded before the subsequent frame-synchronization pulse. Thus, B is again transmitted on DX.

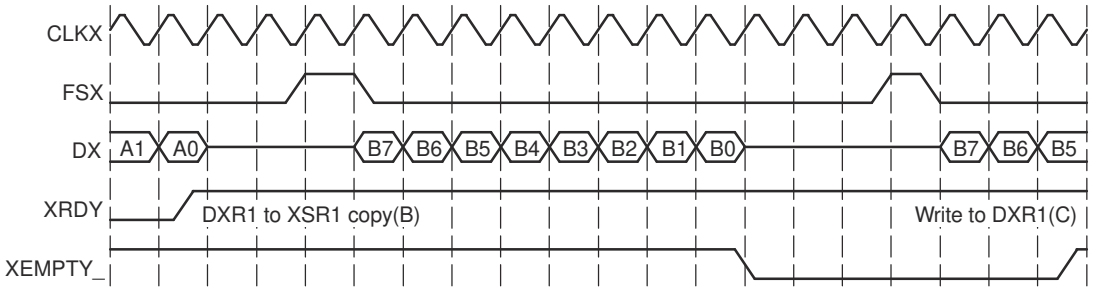


Figure 34-27. Underflow During McBSP Transmission

34.5.5.2 Example of Preventing Underflow Condition

Figure 34-28 shows the case of writing to DXR1 just before an underflow condition can otherwise occur. After B is transmitted, C is written to DXR1 before the next frame-synchronization pulse. As a result, there is no underflow; B is not transmitted twice.

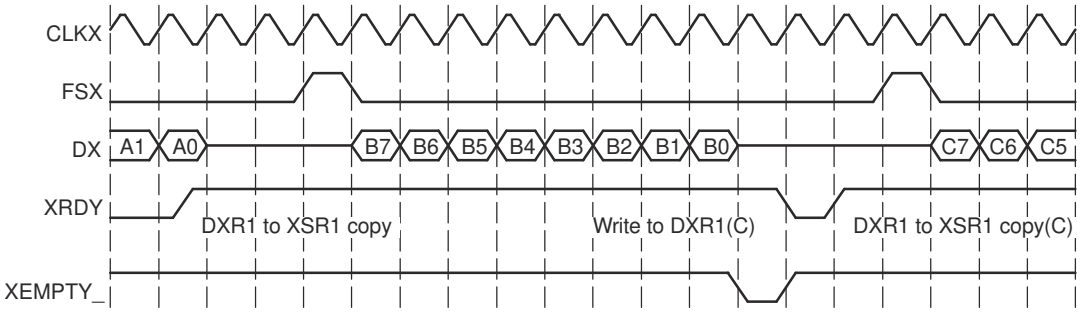


Figure 34-28. Underflow Prevented in the McBSP Transmitter

34.5.6 Unexpected Transmit Frame-Synchronization Pulse

Section 34.5.6.1 shows how the McBSP responds to any transmit frame-synchronization pulses, including an unexpected pulse. Section 34.5.6.2 and Section 34.5.6.3 show examples of a frame-synchronization error and an example of how to prevent such an error, respectively.

34.5.6.1 Possible Responses to Transmit Frame-Synchronization Pulses

Figure 34-29 shows the decision tree that the transmitter uses to handle all incoming frame-synchronization pulses. The figure assumes that the transmitter has been started (XRST = 1 in SPCR2). Case 3 shows where an error occurs.

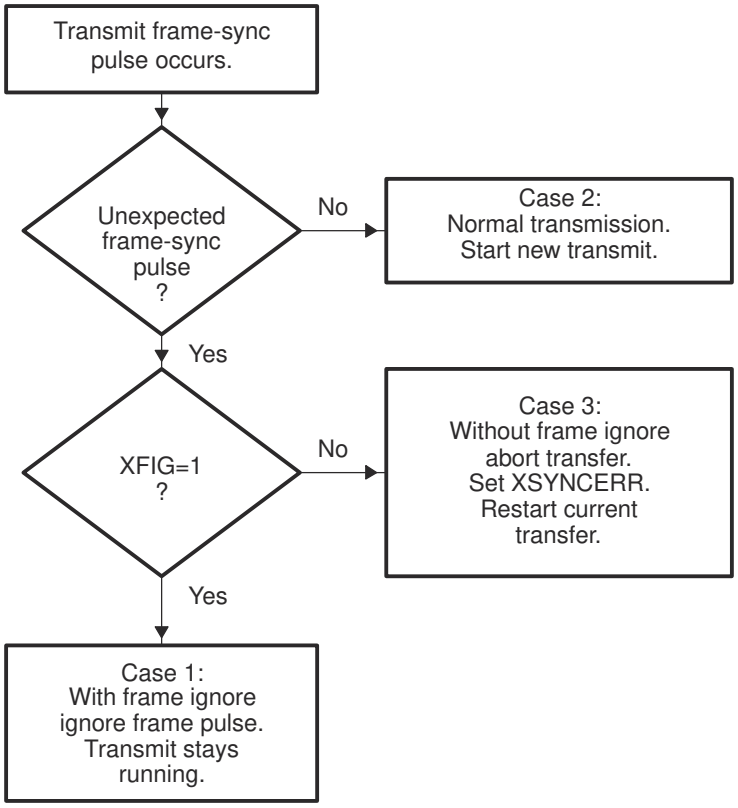


Figure 34-29. Possible Responses to Transmit Frame-Synchronization Pulses

Any one of three cases can occur:

- Case 1: Unexpected internal FSX pulses with XFIG = 1 in XCR2. Transmit frame-synchronization pulses are ignored, and the transmission continues.
- Case 2: Normal serial port transmission. Transmission continues normally because the frame-synchronization pulse is not unexpected. There are two possible reasons why a transmit operations might *not* be in progress when the pulse occurs:

  This FSX pulse is the first after the transmitter is enabled (XRST = 1).

  The serial port is in the interpacket intervals. The programmed data delay for transmission (programmed with the XDATDLY bits of XCR2) may start during these interpacket intervals before the first bit of the previous word is transmitted. Thus, at maximum packet frequency, frame synchronization can still be received 0 to 2 clock cycles before the first bit of the synchronized frame.
- Case 3: Unexpected transmit frame synchronization with XFIG = 0 (frame-synchronization pulses not ignored). Unexpected frame-synchronization pulses can originate from an external source or from the internal sample rate generator.

  If a frame-synchronization pulse starts the transfer of a new frame before the current frame is fully transmitted, this pulse is treated as an unexpected frame-synchronization pulse, and the transmitter sets the transmit frame-synchronization error bit (XSYNCERR) in SPCR2. XSYNCERR can be cleared only by a transmitter reset or by a write of 0 to this bit.

  If you want the McBSP to notify the CPU of frame-synchronization errors, you can set a special transmit interrupt mode with the XINTM bits of SPCR2. When XINTM = 11b, the McBSP sends a transmit interrupt (XINT) request to the CPU each time that XSYNCERR is set.

34.5.6.2 Example of Unexpected Transmit Frame-Synchronization Pulse

Section 34.5.3.2 shows an unexpected transmit frame-synchronization pulse during normal operation of the serial port with intervals between the data packets. When the unexpected frame-synchronization pulse occurs, the XSYNCERR bit is set and the transmission of data B is restarted because no new data has been passed to XSR1 yet. In addition, if XINTM = 11b, the McBSP sends a transmit interrupt (XINT) request to the CPU.

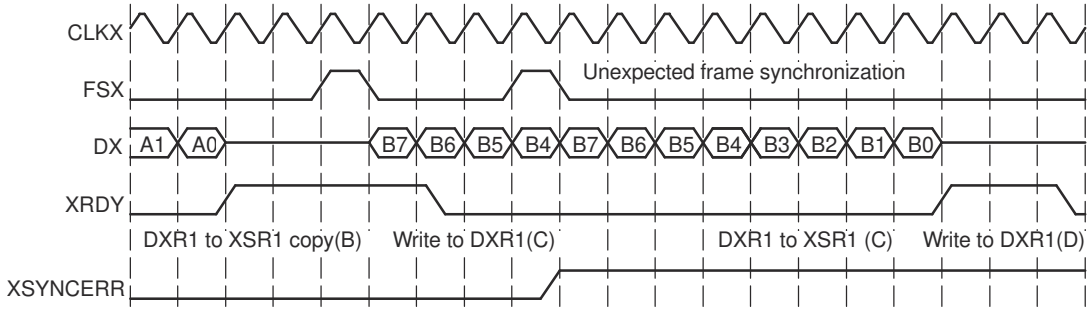


Figure 34-30. An Unexpected Frame-Synchronization Pulse During a McBSP Transmission

34.5.6.3 Preventing Unexpected Transmit Frame-Synchronization Pulses

Each frame transfer can be delayed by 0, 1, or 2 CLKX cycles, depending on the value in the XDATDLY bits of XCR2. For each possible data delay, Figure 34-31 shows when a new frame-synchronization pulse on FSX can safely occur relative to the last bit of the current frame.

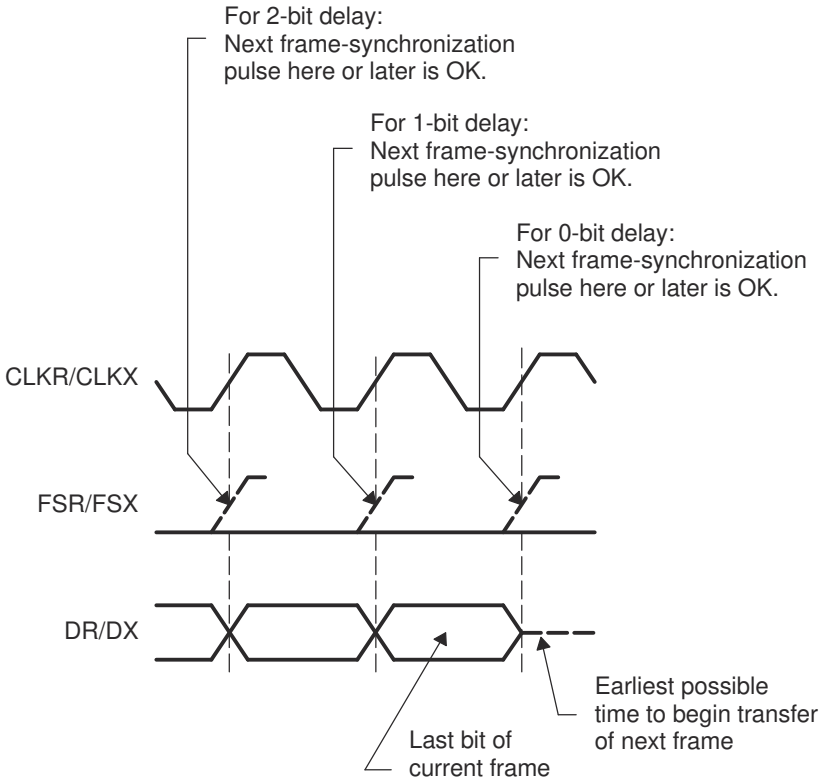


Figure 34-31. Proper Positioning of Frame-Synchronization Pulses

34.6 Multichannel Selection Modes

This section discusses the multichannel selection modes for the McBSP.

34.6.1 Channels, Blocks, and Partitions

A McBSP channel is a time slot for shifting in/out the bits of one serial word. Each McBSP supports up to 128 channels for reception and 128 channels for transmission.

In the receiver and in the transmitter, the 128 available channels are divided into eight blocks that each contain 16 contiguous channels (see Table 34-8 through Table 34-10):

- It is possible to have two receive partitions (A and B) and 8 transmit partitions (A to H).
- McBSP can transmit/receive on selected channels.
- Each channel partition has a dedicated channel-enable register. Each bit controls whether data flow is allowed or prevented in one of the channels assigned to that partition.
- There are three transmit multichannel modes and one receive multichannel mode.

The blocks are assigned to partitions according to the selected partition mode. In the two-partition mode (described in Section 34.6.4), you assign one even-numbered block (0, 2, 4, or 6) to partition A and one odd-numbered block (1, 3, 5, or 7) to partition B. In the 8-partition mode (described in Section 34.6.5), blocks 0 through 7 are automatically assigned to partitions, A through H, respectively.

The number of partitions for reception and the number of partitions for transmission are independent. For example, it is possible to use two receive partitions (A and B) and eight transmit partitions (A to H).

Table 34-8. Block - Channel Assignment

Block	Channels
0	0 -15
1	16 - 31
2	32 - 47
3	48 - 63
4	64 - 79
5	80 - 95
6	96 - 111
7	112 - 127

Table 34-9. 2-Partition Mode

Partition	Blocks
A	0, 2, 4, or 6
B	1, 3, 5, or 7

Table 34-10. 8-Partition Mode

Partition	Blocks	Channels
A	0	0 -15
B	1	16 - 31
C	2	32 - 47
D	3	48 - 63
E	4	64 - 79
F	5	80 - 95
G	6	96 - 111
H	7	112 - 127

34.6.2 Multichannel Selection

When a McBSP uses a time-division multiplexed (TDM) data stream while communicating with other McBSPs or serial devices, the McBSP may need to receive and/or transmit on only a few channels. To save memory and bus bandwidth, you can use a multichannel selection mode to prevent data flow in some of the channels.

Each channel partition has a dedicated channel enable register. If the appropriate multichannel selection mode is on, each bit in the register controls whether data flow is allowed or prevented in one of the channels that is assigned to that partition.

The McBSP has one receive multichannel selection mode (described in Section 34.6.6) and three transmit multichannel selection modes (described in Section 34.6.7).

34.6.3 Configuring a Frame for Multichannel Selection

Before you enable a multichannel selection mode, make sure you properly configure the data frame:

- Select a single-phase frame (RPHASE/XPHASE = 0). Each frame represents a TDM data stream.
- Set a frame length (in RFRLEN1/XFRLEN1) that includes the highest-numbered channel to be used. For example, if you plan to use channels 0, 15, and 39 for reception, the receive frame length must be at least 40 (RFRLEN1 = 39). If XFRLEN1 = 39 in this case, the receiver creates 40 time slots per frame but only receives data during time slots 0, 15, and 39 of each frame.

34.6.4 Using Two Partitions

For multichannel selection operation in the receiver and/or the transmitter, you can use two partitions or eight partitions (described in Section 34.6.5). If you choose the 2-partition mode (RMCME = 0 for reception, XMCME = 0 for transmission), McBSP channels are activated using an alternating scheme. In response to a frame-synchronization pulse, the receiver or transmitter begins with the channels in partition A and then alternates between partitions B and A until the complete frame has been transferred. When the next frame-synchronization pulse occurs, the next frame is transferred beginning with the channels in partition A.

34.6.4.1 Assigning Blocks to Partitions A and B

For reception, any two of the eight receive-channel blocks can be assigned to receive partitions A and B, which means up to 32 receive channels can be enabled at any given point in time. Similarly, any two of the eight transmit-channel blocks (up 32 enabled transmit channels) can be assigned to transmit partitions A and B.

For reception:

- Assign an even-numbered channel block (0, 2, 4, or 6) to receive partition A by writing to the RPABLK bits. In the receive multichannel selection mode (described in Section 34.6.6), the channels in this partition are controlled by receive channel enable register A (RCERA).
- Assign an odd-numbered block (1, 3, 5, or 7) to receive partition B with the RPBBLK bits. In the receive multichannel selection mode, the channels in this partition are controlled by receive channel enable register B (RCERB).

For transmission:

- Assign an even-numbered channel block (0, 2, 4, or 6) to transmit partition A by writing to the XPABLK bits. In one of the transmit multichannel selection modes (described in Section 34.6.7), the channels in this partition are controlled by transmit channel enable register A (XCERA).
- Assign an odd-numbered block (1, 3, 5, or 7) to transmit partition B with the XPBBLK bits. In one of the transmit multichannel selection modes, the channels in this partition are controlled by transmit channel enable register B (XCERB).

Figure 34-32 shows an example of alternating between the channels of partition A and the channels of partition B. Channels 0-15 have been assigned to partition A, and channels 16-31 have been assigned to partition B. In response to a frame-synchronization pulse, the McBSP begins a frame transfer with partition A and then alternates between partitions B and A until the complete frame is transferred.

As explained in Section 34.6.4.2, you can dynamically change which blocks of channels are assigned to the partitions.

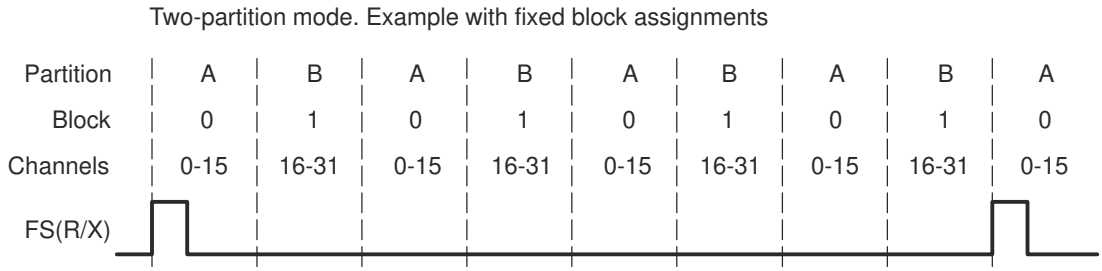


Figure 34-32. Alternating Between the Channels of Partition A and the Channels of Partition B

34.6.4.2 Reassigning Blocks During Reception/Transmission

If you want to use more than 32 channels, you can change which channel blocks are assigned to partitions A and B during the course of a data transfer. However, these changes must be carefully timed. While a partition is being transferred, the associated block assignment bits cannot be modified and the associated channel enable register cannot be modified. For example, if block 3 is being transferred and block 3 is assigned to partition A, you cannot modify (R/X)PABLK to assign different channels to partition A, nor (R/X)CERA to change the channel configuration for partition A.

Several features of the McBSP help you time the reassignment:

- The block of channels currently involved in reception and transmission (the current block) is reflected in the RCBLK/XCBLK bits. Your program can poll these bits to determine which partition is active. When a partition is not active, it is safe to change its block assignment and channel configuration.
- At the end of every block (at the boundary of two partitions), an interrupt can be sent to the CPU. In response to the interrupt, the CPU can then check the RCBLK/XCBLK bits and update the inactive partition. See Section 34.6.8.

Figure 34-33 shows an example of reassigning channels throughout a data transfer. In response to a frame-synchronization pulse, the McBSP alternates between partitions A and B. Whenever partition B is active, the CPU changes the block assignment for partition A. Whenever partition A is active, the CPU changes the block assignment for partition B.

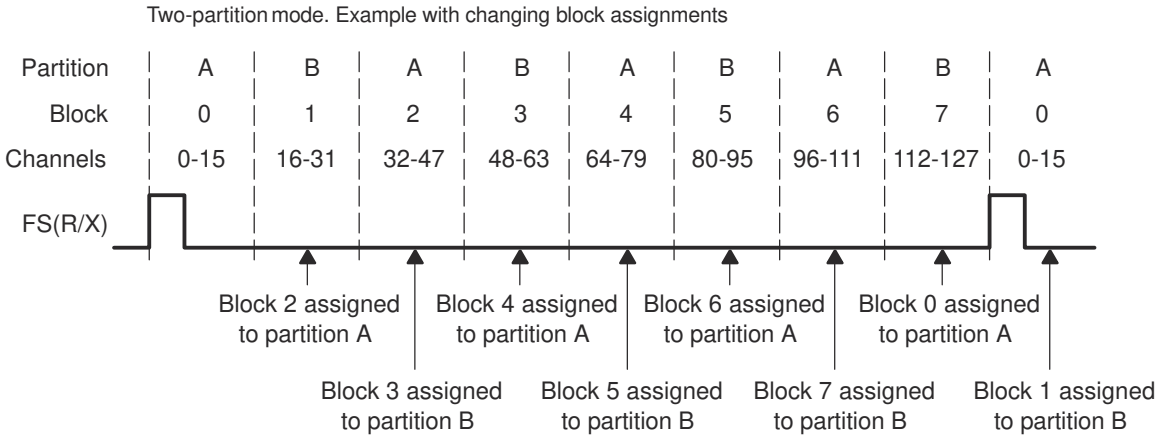


Figure 34-33. Reassigning Channel Blocks Throughout a McBSP Data Transfer

34.6.5 Using Eight Partitions

For multichannel selection operation in the receiver and/or the transmitter, you can use eight partitions or two partitions (described in Section 34.6.4). If you choose the 8-partition mode (RMCME = 1 for reception, XMCME = 1 for transmission), McBSP channels are activated in the following order: A, B, C, D, E, F, G, H. In response to a frame-synchronization pulse, the receiver or transmitter begins with the channels in partition A and then continues with the other partitions in order until the complete frame has been transferred. When the next frame-synchronization pulse occurs, the next frame is transferred, beginning with the channels in partition A.

In the 8-partition mode, the (R/X)PABLK and (R/X)PBBLK bits are ignored and the 16-channel blocks are assigned to the partitions as shown in Table 34-11 and Table 34-12. These assignments cannot be changed. The tables also show the registers used to control the channels in the partitions.

Table 34-11. Receive Channel Assignment and Control With Eight Receive Partitions

Receive Partition	Assigned Block of Receive Channels	Register Used For Channel Control
A	Block 0: channels 0 through 15	RCERA
B	Block 1: channels 16 through 31	RCERB
C	Block 2: channels 32 through 47	RCERC
D	Block 3: channels 48 through 63	RCERD
E	Block 4: channels 64 through 79	RCERE
F	Block 5: channels 80 through 95	RCERF
G	Block 6: channels 96 through 111	RCERG
H	Block 7: channels 112 through 127	RCERH

Table 34-12. Transmit Channel Assignment and Control When Eight Transmit Partitions Are Used

Transmit Partition	Assigned Block of Transmit Channels	Register Used For Channel Control
A	Block 0: channels 0 through 15	XCERA
B	Block 1: channels 16 through 31	XCERB
C	Block 2: channels 32 through 47	XCERC
D	Block 3: channels 48 through 63	XCERD
E	Block 4: channels 64 through 79	XCERE
F	Block 5: channels 80 through 95	XCERF
G	Block 6: channels 96 through 111	XCERG
H	Block 7: channels 112 through 127	XCERH

Figure 34-34 shows an example of the McBSP using the 8-partition mode. In response to a frame-synchronization pulse, the McBSP begins a frame transfer with partition A and then activates B, C, D, E, F, G, and H to complete a 128-word frame.

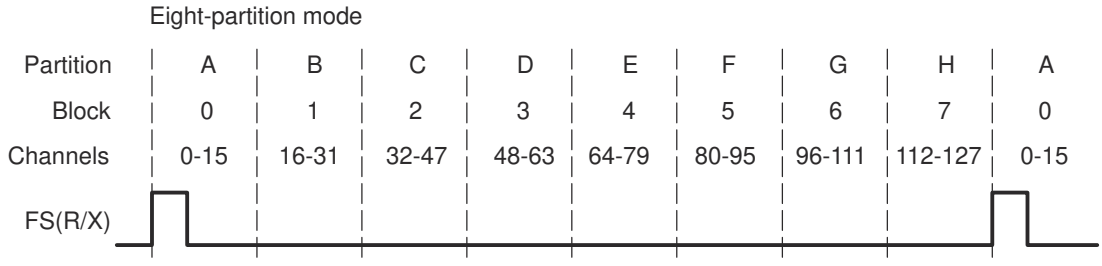


Figure 34-34. McBSP Data Transfer in the 8-Partition Mode

34.6.6 Receive Multichannel Selection Mode

The RMCM bit of MCR1 determines whether all channels or only selected channels are enabled for reception. When RMCM = 0, all 128 receive channels are enabled and cannot be disabled. When RMCM = 1, the receive multichannel selection mode is enabled. In this mode:

- Channels can be individually enabled or disabled. The only channels enabled are those selected in the appropriate receive channel enable registers (RCERs). The way channels are assigned to the RCERs depends on the number of receive channel partitions (2 or 8), as defined by the RMCME bit of MCR1.
- If a receive channel is disabled, any bits received in that channel are passed only as far as the receive buffer registers (RBRs). The receiver does not copy the content of the RBRs to the DRRs, and as a result, does not set the receiver ready bit (RRDY). Therefore, no DMA synchronization event (REVT) is generated and, if the receiver interrupt mode depends on RRDY (RINTM = 00b), no interrupt is generated.

As an example of how the McBSP behaves in the receive multichannel selection mode, suppose you enable only channels 0, 15, and 39 and that the frame length is 40. The McBSP:

1. Accepts bits shifted in from the DR pin in channel 0
2. Ignores bits received in channels 1-14
3. Accepts bits shifted in from the DR pin in channel 15
4. Ignores bits received in channels 16-38
5. Accepts bits shifted in from the DR pin in channel 39

34.6.7 Transmit Multichannel Selection Modes

The XMCM bits of XCR2 determine whether all channels or only selected channels are enabled and unmasked for transmission. More details on enabling and masking are in Section 34.6.7.1. The McBSP has three transmit multichannel selection modes (XMCM = 01b, XMCM = 10b, and XMCM = 11b), which are described in Table 34-13.

Table 34-13. Selecting a Transmit Multichannel Selection Mode With the XMCM Bits

XMCM	Transmit Multichannel Selection Mode
00b	No transmit multichannel selection mode is on. All channels are enabled and unmasked. No channels can be disabled or masked.
01b	All channels are disabled unless the channels are selected in the appropriate transmit channel enable registers (XCERs). If enabled, a channel in this mode is also unmasked. The XMCME bit of MCR2 determines whether 32 channels or 128 channels are selectable in XCERs.
10b	All channels are enabled, but the channels are masked unless the channels are selected in the appropriate transmit channel enable registers (XCERs). The XMCME bit of MCR2 determines whether 32 channels or 128 channels are selectable in XCERs.
11b	This mode is used for symmetric transmission and reception. All channels are disabled for transmission unless the channels are enabled for reception in the appropriate receive channel enable registers (RCERs). Once enabled, the channels are masked unless the channels are also selected in the appropriate transmit channel enable registers (XCERs). The XMCME bit of MCR2 determines whether 32 channels or 128 channels are selectable in RCERs and XCERs.

As an example of how the McBSP behaves in a transmit multichannel selection mode, suppose that XMCM = 01b (all channels disabled unless individually enabled) and that you have enabled only channels 0, 15, and 39. Suppose also that the frame length is 40. The McBSP:

1. Shifts data to the DX pin in channel 0
2. Places the DX pin in the high impedance state in channels 1-14
3. Shifts data to the DX pin in channel 15
4. Places the DX pin in the high impedance state in channels 16-38
5. Shifts data to the DX pin in channel 39

34.6.7.1 Disabling/Enabling Versus Masking/Unmasking

For transmission, a channel can be:

- Enabled and unmasked (transmission can begin and can be completed)
- Enabled but masked (transmission can begin but cannot be completed)
- Disabled (transmission cannot occur)

The following definitions explain the channel control options:

Enabled channel	A channel that can begin transmission by passing data from the data transmit register(s) (DXR(s)) to the transmit shift registers (XSR(s)).
Masked channel	A channel that cannot complete transmission. The DX pin is held in the high impedance state; data cannot be shifted out on the DX pin. In systems where symmetric transmit and receive provides software benefits, this feature allows transmit channels to be disabled on a shared serial bus. A similar feature is not needed for reception because multiple receptions cannot cause serial bus contention.
Disabled channel	A channel that is not enabled. A disabled channel is also masked. Because no DXR-to-XSR copy occurs, the XRDY bit of SPCR2 is not set. Therefore, no DMA synchronization event (XEVT) is generated, and if the transmit interrupt mode depends on XRDY (XINTM = 00b in SPCR2), no interrupt is generated. The XEMPTY bit of SPCR2 is not affected.
Unmasked channel	A channel that is not masked. Data in the XSR(s) is shifted out on the DX pin.

34.6.7.2 Activity on McBSP Pins for Different Values of XMCM

Figure 34-35 shows the activity on the McBSP pins for the various XMCM values. In all cases, the transmit frame is configured as follows:

- XPHASE = 0: Single-phase frame (required for multichannel selection modes)
- XFRLEN1 = 0000011b: 4 words per frame
- XWDLEN1 = 000b: 8 bits per word
- XMCME = 0: 2-partition mode (only partitions A and B used)

In the case where XMCM = 11b, transmission and reception are symmetric, which means the corresponding bits for the receiver (RPHASE, RFRLEN1, RWDLEN1, and RMCME) must have the same values as XPHASE, XFRLEN1, and XWDLEN1, respectively.

In the figure, the arrows showing where the various events occur are only sample indications. Wherever possible, there is a time window in which these events can occur.

34.6.8 Using Interrupts Between Block Transfers

When a multichannel selection mode is used, an interrupt request can be sent to the CPU at the end of every 16-channel block (at the boundary between partitions and at the end of the frame). In the receive multichannel selection mode, a receive interrupt (RINT) request is generated at the end of each block transfer if RINTM = 01b. In any of the transmit multichannel selection modes, a transmit interrupt (XINT) request is generated at the end of each block transfer if XINTM = 01b. When RINTM/XINTM = 01b, no interrupt is generated unless a multichannel selection mode is on.

These interrupt pulses are active high and last for two CPU clock cycles.

This type of interrupt is especially helpful if you are using the two-partition mode (described in Section 34.6.4) and you want to know when you can assign a different block of channels to partition A or B.

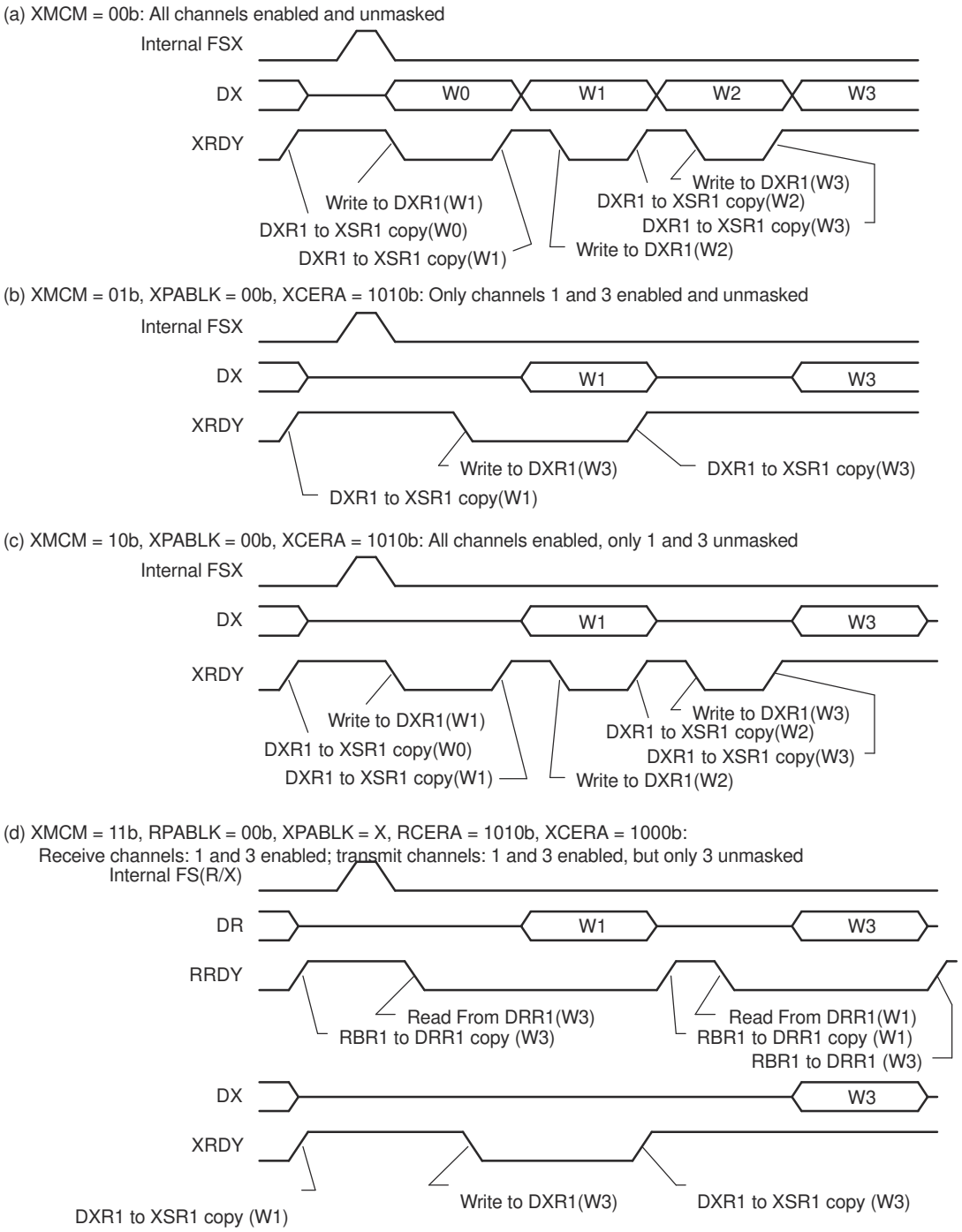


Figure 34-35. Activity on McBSP Pins for the Possible Values of XMCM

34.7 SPI Operation Using the Clock Stop Mode

This chapter explains how to use the McBSP in SPI mode.

34.7.1 SPI Protocol

The SPI protocol is a master-slave configuration with one master device and one or more slave devices. The interface consists of the following four signals:

- Serial data input (also referred to as master in/slave out, or SPISOMI)

- Serial data output (also referred to as master out/slave in, or SPISIMO)
- Shift-clock (also referred to as SPICLK)
- Slave-enable signal (also referred to as $\overline{\text{SPISTE}}$)

A typical SPI interface with a single slave device is shown in Figure 34-36.

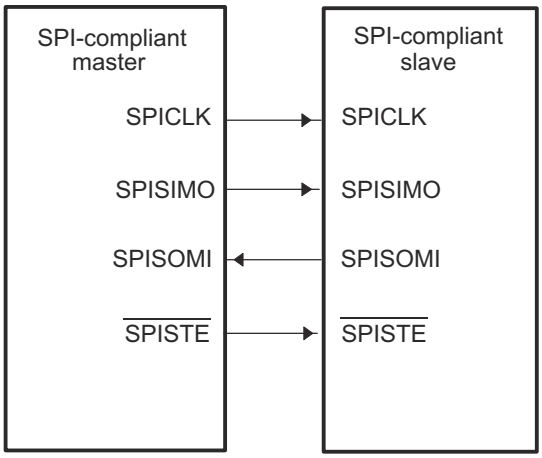


Figure 34-36. Typical SPI Interface

The master device controls the flow of communication by providing shift-clock and slave-enable signals. The slave-enable signal is an optional active-low signal that enables the serial data input and output of the slave device (device not sending out the clock).

In the absence of a dedicated slave-enable signal, communication between the master and slave is determined by the presence or absence of an active shift-clock. When the McBSP is operating in SPI master mode and the $\overline{\text{SPISTE}}$ signal is not used by the SPI slave port, the slave device must remain enabled at all times, and multiple slaves cannot be used.

34.7.2 Clock Stop Mode

The clock stop mode of the McBSP provides compatibility with the SPI protocol. When the McBSP is configured in clock stop mode, the transmitter and receiver are internally synchronized so that the McBSP functions as an SPI master or slave device. The transmit clock signal (CLKX) corresponds to the serial clock signal (SPICLK) of the SPI protocol, while the transmit frame-synchronization signal (FSX) is used as the slave-enable signal ($\overline{\text{SPISTE}}$).

The receive clock signal (MCLKR) and receive frame-synchronization signal (FSR) are not used in the clock stop mode because these signals are internally connected to their transmit counterparts, CLKX and FSX.

34.7.3 Enable and Configure the Clock Stop Mode

The bits required to configure the McBSP as an SPI device are introduced in Table 34-14. Table 34-15 shows how the various combinations of the CLKSTP bit and the polarity bits CLKXP and CLKRP create four possible clock stop mode configurations. The timing diagrams in Section 34.7.4 show the effects of CLKSTP, CLKXP, and CLKRP.

Table 34-14. Bits Used to Enable and Configure the Clock Stop Mode

Bit Field	Description
CLKSTP bits of SPCR1	Use these bits to enable the clock stop mode and to select one of two timing variations. (See also Table 34-15.)
CLKXP bit of PCR	This bit determines the polarity of the CLKX signal. (See also Table 34-15.)
CLKRP bit of PCR	This bit determines the polarity of the MCLKR signal. (See also Table 34-15.)
CLKXM bit of PCR	This bit determines whether CLKX is an input signal (McBSP as slave) or an output signal (McBSP as master).

Table 34-14. Bits Used to Enable and Configure the Clock Stop Mode (continued)

Bit Field	Description
XPHASE bit of XCR2	You must use a single-phase transmit frame (XPHASE = 0).
RPHASE bit of RCR2	You must use a single-phase receive frame (RPHASE = 0).
XFRLEN1 bits of XCR1	You must use a transmit frame length of 1 serial word (XFRLEN1 = 0).
RFRLEN1 bits of RCR1	You must use a receive frame length of 1 serial word (RFRLEN1 = 0).
XWDLEN1 bits of XCR1	The XWDLEN1 bits determine the transmit packet length. XWDLEN1 must be equal to RWDLEN1 because in the clock stop mode. The McBSP transmit and receive circuits are synchronized to a single clock.
RWDLEN1 bits of RCR1	The RWDLEN1 bits determine the receive packet length. RWDLEN1 must be equal to XWDLEN1 because in the clock stop mode. The McBSP transmit and receive circuits are synchronized to a single clock.

Table 34-15. Effects of CLKSTP, CLKXP, and CLKRP on the Clock Scheme

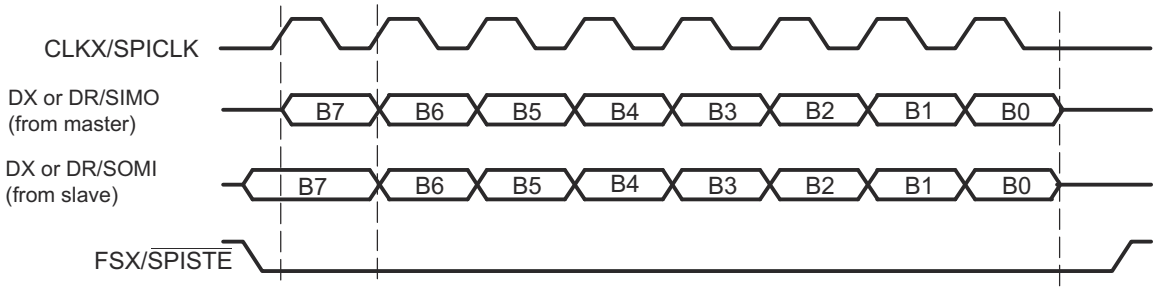
Bit Settings	Clock Scheme
CLKSTP = 00b or 01b CLKXP = 0 or 1 CLKRP = 0 or 1	Clock stop mode disabled. Clock enabled for non-SPI mode.
CLKSTP = 10b CLKXP = 0 CLKRP = 0	Low inactive state without delay: The McBSP transmits data on the rising edge of CLKX and receives data on the falling edge of MCLKR.
CLKSTP = 11b CLKXP = 0 CLKRP = 1	Low inactive state with delay: The McBSP transmits data one-half cycle ahead of the rising edge of CLKX and receives data on the rising edge of MCLKR.
CLKSTP = 10b CLKXP = 1 CLKRP = 0	High inactive state without delay: The McBSP transmits data on the falling edge of CLKX and receives data on the rising edge of MCLKR.
CLKSTP = 11b CLKXP = 1 CLKRP = 1	High inactive state with delay: The McBSP transmits data one-half cycle ahead of the falling edge of CLKX and receives data on the falling edge of MCLKR.

34.7.4 Clock Stop Mode Timing Diagrams

The timing diagrams for the four possible clock stop mode configurations are shown here. Notice that the frame-synchronization signal used in clock stop mode is active throughout the entire transmission as a slave-enable signal. Although the timing diagrams show 8-bit transfers, the packet length can be set to 8, 12, 16, 20, 24, or 32 bits per packet. The receive packet length is selected with the RWDLEN1 bits of RCR1, and the transmit packet length is selected with the XWDLEN1 bits of XCR1. For clock stop mode, the values of RWDLEN1 and XWDLEN1 must be the same because the McBSP transmit and receive circuits are synchronized to a single clock.

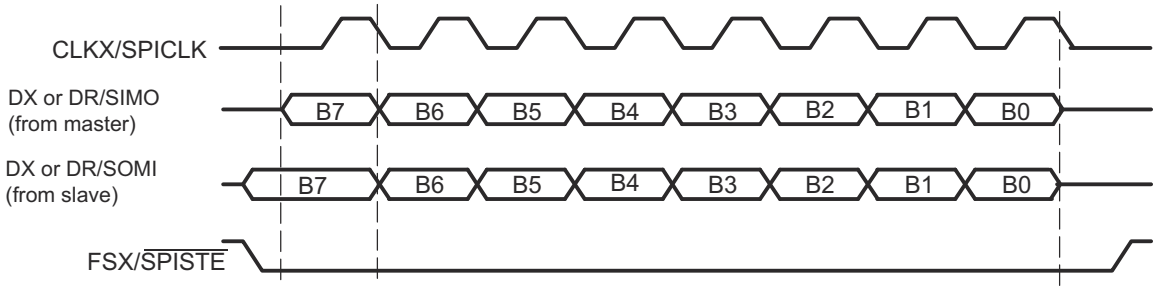
Note

Even if multiple words are consecutively transferred, the CLKX signal is always stopped and the FSX signal returns to the inactive state after a packet transfer. When consecutive packet transfers are performed, this leads to a minimum idle time of two bit-periods between each packet transfer.



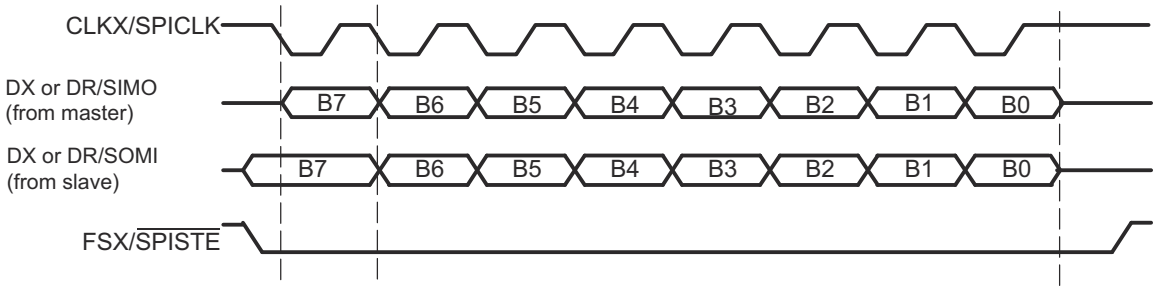
A. If the McBSP is the SPI master (CLKXM = 1), SIMO = DX. If the McBSP is the SPI slave (CLKXM = 0), SIMO = DR.
B. If the McBSP is the SPI master (CLKXM = 1), SOMI = DR. If the McBSP is the SPI slave (CLKXM = 0), SOMI = DX.

Figure 34-37. SPI Transfer with CLKSTP = 10b (No Clock Delay), CLKXP = 0, and CLKRP = 0



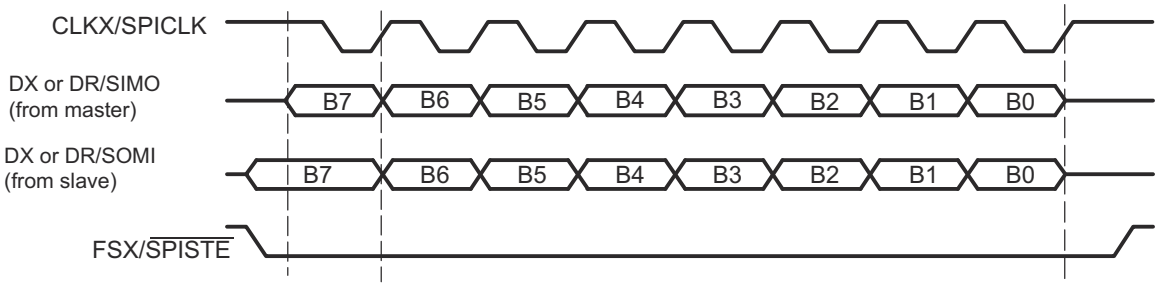
A. If the McBSP is the SPI master (CLKXM = 1), SIMO = DX. If the McBSP is the SPI slave (CLKXM = 0), SIMO = DR.
B. If the McBSP is the SPI master (CLKXM = 1), SOMI = DR. If the McBSP is the SPI slave (CLKXM = 0), SOMI = DX.

Figure 34-38. SPI Transfer with CLKSTP = 11b (Clock Delay), CLKXP = 0, CLKRP = 1



A. If the McBSP is the SPI master (CLKXM = 1), SIMO = DX. If the McBSP is the SPI slave (CLKXM = 0), SIMO = DR.
B. If the McBSP is the SPI master (CLKXM = 1), SOMI = DR. If the McBSP is the SPI slave (CLKXM = 0), SOMI = DX.

Figure 34-39. SPI Transfer with CLKSTP = 10b (No Clock Delay), CLKXP = 1, and CLKRP = 0



A. If the McBSP is the SPI master (CLKXM = 1), SIMO = DX. If the McBSP is the SPI slave (CLKXM = 0), SIMO = DR.
B. If the McBSP is the SPI master (CLKXM = 1), SOMI = DR. If the McBSP is the SPI slave (CLKXM = 0), SOMI = DX.

Figure 34-40. SPI Transfer with CLKSTP = 11b (Clock Delay), CLKXP = 1, CLKRP = 1

34.7.5 Procedure for Configuring a McBSP for SPI Operation

To configure the McBSP for SPI master or slave operation:

1. Place the transmitter and receiver in reset.
   Clear the transmitter reset bit (XRST = 0) in SPCR2 to reset the transmitter. Clear the receiver reset bit (RRST = 0) in SPCR1 to reset the receiver.
2. Place the sample rate generator in reset.
   Clear the sample rate generator reset bit (GRST = 0) in SPCR2 to reset the sample rate generator.
3. Program registers that affect SPI operation.
   Program the appropriate McBSP registers to configure the McBSP for proper operation as an SPI master or an SPI slave. For a list of important bits settings, see one of the following topics:
   - Section 34.7.6
   - Section 34.7.7
4. Enable the sample rate generator.
   To release the sample rate generator from reset, set the sample rate generator reset bit (GRST = 1) in SPCR2.
   Make sure that during the write to SPCR2, you only modify GRST. Otherwise, you modify the McBSP configuration you selected in the previous step.
5. Enable the transmitter and receiver.
   After the sample rate generator is released from reset, wait two sample rate generator clock periods for the McBSP logic to stabilize.
   If the CPU services the McBSP transmit and receive buffers, then you can immediately enable the transmitter (XRST = 1 in SPCR2) and enable the receiver (RRST = 1 in SPCR1).
   If the DMA controller services the McBSP transmit and receive buffers, then you must first configure the DMA controller (this includes enabling the channels that service the McBSP buffers). When the DMA controller is ready, make XRST = 1 and RRST = 1.
   In either case, make sure you only change XRST and RRST when you write to SPCR2 and SPCR1. Otherwise, you modify the bit settings you selected earlier in this procedure.
   After the transmitter and receiver are released from reset, wait two sample rate generator clock periods for the McBSP logic to stabilize.
6. If necessary, enable the frame-synchronization logic of the sample rate generator.
   After the required data acquisition setup is done (DXR[1,2] is loaded with data), set FRST = 1 if an internally generated frame-synchronization pulse is required (that is, if the McBSP is the SPI master).

34.7.6 McBSP as the SPI Master

An SPI interface with the McBSP used as the master is shown in Figure 34-41. When the McBSP is configured as a master, the transmit output signal (DX) is used as the SPISIMO signal of the SPI protocol and the receive input signal (DR) is used as the SPISOMI signal.

The register bit values required to configure the McBSP as a master are listed in Table 34-16. After Table 34-16 are more details about the configuration requirements.

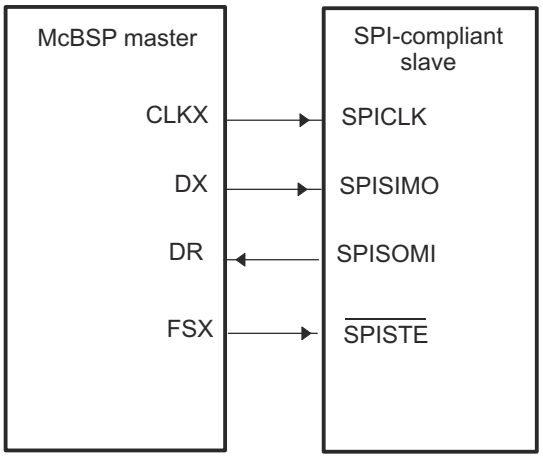


Figure 34-41. SPI Interface with McBSP Used as Master

Table 34-16. Bit Values Required to Configure the McBSP as an SPI Master

Required Bit Setting	Description
CLKSTP = 10b or 11b	The clock stop mode (without or with a clock delay) is selected.
CLKXP = 0 or 1	The polarity of CLKX as seen on the MCLKX pin is positive (CLKXP = 0) or negative (CLKXP = 1).
CLKRP = 0 or 1	The polarity of MCLKR as seen on the MCLKR pin is positive (CLKRP = 0) or negative (CLKRP = 1).
CLKXM = 1	The MCLKX pin is an output pin driven by the internal sample rate generator. Because CLKSTP is equal to 10b or 11b, MCLKR is driven internally by CLKX.
SCLKME = 0	The clock generated by the sample rate generator (CLKG) is derived from the CPU clock.
CLKSM = 1	
CLKGDV is a value from 1 to 255	CLKGDV defines the divide down value for CLKG.
FSXM = 1	The FSX pin is an output pin driven according to the FSGM bit.
FSGM = 0	The transmitter drives a frame-synchronization pulse on the FSX pin every time data is transferred from DXR1 to XSR1.
FSXP = 1	The FSX pin is active low.
XDATDLY = 01b	This setting provides the correct setup time on the FSX signal.
RDATDLY = 01b	

When the McBSP functions as the SPI master, the McBSP controls the transmission of data by producing the serial clock signal. The clock signal on the MCLKX pin is enabled only during packet transfers. When packets are not being transferred, the MCLKX pin remains high or low depending on the polarity used.

For SPI master operation, the MCLKX pin must be configured as an output. The sample rate generator is then used to derive the CLKX signal from the CPU clock. The clock stop mode internally connects the MCLKX pin to the MCLKR signal so that no external signal connection is required on the MCLKR pin and both the transmit and receive circuits are clocked by the master clock (CLKX).

The data delay parameters of the McBSP (XDATDLY and RDATDLY) must be set to 1 for proper SPI master operation. A data delay value of 0 or 2 is undefined in the clock stop mode.

The McBSP can also provide a slave-enable signal ($\overline{\text{SPISTE}}$) on the FSX pin. If a slave-enable signal is required, the FSX pin must be configured as an output and the transmitter must be configured so that a frame-synchronization pulse is generated automatically each time a packet is transmitted (FSGM = 0). The polarity of the FSX pin is programmable high or low; however, in most cases the pin must be configured active low.

When the McBSP is configured as described for SPI master operation, the bit fields for frame-synchronization pulse width (FWID) and frame-synchronization period (FPER) are overridden, and custom frame-synchronization waveforms are not allowed. To see the resulting waveform produced on the FSX pin, see the timing diagrams in Section 34.7.4. The signal becomes active before the first bit of a packet transfer, and remains active until the last bit of the packet is transferred. After the packet transfer is complete, the FSX signal returns to the inactive state.

34.7.7 McBSP as an SPI Slave

An SPI with the McBSP used as a slave is shown in Figure 34-42. When the McBSP is configured as a slave, DX is used as the SPISOMI signal and DR is used as the SPISIMO signal.

The register bit values required to configure the McBSP as a slave are listed in Table 34-17. After Table 34-17 are more details about configuration requirements.

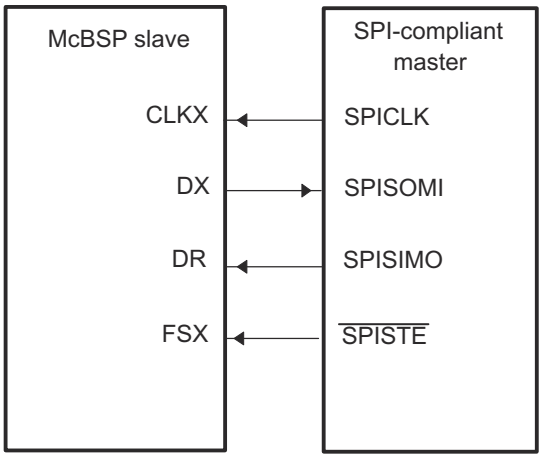


Figure 34-42. SPI Interface With McBSP Used as Slave

Table 34-17. Bit Values Required to Configure the McBSP as an SPI Slave

Required Bit Setting	Description
CLKSTP = 10b or 11b	The clock stop mode (without or with a clock delay) is selected.
CLKXP = 0 or 1	The polarity of CLKX as seen on the MCLKX pin is positive (CLKXP = 0) or negative (CLKXP = 1).
CLKRP = 0 or 1	The polarity of MCLKR as seen on the MCLKR pin is positive (CLKRP = 0) or negative (CLKRP = 1).
CLKXM = 0	The MCLKX pin is an input pin, so that the pin can be driven by the SPI master. Because CLKSTP = 10b or 11b, MCLKR is driven internally by CLKX.
SCLKME = 0 CLKSM = 1	The clock generated by the sample rate generator (CLKG) is derived from the CPU clock. (The sample rate generator is used to synchronize the McBSP logic with the externally-generated master clock.)
CLKGDV = 1	The sample rate generator divides the CPU clock before generating CLKG.
FSXM = 0	The FSX pin is an input pin, so that the pin can be driven by the SPI master.
FSXP = 1	The FSX pin is active low.
XDATDLY = 00b	These bits must be 0s for SPI slave operation.
RDATDLY = 00b	

When the McBSP is used as an SPI slave, the master clock and slave-enable signals are generated externally by a master device. Accordingly, the CLKX and FSX pins must be configured as inputs. The MCLKX pin is internally connected to the MCLKR signal, so that both the transmit and receive circuits of the McBSP are clocked by the external master clock. The FSX pin is also internally connected to the FSR signal, and no external signal connections are required on the MCLKR and FSR pins.

Although the CLKX signal is generated externally by the master and is asynchronous to the McBSP, the sample rate generator of the McBSP must be enabled for proper SPI slave operation. The sample rate generator must

be programmed to the maximum rate of half the CPU clock rate. The internal sample rate clock is then used to synchronize the McBSP logic to the external master clock and slave-enable signals.

The McBSP requires an active edge of the slave-enable signal on the FSX input for each transfer. This means that the master device must assert the slave-enable signal at the beginning of each transfer, and deassert the signal after the completion of each packet transfer; the slave-enable signal cannot remain active between transfers. Unlike the standard SPI, this pin cannot be tied low all the time.

The data delay parameters of the McBSP must be set to 0 for proper SPI slave operation. A value of 1 or 2 is undefined in the clock stop mode.

34.8 Receiver Configuration

To configure the McBSP receiver, perform the following procedure:

1. Place the McBSP/receiver in reset (see Section 34.8.2).
2. Program McBSP registers for the desired receiver operation (see Section 34.8.1).
3. Take the receiver out of reset (see Section 34.8.2).

34.8.1 Programming the McBSP Registers for the Desired Receiver Operation

The following is a list of important tasks to be performed when you are configuring the McBSP receiver. Each task corresponds to one or more McBSP register bit fields.

- Global behavior:
  - Set the receiver pins to operate as McBSP pins.
  - Enable/disable the digital loopback mode.
  - Enable/disable the clock stop mode.
  - Enable/disable the receive multichannel selection mode.
- Data behavior:
  - Choose 1 or 2 phases for the receive frame.
  - Set the receive word length(s).
  - Set the receive frame length.
  - Enable/disable the receive frame-synchronization ignore function.
  - Set the receive companding mode.
  - Set the receive data delay.
  - Set the receive sign-extension and justification mode.
  - Set the receive interrupt mode.
- Frame-synchronization behavior:
  - Set the receive frame-synchronization mode.
  - Set the receive frame-synchronization polarity.
  - Set the sample rate generator (SRG) frame-synchronization period and pulse width.
- Clock behavior:
  - Set the receive clock mode.
  - Set the receive clock polarity.
  - Set the SRG clock divide-down value.
  - Set the SRG clock synchronization mode.
  - Set the SRG clock mode (choose an input clock).
  - Set the SRG input clock polarity.

34.8.2 Resetting and Enabling the Receiver

The first step of the receiver configuration procedure is to reset the receiver, and the last step is to enable the receiver (to take the receiver out of reset). Table 34-18 describes the bits used for both of these steps.

Table 34-18. Register Bits Used to Reset or Enable the McBSP Receiver Field Descriptions

Register	Bit	Field	Value	Description
SPCR2	7	FRST		Frame-synchronization logic reset
			0	Frame-synchronization logic is reset. The sample rate generator does not generate frame-synchronization signal FSG, even if GRST = 1.
			1	If GRST = 1, frame-synchronization signal FSG is generated after (FPER + 1) number of CLKG clock cycles; all frame counters are loaded with their programmed values.
SPCR2	6	GRST		Sample rate generator reset
			0	Sample rate generator is reset. If GRST = 0 due to a DSP reset, CLKG is driven by the CPU clock divided by 2, and FSG is driven low (inactive). If GRST = 0 due to program code, CLKG and FSG are both driven low (inactive).
			1	Sample rate generator is enabled. CLKG is driven according to the configuration programmed in the sample rate generator registers (SRGR[1,2]). If FRST = 1, the generator also generates the frame-synchronization signal FSG as programmed in the sample rate generator registers.
SPCR1	0	RRST		Receiver reset
			0	The serial port receiver is disabled and in the reset state.
			1	The serial port receiver is enabled.

34.8.2.1 Reset Considerations

The serial port can be reset in the following two ways:

1. The DSP reset ($\overline{\text{XRS}}$ signal driven low) places the receiver, transmitter, and sample rate generator in reset. When the device reset is removed ($\overline{\text{XRS}}$ signal released), GRST = FRST = RRST = XRST = 0 keep the entire serial port in the reset state, provided the McBSP clock is turned on.
2. The serial port transmitter and receiver can be reset directly using the RRST and XRST bits in the serial port control registers. The sample rate generator can be reset directly using the GRST bit in SPCR2.

Table 34-19 shows the state of McBSP pins when the serial port is reset due to a device reset and a direct receiver/transmitter reset.

For more details about McBSP reset conditions and effects, see Section 34.10.2.

Table 34-19. Reset State of Each McBSP Pin

Pin	Possible States(1)	State Forced By Device Reset	State Forced By Receiver Reset (RRST = 0 and GRST = 1)
MDRx	I	GPIO Input	Input
MCLKRx	I/O/Z	GPIO Input	Known state, if input; MCLKR running, if output
MFSRx	I/O/Z	GPIO Input	Known state, if input; FSRP inactive state, if output
			Transmitter reset (XRST = 0 and GRST = 1)
MDXx	O/Z	GPIO Input	Low impedance after transmit bit clock provided
MCLKXx	I/O/Z	GPIO Input	Known state, if input; CLKX running, if output
MFSXx	I/O/Z	GPIO Input	Known state, if input; FSXP inactive state, if output

(1) In Possible States column, I = input, O = output, Z = high impedance

34.8.3 Set the Receiver Pins to Operate as McBSP Pins

To configure a pin for McBSP functioning, configure the bits of the GPxMUXn register appropriately. In addition to this, bits 12 and 13 of the PCR register must be set to 0. These bits are defined as reserved.

34.8.4 Digital Loopback Mode

The DLB bit determines whether the digital loopback mode is on. DLB is described in Table 34-20.

Table 34-20. Register Bit Used to Enable/Disable the Digital Loopback Mode

Register	Bit	Name	Function		Type	Reset Value
SPCR1	15	DLB	Digital loopback mode		R/W	0
			DLB = 0	Digital loopback mode is disabled.		
			DLB = 1	Digital loopback mode is enabled.		

In the digital loopback mode, the receive signals are connected internally through multiplexers to the corresponding transmit signals, as shown in Table 34-21. This mode allows testing of serial port code with a single DSP device; the McBSP receives the data it transmits.

Table 34-21. Receive Signals Connected to Transmit Signals in Digital Loopback Mode

This Receive Signal	Is Fed Internally by This Transmit Signal
MDR (receive data)	MDX (transmit data)
MFSR (receive frame synchronization)	MFSX (transmit frame synchronization)
MCLKR (receive clock)	MCLKX (transmit clock)

34.8.5 Clock Stop Mode

The CLKSTP bits determine whether the clock stop mode is on. CLKSTP is described in Table 34-22.

Table 34-22. Register Bits Used to Enable/Disable the Clock Stop Mode

Register	Bit	Name	Function		Type	Reset Value
SPCR1	12-11	CLKSTP	Clock stop mode		R/W	00
			CLKSTP = 0Xb	Clock stop mode disabled; normal clocking for non-SPI mode		
			CLKSTP = 10b	Clock stop mode enabled, without clock delay		
			CLKSTP = 11b	Clock stop mode enabled, with clock delay		

The clock stop mode supports the SPI master-slave protocol. If you do not plan to use the SPI protocol, you can clear CLKSTP to disable the clock stop mode.

In the clock stop mode, the clock stops at the end of each data transfer. At the beginning of each data transfer, the clock starts immediately (CLKSTP = 10b) or after a half-cycle delay (CLKSTP = 11b). The CLKXP bit determines whether the starting edge of the clock on the MCLKX pin is rising or falling. The CLKRP bit determines whether receive data is sampled on the rising or falling edge of the clock shown on the MCLKR pin.

Table 34-23 summarizes the impact of CLKSTP, CLKXP, and CLKRP on serial port operation. In the clock stop mode, the receive clock is tied internally to the transmit clock, and the receive frame-synchronization signal is tied internally to the transmit frame-synchronization signal.

Table 34-23. Effects of CLKSTP, CLKXP, and CLKRP on the Clock Scheme

Bit Settings	Clock Scheme
CLKSTP = 00b or 01b CLKXP = 0 or 1 CLKRP = 0 or 1	Clock stop mode disabled. Clock enabled for non-SPI mode.
CLKSTP = 10b CLKXP = 0 CLKRP = 0	Low inactive state without delay: The McBSP transmits data on the rising edge of CLKX and receives data on the falling edge of MCLKR.
CLKSTP = 11b CLKXP = 0 CLKRP = 1	Low inactive state with delay: The McBSP transmits data one-half cycle ahead of the rising edge of CLKX and receives data on the rising edge of MCLKR.
CLKSTP = 10b CLKXP = 1 CLKRP = 0	High inactive state without delay: The McBSP transmits data on the falling edge of CLKX and receives data on the rising edge of MCLKR.
CLKSTP = 11b CLKXP = 1 CLKRP = 1	High inactive state with delay: The McBSP transmits data one-half cycle ahead of the falling edge of CLKX and receives data on the falling edge of MCLKR.

34.8.6 Receive Multichannel Selection Mode

The RMCM bit determines whether the receive multichannel selection mode is on. RMCM is described in Table 34-24. For more details, see Section 34.6.6.

Table 34-24. Register Bit Used to Enable/Disable the Receive Multichannel Selection Mode

Register	Bit	Name	Function		Type	Reset Value
MCR1	0	RMCM	Receive multichannel selection mode		R/W	0
			RMCM = 0	The mode is disabled. All 128 channels are enabled.		
			RMCM = 1	The mode is enabled. Channels can be individually enabled or disabled. The only channels enabled are those selected in the appropriate receive channel enable registers (RCERs). The way channels are assigned to the RCERs depends on the number of receive channel partitions (2 or 8), as defined by the RMCME bit.		

34.8.7 Receive Frame Phases

The RPHASE bit (see Table 34-25) determines whether the receive data frame has one or two phases.

Table 34-25. Register Bit Used to Choose One or Two Phases for the Receive Frame

Register	Bit	Name	Function		Type	Reset Value
RCR2	15	RPHASE	Receive phase number		R/W	0
			Specifies whether the receive frame has 1 or 2 phases.			
			RPHASE = 0	Single-phase frame		
			RPHASE = 1	Dual-phase frame		

34.8.8 Receive Word Lengths

The RWDLEN1 and RWDLEN2 bit fields (see Table 34-26) determine how many bits are in each serial word in phase 1 and in phase 2, respectively, of the receive data frame.

Table 34-26. Register Bits Used to Set the Receive Word Lengths

Register	Bit	Name	Function		Type	Reset Value
RCR1	7-5	RWDLEN1	Receive word length 1		R/W	000
			Specifies the length of every serial word in phase 1 of the receive frame.			
			RWDLEN1 = 000	8 bits		
			RWDLEN1 = 001	12 bits		
			RWDLEN1 = 010	16 bits		
			RWDLEN1 = 011	20 bits		
			RWDLEN1 = 100	24 bits		
			RWDLEN1 = 101	32 bits		
			RWDLEN1 = 11X	Reserved		
RCR2	7-5	RWDLEN2	Receive word length 2		R/W	000
			If a dual-phase frame is selected, RWDLEN2 specifies the length of every serial word in phase 2 of the frame.			
			RWDLEN2 = 000	8 bits		
			RWDLEN2 = 001	12 bits		
			RWDLEN2 = 010	16 bits		
			RWDLEN2 = 011	20 bits		
			RWDLEN2 = 100	24 bits		
			RWDLEN2 = 101	32 bits		
			RWDLEN2 = 11X	Reserved		

34.8.8.1 Word Length Bits

Each frame can have one or two phases, depending on the value that you load into the RPHASE bit. If a single-phase frame is selected, RWDLEN1 selects the length for every serial word received in the frame. If a dual-phase frame is selected, RWDLEN1 determines the length of the serial words in phase 1 of the frame and RWDLEN2 determines the word length in phase 2 of the frame.

34.8.9 Receive Frame Length

The RFRLEN1 and RFRLEN2 bit fields (see Table 34-27) determine how many serial words are in phase 1 and in phase 2, respectively, of the receive data frame.

Table 34-27. Register Bits Used to Set the Receive Frame Length

Register	Bit	Name	Function		Type	Reset Value
RCR1	14-8	RFRLEN1	Receive frame length 1		R/W	000 0000
			(RFRLEN1 + 1) is the number of serial words in phase 1 of the receive frame.			
			RFRLEN1 = 000 0000	1 word in phase 1		
			RFRLEN1 = 000 0001	2 words in phase 1		
			|	|		
			|	|		
			RFRLEN1 = 111 1111	128 words in phase 1		
RCR2	14-8	RFRLEN2	Receive frame length 2		R/W	000 0000
			If a dual-phase frame is selected, (RFRLEN2 + 1) is the number of serial words in phase 2 of the receive frame.			
			RFRLEN2 = 000 0000	1 word in phase 2		
			RFRLEN2 = 000 0001	2 words in phase 2		
			|	|		
			|	|		
			RFRLEN2 = 111 1111	128 words in phase 2		

34.8.9.1 Selected Frame Length

The receive frame length is the number of serial words in the receive frame. Each frame can have one or two phases, depending on value that you load into the RPHASE bit.

If a single-phase frame is selected (RPHASE = 0), the frame length is equal to the length of phase 1. If a dual-phase frame is selected (RPHASE = 1), the frame length is the length of phase 1 plus the length of phase 2.

The 7-bit RFRLEN fields allow up to 128 words per phase. See Table 34-28 for a summary of how to calculate the frame length. This length corresponds to the number of words or logical time slots or channels per frame-synchronization pulse.

Program the RFRLEN fields with [*w minus 1*], where *w* represents the number of words per phase. For the example, if you want a phase length of 128 words in phase 1, load 127 into RFRLEN1.

Table 34-28. How to Calculate the Length of the Receive Frame

RPHASE	RFRLEN1	RFRLEN2	Frame Length
0	0 ≤ RFRLEN1 ≤ 127	Don't care	(RFRLEN1 + 1) words
1	0 ≤ RFRLEN1 ≤ 127	0 ≤ RFRLEN2 ≤ 127	(RFRLEN1 + 1) + (RFRLEN2 + 1) words

34.8.10 Receive Frame-Synchronization Ignore Function

The RFIG bit (see Table 34-29) controls the receive frame-synchronization ignore function.

Table 34-29. Register Bit Used to Enable/Disable the Receive Frame-Synchronization Ignore Function

Register	Bit	Name	Function		Type	Reset Value
RCR2	2	RFIG	Receive frame-synchronization ignore		R/W	0
			RFIG = 0	An unexpected receive frame-synchronization pulse causes the McBSP to restart the frame transfer.		
			RFIG = 1	The McBSP ignores unexpected receive frame-synchronization pulses.		

34.8.10.1 Unexpected Frame-Synchronization Pulses and the Frame-Synchronization Ignore Function

If a frame-synchronization pulse starts the transfer of a new frame before the current frame is fully received, this pulse is treated as an unexpected frame-synchronization pulse.

When RFIG = 1, reception continues, ignoring the unexpected frame-synchronization pulses.

When RFIG = 0, an unexpected FSR pulse causes the McBSP to discard the contents of RSR[1,2] in favor of the new incoming data. Therefore, if RFIG = 0 and an unexpected frame-synchronization pulse occurs, the serial port:

1. Aborts the current data transfer
2. Sets RSYNCERR in SPCR1 to 1
3. Begins the transfer of a new data word

For more details about the frame-synchronization error condition, see Section 34.5.3.

34.8.10.2 Examples of Effects of RFIG

Figure 34-43 shows an example in which word B is interrupted by an unexpected frame-synchronization pulse when (R/X)FIG = 0. In the case of reception, the reception of B is aborted (B is lost), and a new data word C in this example) is received after the appropriate data delay. This condition is a receive synchronization error, which sets the RSYNCERR bit.

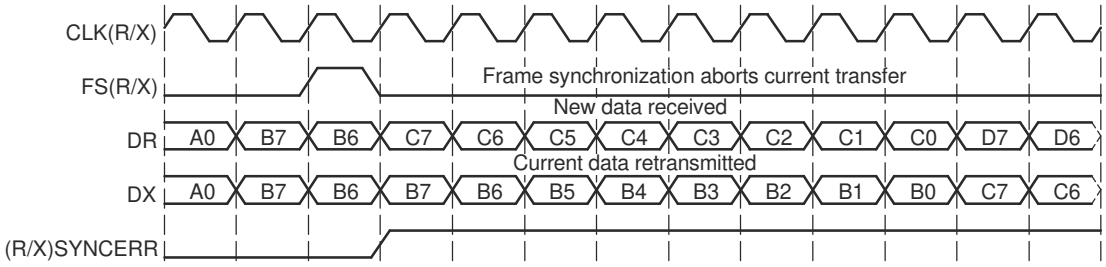


Figure 34-43. Unexpected Frame-Synchronization Pulse With (R/X)FIG = 0

In contrast with Figure 34-43, Figure 34-44 shows McBSP operation when unexpected frame-synchronization signals are ignored (when (R/X)FIG = 1). Here, the transfer of word B is not affected by an unexpected pulse.

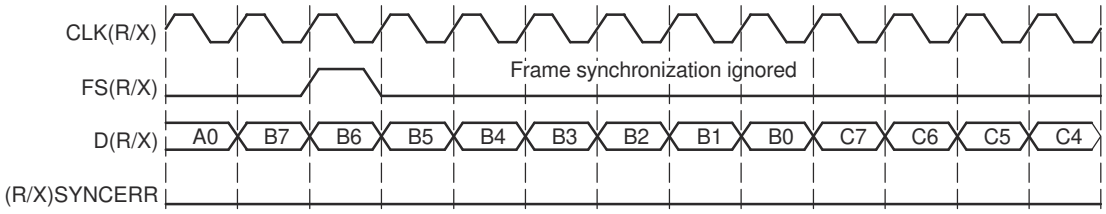


Figure 34-44. Unexpected Frame-Synchronization Pulse With (R/X)FIG = 1

34.8.11 Receive Companding Mode

The RCOMPAND bits (see Table 34-30) determine whether companding or another data transfer option is chosen for McBSP reception.

Table 34-30. Register Bits Used to Set the Receive Companding Mode

Register	Bit	Name	Function		Type	Reset Value
RCR2	4-3	RCOMPAND	Receive companding mode		R/W	00
			Modes other than 00b are enabled only when the appropriate RWDLEN is 000b, indicating 8-bit data.			
			RCOMPAND = 00	No companding, any size data, MSB received first		
			RCOMPAND = 01	No companding, 8-bit data, LSB received first (for details, see Section 34.8.11.4).		
			RCOMPAND = 10	µ-law companding, 8-bit data, MSB received first		
			RCOMPAND = 11	A-law companding, 8-bit data, MSB received first		

34.8.11.1 Companding

Companding (COMpressing and exPANDing) hardware allows compression and expansion of data in either μ-law or A-law format. The companding standard employed in the United States and Japan is μ-law. The European companding standard is referred to as A-law. The specifications for μ-law and A-law log PCM are part of the CCITT G.711 recommendation.

A-law and μ-law allow 13 bits and 14 bits of dynamic range, respectively. Any values outside this range are set to the most positive or most negative value. Thus, for companding to work best, the data transferred to and from the McBSP via the CPU or DMA controller must be at least 16 bits wide.

The μ-law and A-law formats both encode data into 8-bit code words. Companded data is always 8 bits wide; the appropriate word length bits (RWDLEN1, RWDLEN2, XWDLEN1, XWDLEN2) must therefore be set to 0, indicating an 8-bit wide serial data stream. If companding is enabled and either of the frame phases does not have an 8-bit word length, companding continues as if the word length is 8 bits.

Figure 34-45 illustrates the companding processes. When companding is chosen for the transmitter, compression occurs during the process of copying data from DXR1 to XSR1. The transmit data is encoded according to the specified companding law (A-law or μ-law). When companding is chosen for the receiver, expansion occurs during the process of copying data from RBR1 to DRR1. The receive data is decoded to 2's-complement format.

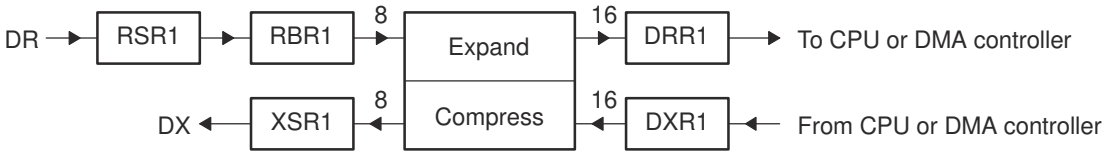


Figure 34-45. Companding Processes for Reception and for Transmission

34.8.11.2 Format of Expanded Data

For reception, the 8-bit compressed data in RBR1 is expanded to left-justified 16-bit data in DRR1. The RJUST bit of SPCR1 is ignored when companding is used.

34.8.11.3 Companding Internal Data

If the McBSP is otherwise unused (the serial port transmit and receive sections are reset), the companding hardware can compand internal data. See Section 34.3.2.2.

34.8.11.4 Option to Receive LSB First

Normally, the McBSP transmits or receives all data with the most significant bit (MSB) first. However, certain 8-bit data protocols (that do not use companded data) require the least significant bit (LSB) to be transferred first. If you set RCOMPAND = 01b in RCR2, the bit ordering of 8-bit words is reversed during reception. Similar to companding, this feature is enabled only if the appropriate word length bits are set to 0, indicating that 8-bit words are to be transferred serially. If either phase of the frame does not have an 8-bit word length, the McBSP assumes the word length is eight bits and LSB-first ordering is done.

34.8.12 Receive Data Delay

The RDATDLY bits (see Table 34-31) determine the length of the data delay for the receive frame.

Table 34-31. Register Bits Used to Set the Receive Data Delay

Register	Bit	Name	Function		Type	Reset Value
RCR2	1-0	RDATDLY	Receive data delay		R/W	00
			RDATDLY = 00	0-bit data delay		
			RDATDLY = 01	1-bit data delay		
			RDATDLY = 10	2-bit data delay		
			RDATDLY = 11	Reserved		

34.8.12.1 Data Delay

The start of a frame is defined by the first clock cycle in which frame synchronization is found to be active. The beginning of actual data reception or transmission with respect to the start of the frame can be delayed if required. This delay is called data delay.

RDATDLY specifies the data delay for reception. The range of programmable data delay is zero to two bit-clocks (RDATDLY = 00b-10b), as described in Table 34-31 and shown in Figure 34-46. In this figure, the data transferred is an 8-bit value with bits labeled B7, B6, B5, and so on. Typically a 1-bit delay is selected, because data often follows a 1-cycle active frame-synchronization pulse.

34.8.12.2 0-Bit Data Delay

Normally, a frame-synchronization pulse is detected or sampled with respect to an edge of internal serial clock CLK(R/X). Thus, on the following cycle or later (depending on the data delay value), data may be received or transmitted. However, in the case of 0-bit data delay, the data must be ready for reception and/or transmission on the same serial clock cycle.

For reception, this problem is solved because receive data is sampled on the first falling edge of MCLKR where an active-high internal FSR is detected. However, data transmission must begin on the rising edge of the internal CLKX clock that generated the frame synchronization. Therefore, the first data bit is assumed to be present in XSR1, and thus on DX. The transmitter then asynchronously detects the frame-synchronization signal (FSX) going active high and immediately starts driving the first bit to be transmitted on the DX pin.

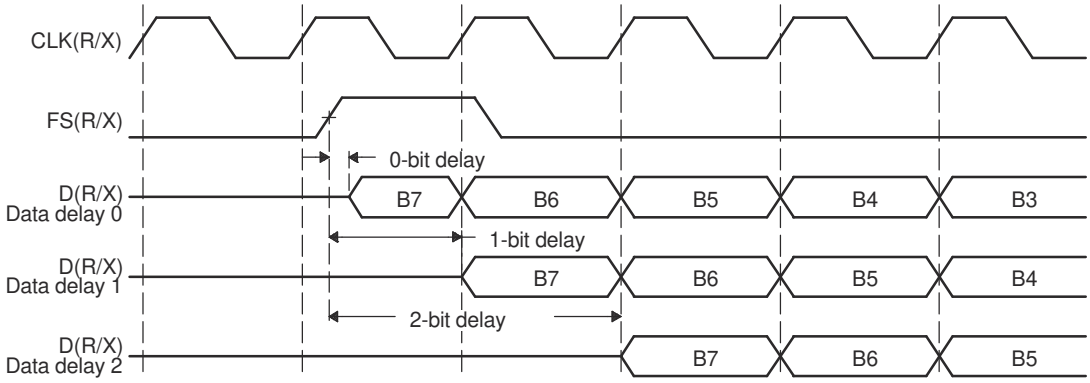


Figure 34-46. Range of Programmable Data Delay

34.8.12.3 2-Bit Data Delay

A data delay of two bit periods allows the serial port to interface to different types of T1 framing devices where the data stream is preceded by a framing bit. During reception of such a stream with data delay of two bits (framing bit appears after a 1-bit delay and data appears after a 2-bit delay), the serial port essentially discards the framing bit from the data stream, as shown in Figure 34-47. In this figure, the data transferred is an 8-bit value with bits labeled B7, B6, B5, and so on.

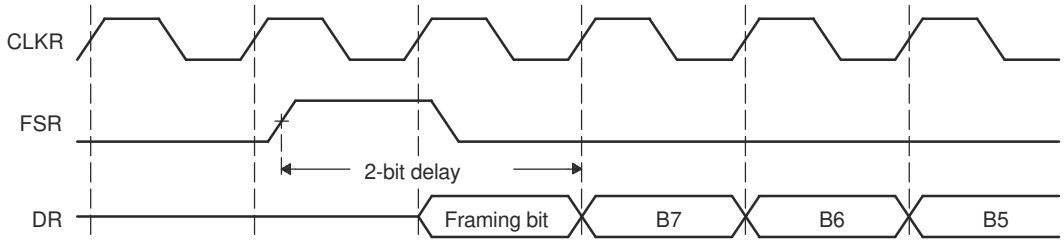


Figure 34-47. 2-Bit Data Delay Used to Skip a Framing Bit

34.8.13 Receive Sign-Extension and Justification Mode

The RJUST bits (see Table 34-32) determine whether data received by the McBSP is sign-extended and how the data is justified.

Table 34-32. Register Bits Used to Set the Receive Sign-Extension and Justification Mode

Register	Bit	Name	Function		Type	Reset Value
SPCR1	14-13	RJUST	Receive sign-extension and justification mode		R/W	00
			RJUST = 00	Right justify data and zero fill MSBs in DRR[1,2]		
			RJUST = 01	Right justify data and sign extend the data into the MSBs in DRR[1,2]		
			RJUST = 10	Left justify data and zero fill LSBs in DRR[1,2]		
			RJUST = 11	Reserved		

34.8.13.1 Sign-Extension and the Justification

RJUST in SPCR1 selects whether data in RBR[1,2] is right- or left-justified (with respect to the MSB) in DRR[1,2] and whether unused bits in DRR[1,2] are filled with zeros or with sign bits.

Table 34-33 and Table 34-34 show the effects of various RJUST values. The first table shows the effect on an example 12-bit receive-data value ABCh. The second table shows the effect on an example 20-bit receive-data value ABCDEh.

Table 34-33. Example: Use of RJUST Field With 12-Bit Data Value ABCh

RJUST	Justification	Extension	Value in DRR2	Value in DRR1
00b	Right	Zero fill MSBs	0000h	0ABCh
01b	Right	Sign extend data into MSBs	FFFFh	FABCh
10b	Left	Zero fill LSBs	0000h	ABC0h
11b	Reserved	Reserved	Reserved	Reserved

Table 34-34. Example: Use of RJUST Field With 20-Bit Data Value ABCDEh

RJUST	Justification	Extension	Value in DRR2	Value in DRR1
00b	Right	Zero fill MSBs	000Ah	BCDEh
01b	Right	Sign extend data into MSBs	FFFAh	BCDEh
10b	Left	Zero fill LSBs	ABCDh	E000h
11b	Reserved	Reserved	Reserved	Reserved

34.8.14 Receive Interrupt Mode

The RINTM bits (see Table 34-35) determine which event generates a receive interrupt request to the CPU.

The receive interrupt (RINT) informs the CPU of changes to the serial port status. Four options exist for configuring this interrupt. The options are set by the receive interrupt mode bits, RINTM, in SPCR1.

Table 34-35. Register Bits Used to Set the Receive Interrupt Mode

Register	Bit	Name	Function		Type	Reset Value
SPCR1	5-4	RINTM	Receive interrupt mode		R/W	00
			RINTM = 00	RINT generated when RRDY changes from 0 to 1. Interrupt on every serial word by tracking the RRDY bit in SPCR1. Regardless of the value of RINTM, RRDY can be read to detect the RRDY = 1 condition.		
			RINTM = 01	RINT generated by an end-of-block or end-of-frame condition in the receive multichannel selection mode. In the multichannel selection mode, interrupt after every 16-channel block boundary has been crossed within a frame and at the end of the frame. For details, see Section 34.6.8. In any other serial transfer case, this setting is not applicable and, therefore, no interrupts are generated.		
			RINTM = 10	RINT generated by a new receive frame-synchronization pulse. Interrupt on detection of receive frame-synchronization pulses. This generates an interrupt even when the receiver is in the reset state. This is done by synchronizing the incoming frame-synchronization pulse to the CPU clock and sending the pulse to the CPU using RINT.		
			RINTM = 11	RINT generated when RSYNCERR is set. Interrupt on frame-synchronization error. Regardless of the value of RINTM, RSYNCERR can be read to detect this condition. For information on using RSYNCERR, see Section 34.5.3.		

34.8.15 Receive Frame-Synchronization Mode

The bits described in Table 34-36 determine the source for receive frame synchronization and the function of the FSR pin.

34.8.15.1 Receive Frame-Synchronization Modes

Table 34-37 shows how you can select various sources to provide the receive frame-synchronization signal and the effect on the FSR pin. The polarity of the signal on the FSR pin is determined by the FSRP bit.

In digital loopback mode (DLB = 1), the transmit frame-synchronization signal is used as the receive frame-synchronization signal.

Also in the clock stop mode, the internal receive clock signal (MCLKR) and the internal receive frame-synchronization signal (FSR) are internally connected to their transmit counterparts, CLKX and FSX.

Table 34-36. Register Bits Used to Set the Receive Frame Synchronization Mode

Register	Bit	Name	Function		Type	Reset Value
PCR	10	FSRM	Receive frame-synchronization mode		R/W	0
			FSRM = 0	Receive frame synchronization is supplied by an external source via the FSR pin.		
			FSRM = 1	Receive frame synchronization is supplied by the sample rate generator. FSR is an output pin reflecting internal FSR, except when GSYNC = 1 in SRGR2.		
SRGR2	15	GSYNC	Sample rate generator clock synchronization mode		R/W	0
			If the sample rate generator creates a frame-synchronization signal (FSG) that is derived from an external input clock, the GSYNC bit determines whether FSG is kept synchronized with pulses on the FSR pin.			
			GSYNC = 0	No clock synchronization is used: CLKG oscillates without adjustment, and FSG pulses every (FPER + 1) CLKG cycles.		
			GSYNC = 1	Clock synchronization is used. When a pulse is detected on the FSR pin: • CLKG is adjusted as necessary so that CLKG is synchronized with the input clock on the MCLKR pin. • FSG pulses FSG only pulses in response to a pulse on the FSR pin. The frame-synchronization period defined in FPER is ignored. For more details, see Section 34.4.3.		
SPCR1	15	DLB	Digital loopback mode		R/W	0
			DLB = 0	Digital loopback mode is disabled.		
			DLB = 1	Digital loopback mode is enabled. The receive signals, including the receive frame-synchronization signal, are connected internally through multiplexers to the corresponding transmit signals.		
SPCR1	12-11	CLKSTP	Clock stop mode		R/W	00
			CLKSTP = 0Xb	Clock stop mode disabled; normal clocking for non-SPI mode.		
			CLKSTP = 10b	Clock stop mode enabled without clock delay. The internal receive clock signal (MCLKR) and the internal receive frame-synchronization signal (FSR) are internally connected to their transmit counterparts, CLKX and FSX.		
			CLKSTP = 11b	Clock stop mode enabled with clock delay. The internal receive clock signal (MCLKR) and the internal receive frame-synchronization signal (FSR) are internally connected to their transmit counterparts, CLKX and FSX.		

Table 34-37. Select Sources to Provide the Receive Frame-Synchronization Signal and the Effect on the FSR Pin

DLB	FSRM	GSYNC	Source of Receive Frame Synchronization	FSR Pin Status
0	0	0 or 1	An external frame-synchronization signal enters the McBSP through the FSR pin. The signal is then inverted as determined by FSRP before being used as internal FSR.	Input
0	1	0	Internal FSR is driven by the sample rate generator frame-synchronization signal (FSG).	Output. FSG is inverted as determined by FSRP before being driven out on the FSR pin.
0	1	1	Internal FSR is driven by the sample rate generator frame-synchronization signal (FSG).	Input. The external frame-synchronization input on the FSR pin is used to synchronize CLKG and generate FSG pulses.
1	0	0	Internal FSX drives internal FSR.	High impedance
1	0 or 1	1	Internal FSX drives internal FSR.	Input. If the sample rate generator is running, external FSR is used to synchronize CLKG and generate FSG pulses.
1	1	0	Internal FSX drives internal FSR.	Output. Receive (same as transmit) frame synchronization is inverted as determined by FSRP before being driven out on the FSR pin.

34.8.16 Receive Frame-Synchronization Polarity

The FSRP bit (see Table 34-38) determines whether frame-synchronization pulses are active high or active low on the FSR pin.

Table 34-38. Register Bit Used to Set Receive Frame-Synchronization Polarity

Register	Bit	Name	Function		Type	Reset Value
PCR	2	FSRP	Receive frame-synchronization polarity		R/W	0
			FSRP = 0	Frame-synchronization pulse FSR is active high.		
			FSRP = 1	Frame-synchronization pulse FSR is active low.		

34.8.16.1 Frame-Synchronization Pulses, Clock Signals, and Their Polarities

Receive frame-synchronization pulses can be generated internally by the sample rate generator (see Section 34.4.2) or driven by an external source. The source of frame synchronization is selected by programming the mode bit, FSRM, in PCR. FSR is also affected by the GSYNC bit in SRGR2. For information about the effects of FSRM and GSYNC, see Section 34.8.15. Similarly, receive clocks can be selected to be inputs or outputs by programming the mode bit, CLKRM, in the PCR (see Section 34.8.17).

When FSR and FSX are inputs (FSXM = FSRM= 0, external frame-synchronization pulses), the McBSP detects them on the internal falling edge of clock, internal MCLKR, and internal CLKX, respectively. The receive data arriving at the DR pin is also sampled on the falling edge of internal MCLKR. These internal clock signals are either derived from an external source via CLK(R/X) pins or driven by the sample rate generator clock (CLKG) internal to the McBSP.

When FSR and FSX are outputs, implying that the outputs are driven by the sample rate generator, the outputs are generated (transition to their active state) on the rising edge of the internal clock, CLK(R/X). Similarly, data on the DX pin is output on the rising edge of internal CLKX.

FSRP, FSXP, CLKRP, and CLKXP in the pin control register (PCR) configure the polarities of the FSR, FSX, MCLKR, and CLKX signals, respectively. All frame-synchronization signals (internal FSR, internal FSX) that are internal to the serial port are active high. If the serial port is configured for external frame synchronization (FSR/FSX are inputs to McBSP), and FSRP = FSXP = 1, the external active-low frame-synchronization signals are inverted before being sent to the receiver (internal FSR) and transmitter (internal FSX). Similarly, if internal synchronization (FSR/FSX are output pins and GSYNC = 0) is selected, the internal active-high frame-synchronization signals are inverted, if the polarity bit FS(R/X)P = 1, before being sent to the FS(R/X) pin.

On the transmit side, the transmit clock polarity bit, CLKXP, sets the edge used to shift and clock out transmit data. Data is always transmitted on the rising edge of internal CLKX. If CLKXP = 1 and external clocking is selected (CLKXM = 0 and CLKX is an input), the external falling-edge triggered input clock on CLKX is inverted to a rising-edge triggered clock before being sent to the transmitter. If CLKXP = 1, and internal clocking selected (CLKXM = 1 and CLKX is an output pin), the internal (rising-edge triggered) clock, internal CLKX, is inverted before being sent out on the MCLKX pin.

Similarly, the receiver can reliably sample data that is clocked with a rising edge clock (by the transmitter). The receive clock polarity bit, CLKRP, sets the edge used to sample received data. The receive data is always sampled on the falling edge of internal MCLKR. Therefore, if CLKRP = 1 and external clocking is selected (CLKRM = 0 and MCLKR is an input pin), the external rising-edge triggered input clock on MCLKR is inverted to a falling-edge triggered clock before being sent to the receiver. If CLKRP = 1 and internal clocking is selected (CLKRM = 1), the internal falling-edge triggered clock is inverted to a rising-edge triggered clock before being sent out on the MCLKR pin.

MCLKRP = CLKXP in a system where the same clock (internal or external) is used to clock the receiver and transmitter. The receiver uses the opposite edge as the transmitter to make sure of valid setup and hold of data around this edge. Figure 34-48 shows how data clocked by an external serial device using a rising edge can be sampled by the McBSP receiver on the falling edge of the same clock.

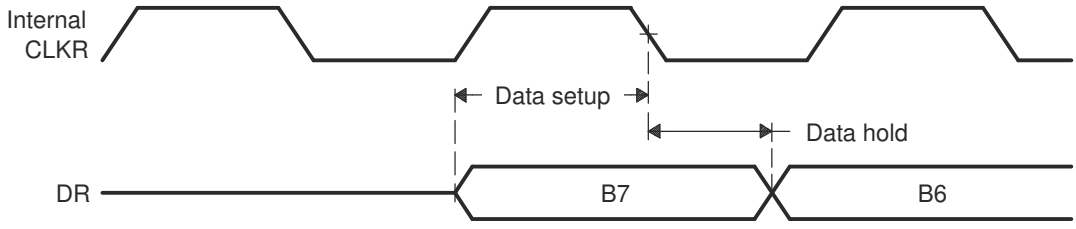


Figure 34-48. Data Clocked Externally Using a Rising Edge and Sampled by the McBSP Receiver on a Falling Edge

Set the SRG Frame-Synchronization Period and Pulse Width.

34.8.16.2 Frame-Synchronization Period and the Frame-Synchronization Pulse Width

The sample rate generator can produce a clock signal, CLKG, and a frame-synchronization signal, FSG. If the sample rate generator is supplying receive or transmit frame synchronization, you must program the bit fields FPER and FWID.

On FSG, the period from the start of a frame-synchronization pulse to the start of the next pulse is (FPER + 1) CLKG cycles. The 12 bits of FPER allow a frame-synchronization period of 1 to 4096 CLKG cycles, which allows up to 4096 data bits per frame. When GSYNC = 1, FPER is a don't care value.

Each pulse on FSG has a width of (FWID + 1) CLKG cycles. The eight bits of FWID allow a pulse width of 1 to 256 CLKG cycles. It is recommended that FWID be programmed to a value less than the programmed word length.

The values in FPER and FWID are loaded into separate down-counters. The 12-bit FPER counter counts down the generated clock cycles from the programmed value (4095 maximum) to 0. The 8-bit FWID counter counts down from the programmed value (255 maximum) to 0. Table 34-39 shows settings for FPER and FWID.

Figure 34-49 shows a frame-synchronization period of 16 CLKG periods (FPER = 15 or 00001111b) and a frame-synchronization pulse with an active width of 2 CLKG periods (FWID = 1).

Table 34-39. Register Bits Used to Set the SRG Frame-Synchronization Period and Pulse Width

Register	Bit	Name	Function	Type	Reset Value
SRGR2	11-0	FPER	Sample rate generator frame-synchronization period For the frame-synchronization signal FSG, (FPER + 1) determines the period from the start of a frame-synchronization pulse to the start of the next frame-synchronization pulse. Range for (FPER + 1): 1 to 4096 CLKG cycles	R/W	0000 0000 0000
SRGR1	15-8	FWID	Sample rate generator frame-synchronization pulse width This field plus 1 determines the width of each frame-synchronization pulse on FSG. Range for (FWID + 1): 1 to 256 CLKG cycles	R/W	0000 0000

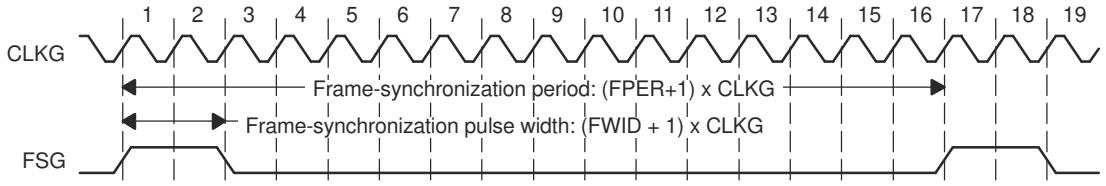


Figure 34-49. Frame of Period 16 CLKG Periods and Active Width of 2 CLKG Periods

When the sample rate generator comes out of reset, FSG is in the inactive state. Then, when GRST = 1 and FSGM = 1, a frame-synchronization pulse is generated. The frame width value (FWID + 1) is counted down on every CLKG cycle until the value reaches 0, at which time FSG goes low. At the same time, the frame period value (FPER + 1) is also counting down. When this value reaches 0, FSG goes high, indicating a new frame.

34.8.17 Receive Clock Mode

Table 34-40 shows the settings for bits used to set receive clock mode.

Table 34-40. Register Bits Used to Set the Receive Clock Mode

Register	Bit	Name	Function		Type	Reset Value
PCR	8	CLKRM	Receive clock mode		R/W	0
			Case 1: Digital loopback mode not set (DLB = 0) in SPCR1.			
			CLKRM = 0	The MCLKR pin is an input pin that supplies the internal receive clock (MCLKR).		
			CLKRM = 1	Internal MCLKR is driven by the sample rate generator of the McBSP. The MCLKR pin is an output pin that reflects internal MCLKR.		
			Case 2: Digital loopback mode set (DLB = 1) in SPCR1.			
			CLKRM = 0	The MCLKR pin is in the high impedance state. The internal receive clock (MCLKR) is driven by the internal transmit clock (CLKX). Internal CLKX is derived according to the CLKXM bit of PCR.		
			CLKRM = 1	Internal MCLKR is driven by internal CLKX. The MCLKR pin is an output pin that reflects internal MCLKR. Internal CLKX is derived according to the CLKXM bit of PCR.		
SPCR1	15	DLB	Digital loopback mode		R/W	00
			DLB = 0	Digital loopback mode is disabled.		
			DLB = 1	Digital loopback mode is enabled. The receive signals, including the receive frame-synchronization signal, are connected internally through multiplexers to the corresponding transmit signals.		
SPCR1	12-11	CLKSTP	Clock stop mode		R/W	00
			CLKSTP = 0Xb	Clock stop mode disabled; normal clocking for non-SPI mode.		
			CLKSTP = 10b	Clock stop mode enabled without clock delay. The internal receive clock signal (MCLKR) and the internal receive frame-synchronization signal (FSR) are internally connected to their transmit counterparts, CLKX and FSX.		
			CLKSTP = 11b	Clock stop mode enabled with clock delay. The internal receive clock signal (MCLKR) and the internal receive frame-synchronization signal (FSR) are internally connected to their transmit counterparts, CLKX and FSX.		

34.8.17.1 Selecting a Source for the Receive Clock and a Data Direction for the MCLKR Pin

Table 34-41 shows how you can select various sources to provide the receive clock signal and affect the MCLKR pin. The polarity of the signal on the MCLKR pin is determined by the CLKRP bit.

In the digital loopback mode (DLB = 1), the transmit clock signal is used as the receive clock signal.

Also, in the clock stop mode, the internal receive clock signal (MCLKR) and the internal receive frame-synchronization signal (FSR) are internally connected to their transmit counterparts, CLKX and FSX.

Table 34-41. Receive Clock Signal Source Selection

DLB in SPCR1	CLKRM in PCR	Source of Receive Clock	MCLKR Pin Status
0	0	The MCLKR pin is an input driven by an external clock. The external clock signal is inverted as determined by CLKRP before being used.	Input
0	1	The sample rate generator clock (CLKG) drives internal MCLKR.	Output. CLKG, inverted as determined by CLKRP, is driven out on the MCLKR pin.
1	0	Internal CLKX drives internal MCLKR. To configure CLKX, see Section 34.9.19.	High impedance
1	1	Internal CLKX drives internal MCLKR. To configure CLKX, see Section 34.9.19.	Output. Internal MCLKR (same as internal CLKX) is inverted as determined by CLKRP before being driven out on the MCLKR pin.

34.8.18 Receive Clock Polarity

Table 34-42 shows which register bits set the Receive Clock Polarity.

Table 34-42. Register Bit Used to Set Receive Clock Polarity

Register	Bit	Name	Function		Type	Reset Value
PCR	0	CLKRP	Receive clock polarity		R/W	0
			CLKRP = 0	Receive data sampled on falling edge of MCLKR		
			CLKRP = 1	Receive data sampled on rising edge of MCLKR		

34.8.18.1 Frame Synchronization Pulses, Clock Signals, and Their Polarities

Receive frame-synchronization pulses can be generated internally by the sample rate generator (see Section 34.4.2) or driven by an external source. The source of frame synchronization is selected by programming the mode bit, FSRM, in PCR. FSR is also affected by the GSYNC bit in SRGR2. For information about the effects of FSRM and GSYNC, see Section 34.8.15. Similarly, receive clocks can be selected to be inputs or outputs by programming the mode bit, CLKRM, in the PCR (see Section 34.8.17).

When FSR and FSX are inputs (FSXM = FSRM= 0, external frame-synchronization pulses), the McBSP detects them on the internal falling edge of clock, internal MCLKR, and internal CLKX, respectively. The receive data arriving at the DR pin is also sampled on the falling edge of internal MCLKR. These internal clock signals are either derived from external source via CLK(R/X) pins or driven by the sample rate generator clock (CLKG) internal to the McBSP.

When FSR and FSX are outputs, implying that the outputs are driven by the sample rate generator, the outputs are generated (transition to their active state) on the rising edge of internal clock, CLK(R/X). Similarly, data on the DX pin is output on the rising edge of internal CLKX.

FSRP, FSXP, CLKRP, and CLKXP in the pin control register (PCR) configure the polarities of the FSR, FSX, MCLKR, and CLKX signals, respectively. All frame-synchronization signals (internal FSR, internal FSX) that are internal to the serial port are active high. If the serial port is configured for external frame synchronization (FSR/FSX are inputs to McBSP) and FSRP = FSXP = 1, the external active-low frame-synchronization signals are inverted before being sent to the receiver (internal FSR) and transmitter (internal FSX). Similarly, if internal synchronization (FSR/FSX are output pins and GSYNC = 0) is selected, the internal active-high frame-synchronization signals are inverted, if the polarity bit FS(R/X)P = 1, before being sent to the FS(R/X) pin.

On the transmit side, the transmit clock polarity bit, CLKXP, sets the edge used to shift and clock out transmit data. Data is always transmitted on the rising edge of internal CLKX. If CLKXP = 1 and external clocking is selected (CLKXM = 0 and CLKX is an input), the external falling-edge triggered input clock on CLKX is inverted to a rising-edge triggered clock before being sent to the transmitter. If CLKXP = 1 and internal clocking is selected (CLKXM = 1 and CLKX is an output pin), the internal (rising-edge triggered) clock, internal CLKX, is inverted before being sent out on the MCLKX pin.

Similarly, the receiver can reliably sample data that is clocked with a rising edge clock (by the transmitter). The receive clock polarity bit, CLKRP, sets the edge used to sample received data. The receive data is always sampled on the falling edge of internal MCLKR. Therefore, if CLKRP = 1 and external clocking is selected (CLKRM = 0 and MCLKR is an input pin), the external rising-edge triggered input clock on MCLKR is inverted to a falling-edge triggered clock before being sent to the receiver. If CLKRP = 1 and internal clocking is selected (CLKRM = 1), the internal falling-edge triggered clock is inverted to a rising-edge triggered clock before being sent out on the MCLKR pin.

CLKRP = CLKXP in a system where the same clock (internal or external) is used to clock the receiver and transmitter. The receiver uses the opposite edge as the transmitter to make sure of valid setup and hold of data around this edge. Figure 34-50 shows how data clocked by an external serial device using a rising edge can be sampled by the McBSP receiver on the falling edge of the same clock.

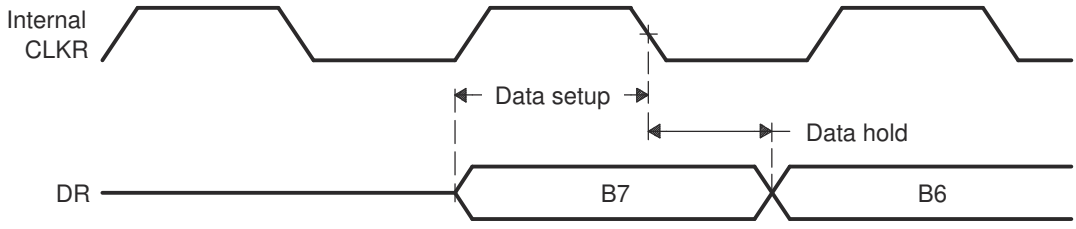


Figure 34-50. Data Clocked Externally Using a Rising Edge and Sampled by the McBSP Receiver on a Falling Edge

34.8.19 SRG Clock Divide-Down Value

Table 34-43. Register Bits Used to Set the Sample Rate Generator (SRG) Clock Divide-Down Value

Register	Bit	Name	Function	Type	Reset Value
SRGR1	7-0	CLKGDV	Sample rate generator clock divide-down value The input clock of the sample rate generator is divided by (CLKGDV + 1) to generate the required sample rate generator clock frequency. The default value of CLKGDV is 1 (divide input clock by 2).	R/W	0000 0001

34.8.19.1 Sample Rate Generator Clock Divider

The first divider stage generates the serial data bit clock from the input clock. This divider stage utilizes a counter, preloaded by CLKGDV, that contains the divide ratio value.

The output of the first divider stage is the data bit clock, which is output as CLKG and which serves as the input for the second and third stages of the divider.

CLKG has a frequency equal to 1/(CLKGDV + 1) of sample rate generator input clock. Thus, the sample generator input clock frequency is divided by a value between 1 and 256. When CLKGDV is odd or equal to 0, the CLKG duty cycle is 50%. When CLKGDV is an even value, 2p, representing an odd divide-down, the high-state duration is p + 1 cycles and the low-state duration is p cycles.

34.8.20 SRG Clock Synchronization Mode

For more details on using the clock synchronization feature, see Section 34.4.3.

Table 34-44. Register Bit Used to Set the SRG Clock Synchronization Mode

Register	Bit	Name	Function	Type	Reset Value
SRGR2	15	GSYNC	Sample rate generator clock synchronization GSYNC is used only when the input clock source for the sample rate generator is external—on the MCLKR or MCLKX pin. GSYNC = 0: The sample rate generator clock (CLKG) is free running. CLKG oscillates without adjustment, and FSG pulses every (FPER + 1) CLKG cycles. GSYNC = 1: Clock synchronization is performed. When a pulse is detected on the FSR pin: • CLKG is adjusted as necessary so that CLKG is synchronized with the input clock on the MCLKR or MCLKX pin. • FSG pulses. FSG only pulses in response to a pulse on the FSR pin. The frame-synchronization period defined in FPER is ignored.	R/W	0

34.8.21 SRG Clock Mode (Choose an Input Clock)

The sample rate generator can produce a clock signal (CLKG) for use by the receiver, the transmitter, or both, but CLKG is derived from an input clock. Table 34-45 shows the four possible sources of the input clock. For more details on generating CLKG, see Section 34.4.1.1.

Table 34-45. Register Bits Used to Set the SRG Clock Mode (Choose an Input Clock)

Register	Bit	Name	Function		Type	Reset Value
PCR	7	SCLKME	Sample rate generator clock mode		R/W	0
SRGR2	13	CLKSM			R/W	1
			SCLKME = 0 CLKSM = 0	Reserved		
			SCLKME = 0 CLKSM = 1	Sample rate generator clock derived from LSPCLK (default)		
			SCLKME = 1 CLKSM = 0	Sample rate generator clock derived from MCLKR pin		
			SCLKME = 1 CLKSM = 1	Sample rate generator clock derived from MCLKX pin		

34.8.22 SRG Input Clock Polarity

Table 34-46 shows which register bits set the SRG Input Clock Polarity.

Table 34-46. Register Bits Used to Set the SRG Input Clock Polarity

Register	Bit	Name	Function		Type	Reset Value
PCR	1	CLKXP	MCLKX pin polarity		R/W	0
			CLKXP determines the input clock polarity when the MCLKX pin supplies the input clock (SCLKME = 1 and CLKSM = 1).			
			CLKXP = 0	Rising edge on MCLKX pin generates transitions on CLKG and FSG.		
			CLKXP = 1	Falling edge on MCLKX pin generates transitions on CLKG and FSG.		
PCR	0	CLKRP	MCLKR pin polarity		R/W	0
			CLKRP determines the input clock polarity when the MCLKR pin supplies the input clock (SCLKME = 1 and CLKSM = 0).			
			CLKRP = 0	Falling edge on MCLKR pin generates transitions on CLKG and FSG.		
			CLKRP = 1	Rising edge on MCLKR pin generates transitions on CLKG and FSG.		

34.8.22.1 Using CLKXP/CLKRP to Choose an Input Clock Polarity

The sample rate generator can produce a clock signal (CLKG) and a frame-synchronization signal (FSG) for use by the receiver, the transmitter, or both. To produce CLKG and FSG, the sample rate generator must be driven by an input clock signal derived from the CPU clock or from an external clock on the CLKX or MCLKR pin. If you use a pin, choose a polarity for that pin by using the appropriate polarity bit (CLKXP for the MCLKX pin, CLKRP for the MCLKR pin). The polarity determines whether the rising or falling edge of the input clock generates transitions on CLKG and FSG.

34.9 Transmitter Configuration

To configure the McBSP transmitter, perform the following procedure:

1. Place the McBSP/transmitter in reset (see Section 34.9.2).
2. Program the McBSP registers for the desired transmitter operation (see Section 34.9.1).
3. Take the transmitter out of reset (see Section 34.9.2).

34.9.1 Programming the McBSP Registers for the Desired Transmitter Operation

The following is a list of important tasks to be performed when you are configuring the McBSP transmitter. Each task corresponds to one or more McBSP register bit fields.

- Global behavior:
  - Set the transmitter pins to operate as McBSP pins.
  - Enable/disable the digital loopback mode.
  - Enable/disable the clock stop mode.
  - Enable/disable transmit multichannel selection.
- Data behavior:
  - Choose 1 or 2 phases for the transmit frame.
  - Set the transmit word length(s).
  - Set the transmit frame length.
  - Enable/disable the transmit frame-synchronization ignore function.
  - Set the transmit companding mode.
  - Set the transmit data delay.
  - Set the transmit DXENA mode.
  - Set the transmit interrupt mode.
- Frame-synchronization behavior:
  - Set the transmit frame-synchronization mode.
  - Set the transmit frame-synchronization polarity.
  - Set the SRG frame-synchronization period and pulse width.
- Clock behavior:
  - Set the transmit clock mode.
  - Set the transmit clock polarity.
  - Set the SRG clock divide-down value.
  - Set the SRG clock synchronization mode.
  - Set the SRG clock mode (choose an input clock).
  - Set the SRG input clock polarity.

34.9.2 Resetting and Enabling the Transmitter

The first step of the transmitter configuration procedure is to reset the transmitter, and the last step is to enable the transmitter (to take the transmitter out of reset). Table 34-47 describes the bits used for both of these steps.

Table 34-47. Register Bits Used to Place Transmitter in Reset Field Descriptions

Register	Bit	Field	Value	Description
SPCR2	7	FRST		Frame-synchronization logic reset
			0	Frame-synchronization logic is reset. The sample rate generator does not generate frame-synchronization signal FSG, even if GRST = 1.
			1	Frame-synchronization is enabled. If GRST = 1, frame-synchronization signal FSG is generated after (FPER + 1) number of CLKG clock cycles; all frame counters are loaded with their programmed values.
SPCR2	6	GRST		Sample rate generator reset
			0	Sample rate generator is reset. If GRST = 0 due to a device reset, CLKG is driven by the CPU clock divided by 2, and FSG is driven low (inactive). If GRST = 0 due to program code, CLKG and FSG are both driven low (inactive).
			1	Sample rate generator is enabled. CLKG is driven according to the configuration programmed in the sample rate generator registers (SRGR[1,2]). If FRST = 1, the generator also generates the frame-synchronization signal FSG as programmed in the sample rate generator registers.
SPCR2	0	XRST		Transmitter reset
			0	The serial port transmitter is disabled and in the reset state.
			1	The serial port transmitter is enabled.

34.9.2.1 Reset Considerations

The serial port can be reset in the following two ways:

1. A DSP reset ($\overline{\text{XRS}}$ signal driven low) places the receiver, transmitter, and sample rate generator in reset. When the device reset is removed, GRST = FRST = RRST = XRST = 0, keeping the entire serial port in the reset state.
2. The serial port transmitter and receiver can be reset directly using the RRST and XRST bits in the serial port control registers. The sample rate generator can be reset directly using the GRST bit in SPCR2.
3. When using the DMA, the order in which McBSP events must occur is important. DMA channel and peripheral interrupts must be configured prior to releasing the McBSP transmitter from reset.

   The reason for this is that an XRDY is fired when XRST = 1. The XRDY signals the DMA to start copying data from the buffer into the transmit register. If the McBSP transmitter is released from reset before the DMA channel and peripheral interrupts are configured, the XRDY signals before the DMA channel can receive the signal; therefore, the DMA does not move the data from the buffer to the transmit register. The DMA PERINTFLG is edge-sensitive and will fail to recognize the XRDY, which is continuously high.

For more details about McBSP reset conditions and effects, see Section 34.10.2.

34.9.3 Set the Transmitter Pins to Operate as McBSP Pins

To configure a pin for McBSP functioning, configure the bits of the GPxMUXn register appropriately. In addition to this, bits 12 and 13 of the PCR register must be set to 0. These bits are defined as reserved.

34.9.4 Digital Loopback Mode

The DLB bit determines whether the digital loopback mode is on. DLB is described in Table 34-48.

Table 34-48. Register Bit Used to Enable/Disable the Digital Loopback Mode

Register	Bit	Name	Function		Type	Reset Value
SPCR1	15	DLB	Digital loopback mode		R/W	0
			DLB = 0	Digital loopback mode is disabled.		
			DLB = 1	Digital loopback mode is enabled.		

In the digital loopback mode, the receive signals are connected internally through multiplexers to the corresponding transmit signals, as shown in Table 34-49. This mode allows testing of serial port code with a single DSP device; the McBSP receives the data it transmits.

Table 34-49. Receive Signals Connected to Transmit Signals in Digital Loopback Mode

This Receive Signal	Is Fed Internally by This Transmit Signal
DR (receive data)	DX (transmit data)
FSR (receive frame synchronization)	FSX (transmit frame synchronization)
MCLKR (receive clock)	CLKX (transmit clock)

34.9.5 Clock Stop Mode

The CLKSTP bits determine whether the clock stop mode is on. CLKSTP is described in Table 34-50.

Table 34-50. Register Bits Used to Enable/Disable the Clock Stop Mode

Register	Bit	Name	Function		Type	Reset Value
SPCR1	12-11	CLKSTP	Clock stop mode		R/W	00
			CLKSTP = 0Xb	Clock stop mode disabled; normal clocking for non-SPI mode.		
			CLKSTP = 10b	Clock stop mode enabled without clock delay		
			CLKSTP = 11b	Clock stop mode enabled with clock delay		

The clock stop mode supports the SPI master-slave protocol. If you do not plan to use the SPI protocol, you can clear CLKSTP to disable the clock stop mode.

In the clock stop mode, the clock stops at the end of each data transfer. At the beginning of each data transfer, the clock starts immediately (CLKSTP = 10b) or after a half-cycle delay (CLKSTP = 11b). The CLKXP bit determines whether the starting edge of the clock on the MCLKX pin is rising or falling. The CLKRP bit determines whether receive data is sampled on the rising or falling edge of the clock shown on the MCLKR pin.

Table 34-51 summarizes the impact of CLKSTP, CLKXP, and CLKRP on serial port operation. In the clock stop mode, the receive clock is tied internally to the transmit clock, and the receive frame-synchronization signal is tied internally to the transmit frame-synchronization signal.

Table 34-51. Effects of CLKSTP, CLKXP, and CLKRP on the Clock Scheme

Bit Settings	Clock Scheme
CLKSTP = 00b or 01b CLKXP = 0 or 1 CLKRP = 0 or 1	Clock stop mode disabled. Clock enabled for non-SPI mode.
CLKSTP = 10b CLKXP = 0 CLKRP = 0	Low inactive state without delay: The McBSP transmits data on the rising edge of CLKX and receives data on the falling edge of MCLKR.
CLKSTP = 11b CLKXP = 0 CLKRP = 1	Low inactive state with delay: The McBSP transmits data one-half cycle ahead of the rising edge of CLKX and receives data on the rising edge of MCLKR.
CLKSTP = 10b CLKXP = 1 CLKRP = 0	High inactive state without delay: The McBSP transmits data on the falling edge of CLKX and receives data on the rising edge of MCLKR.
CLKSTP = 11b CLKXP = 1 CLKRP = 1	High inactive state with delay: The McBSP transmits data one-half cycle ahead of the falling edge of CLKX and receives data on the falling edge of MCLKR.

34.9.6 Transmit Multichannel Selection Mode

For more details, see Section 34.6.7.

Table 34-52. Register Bits Used to Enable/Disable Transmit Multichannel Selection

Register	Bit	Name	Function		Type	Reset Value
MCR2	1-0	XMCM	Transmit multichannel selection		R/W	00
			XMCM = 00b	No transmit multichannel selection mode is on. All channels are enabled and unmasked. No channels can be disabled or masked.		
			XMCM = 01b	All channels are disabled unless the channels are selected in the appropriate transmit channel enable registers (XCERs). If enabled, a channel in this mode is also unmasked. The XMCME bit determines whether 32 channels or 128 channels are selectable in XCERs.		
			XMCM = 10b	All channels are enabled, but the channels are masked unless the channels are selected in the appropriate transmit channel enable registers (XCERs). The XMCME bit determines whether 32 channels or 128 channels are selectable in XCERs.		
			XMCM = 11b	This mode is used for symmetric transmission and reception. All channels are disabled for transmission unless the channels are enabled for reception in the appropriate receive channel enable registers (RCERs). Once enabled, the channels are masked unless the channels are also selected in the appropriate transmit channel enable registers (XCERs). The XMCME bit determines whether 32 channels or 128 channels are selectable in RCERs and XCERs.		

34.9.7 XCERs Used in the Transmit Multichannel Selection Mode

For multichannel selection operation, the assignment of channels to the XCERs depends on whether 32 or 128 channels are individually selectable by the XMCME bit, as shown in Table 34-53. The table shows which block of channels is assigned to each XCER that is used. For each XCER, the table shows which channel is assigned to each of the bits.

Note

When XMCM = 11b (for symmetric transmission and reception), the transmitter uses the receive channel enable registers (RCERs) to enable channels and uses the XCERs to unmask channels for transmission.

Table 34-53. Use of the Transmit Channel Enable Registers

Number of Selectable Channels	Block Assignments		Channel Assignments	
	XCERx	Block Assigned	Bit in XCERx	Channel Assigned
32 (XMCME = 0)	XCERA	Channels n to (n + 15)	XCE0	Channel n
			XCE1	Channel (n + 1)
			XCE2	Channel (n + 2)
			:	:
		When XMCM = 01b or 10b, the block of channels is chosen with the XPABLK bits. When XMCM = 11b, the block is chosen with the RPABLK bits.	XCE15	Channel (n + 15)
	XCERB	Channels m to (m + 15)	XCE0	Channel m
			XCE1	Channel (m + 1)
			XCE2	Channel (m + 2)
			:	:
		When XMCM = 01b or 10b, the block of channels is chosen with the XPBBLK bits. When XMCM = 11b, the block is chosen with the RPBBLK bits.	XCE15	Channel (m + 15)
128 (XMCME = 1)	XCERA	Block 0	XCE0	Channel 0
			XCE1	Channel 1
			XCE2	Channel 2
			:	:
			XCE15	Channel 15
	XCERB	Block 1	XCE0	Channel 16
			XCE1	Channel 17
			XCE2	Channel 18
			:	:
			XCE15	Channel 31
	XCERC	Block 2	XCE0	Channel 32
			XCE1	Channel 33
			XCE2	Channel 34
			:	:
			XCE15	Channel 47
	XCERD	Block 3	XCE0	Channel 48
			XCE1	Channel 49
			XCE2	Channel 50
			:	:
			XCE15	Channel 63

Table 34-53. Use of the Transmit Channel Enable Registers (continued)

Number of Selectable Channels	Block Assignments		Channel Assignments	
	XCERx	Block Assigned	Bit in XCERx	Channel Assigned
	XCERE	Block 4	XCE0	Channel 64
			XCE1	Channel 65
			XCE2	Channel 66
			:	:
			XCE15	Channel 79
	XCERF	Block 5	XCE0	Channel 80
			XCE1	Channel 81
			XCE2	Channel 82
			:	:
			XCE15	Channel 95
	XCERG	Block 6	XCE0	Channel 96
			XCE1	Channel 97
			XCE2	Channel 98
			:	:
			XCE15	Channel 111
	XCERH	Block 7	XCE0	Channel 112
			XCE1	Channel 113
			XCE2	Channel 114
			:	:
			XCE15	Channel 127

34.9.8 Transmit Frame Phases

The XPHASE bit (see Table 34-54) determines whether the transmit data frame has one or two phases.

Table 34-54. Register Bit Used to Choose 1 or 2 Phases for the Transmit Frame

Register	Bit	Name	Function		Type	Reset Value
XCR2	15	XPHASE	Transmit phase number		R/W	0
			Specifies whether the transmit frame has 1 or 2 phases.			
			XPHASE = 0	Single-phase frame		
			XPHASE = 1	Dual-phase frame		

34.9.9 Transmit Word Lengths

The XWDLEN1 and XWDLEN2 bit fields (see Table 34-55) determine how many bits are in each serial word in phase 1 and in phase 2, respectively, of the transmit data frame.

Table 34-55. Register Bits Used to Set the Transmit Word Lengths

Register	Bit	Name	Function		Type	Reset Value
XCR1	7-5	XWDLEN1	Transmit word length of frame phase 1		R/W	000
			XWDLEN1 = 000b	8 bits		
			XWDLEN1 = 001b	12 bits		
			XWDLEN1 = 010b	16 bits		
			XWDLEN1 = 011b	20 bits		
			XWDLEN1 = 100b	24 bits		
			XWDLEN1 = 101b	32 bits		
			XWDLEN1 = 11Xb	Reserved		
XCR2	7-5	XWDLEN2	Transmit word length of frame phase 2		R/W	000
			XWDLEN2 = 000b	8 bits		
			XWDLEN2 = 001b	12 bits		
			XWDLEN2 = 010b	16 bits		
			XWDLEN2 = 011b	20 bits		
			XWDLEN2 = 100b	24 bits		
			XWDLEN2 = 101b	32 bits		
			XWDLEN2 = 11Xb	Reserved		

34.9.9.1 Word Length Bits

Each frame can have one or two phases, depending on the value that you load into the RPHASE bit. If a single-phase frame is selected, XWDLEN1 selects the length for every serial word transmitted in the frame. If a dual-phase frame is selected, XWDLEN1 determines the length of the serial words in phase 1 of the frame, and XWDLEN2 determines the word length in phase 2 of the frame.

34.9.10 Transmit Frame Length

The XFRLEN1 and XFRLEN2 bit fields (see Table 34-56) determine how many serial words are in phase 1 and in phase 2, respectively, of the transmit data frame.

Table 34-56. Register Bits Used to Set the Transmit Frame Length

Register	Bit	Name	Function		Type	Reset Value
XCR1	14-8	XFRLEN1	Transmit frame length 1		R/W	000 0000
			(XFRLEN1 + 1) is the number of serial words in phase 1 of the transmit frame.			
			XFRLEN1 = 000 0000	1 word in phase 1		
			XFRLEN1 = 000 0001	2 words in phase 1		
			|	|		
			|	|		
			XFRLEN1 = 111 1111	128 words in phase 1		
XCR2	14-8	XFRLEN2	Transmit frame length 2		R/W	000 0000
			If a dual-phase frame is selected, (XFRLEN2 + 1) is the number of serial words in phase 2 of the transmit frame.			
			XFRLEN2 = 000 0000	1 word in phase 2		
			XFRLEN2 = 000 0001	2 words in phase 2		
			|	|		
			|	|		
			XFRLEN2 = 111 1111	128 words in phase 2		

34.9.10.1 Selected Frame Length

The transmit frame length is the number of serial words in the transmit frame. Each frame can have one or two phases, depending on the value that you load into the XPHASE bit.

If a single-phase frame is selected (XPHASE = 0), the frame length is equal to the length of phase 1. If a dual-phase frame is selected (XPHASE = 1), the frame length is the length of phase 1 plus the length of phase 2.

The 7-bit XFRLEN fields allow up to 128 words per phase. See Table 34-57 for a summary of how to calculate the frame length. This length corresponds to the number of words or logical time slots or channels per frame-synchronization pulse.

Note

Program the XFRLEN fields with [*w minus 1*], where w represents the number of words per phase. For example, if you want a phase length of 128 words in phase 1, load 127 into XFRLEN1.

Table 34-57. How to Calculate Frame Length

XPHASE	XFRLEN1	XFRLEN2	Frame Length
0	0 ≤ XFRLEN1 ≤ 127	Don't care	(XFRLEN1 + 1) words
1	0 ≤ XFRLEN1 ≤ 127	0 ≤ XFRLEN2 ≤ 127	(XFRLEN1 + 1) + (XFRLEN2 + 1) words

34.9.11 Enable/Disable the Transmit Frame-Synchronization Ignore Function

Table 34-58. Register Bit Used to Enable/Disable the Transmit Frame-Synchronization Ignore Function

Register	Bit	Name	Function		Type	Reset Value
XCR2	2	XFIG	Transmit frame-synchronization ignore		R/W	0
			XFIG = 0	An unexpected transmit frame-synchronization pulse causes the McBSP to restart the frame transfer.		
			XFIG = 1	The McBSP ignores unexpected transmit frame-synchronization pulses.		

34.9.11.1 Unexpected Frame-Synchronization Pulses and Frame-Synchronization Ignore

If a frame-synchronization pulse starts the transfer of a new frame before the current frame is fully transmitted, this pulse is treated as an unexpected frame-synchronization pulse.

When XFIG = 1, normal transmission continues with unexpected frame-synchronization signals ignored.

When XFIG = 0 and an unexpected frame-synchronization pulse occurs, the serial port:

1. Aborts the present transmission
2. Sets XSYNCERR to 1 in SPCR2
3. Reinitiates transmission of the current word that was aborted

For more details about the frame-synchronization error condition, see Section 34.5.6.

34.9.11.2 Examples Showing the Effects of XFIG

Figure 34-51 shows an example in which word B is interrupted by an unexpected frame-synchronization pulse when (R/X)FIG = 0. In the case of transmission, the transmission of B is aborted (B is lost). This condition is a transmit synchronization error, which sets the XSYNCERR bit. No new data has been written to DXR[1,2]; therefore, the McBSP transmits B again.

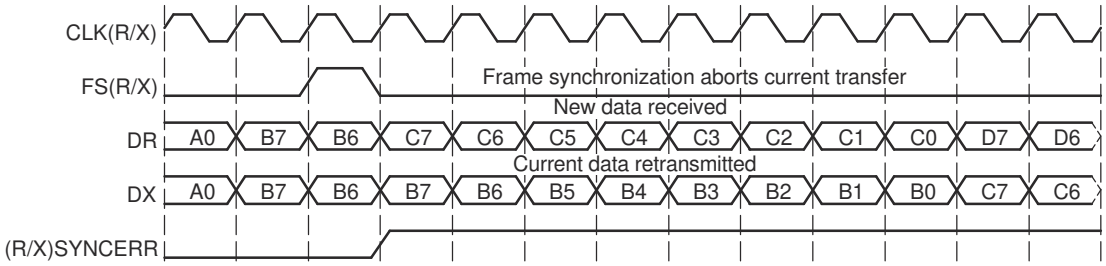


Figure 34-51. Unexpected Frame-Synchronization Pulse With (R/X) FIG = 0

In contrast with Figure 34-51, Figure 34-52 shows McBSP operation when unexpected frame-synchronization signals are ignored (when (R/X)FIG = 1). Here, the transfer of word B is not affected by an unexpected frame-synchronization pulse.

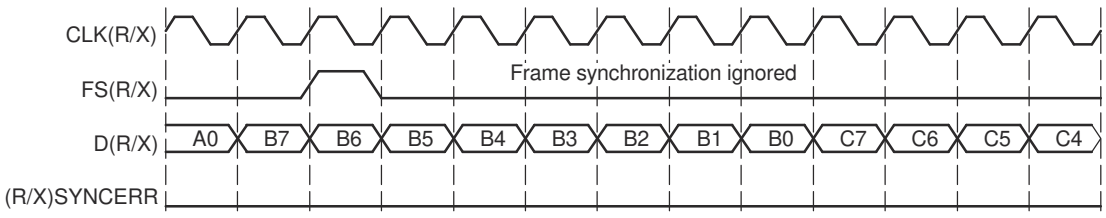


Figure 34-52. Unexpected Frame-Synchronization Pulse With (R/X) FIG = 1

34.9.12 Transmit Companding Mode

The XCOMPAND bits (see Table 34-59) determine whether companding or another data transfer option is chosen for McBSP transmission.

Table 34-59. Register Bits Used to Set the Transmit Companding Mode

Register	Bit	Name	Function		Type	Reset Value
XCR2	4-3	XCOMPAND	Transmit companding mode		R/W	00
			Modes other than 00b are enabled only when the appropriate XWDLEN is 000b, indicating 8-bit data.			
			XCOMPAND = 00b	No companding, any size data, MSB transmitted first		
			XCOMPAND = 01b	No companding, 8-bit data, LSB transmitted first (for details, see Section 34.8.11.4)		
			XCOMPAND = 10b	μ-law companding, 8-bit data, MSB transmitted first		
			XCOMPAND = 11b	A-law companding, 8-bit data, MSB transmitted first		

34.9.12.1 Companding

Companding (COMpressing and exPANDing) hardware allows compression and expansion of data in either μ-law or A-law format. The companding standard employed in the United States and Japan is μ-law. The European companding standard is referred to as A-law. The specifications for μ-law and A-law log PCM are part of the CCITT G.711 recommendation.

A-law and μ-law allow 13 bits and 14 bits of dynamic range, respectively. Any values outside this range are set to the most positive or most negative value. Thus, for companding to work best, the data transferred to and from the McBSP via the CPU or DMA controller must be at least 16 bits wide.

The μ-law and A-law formats both encode data into 8-bit code words. Companded data is always 8 bits wide; the appropriate word length bits (RWDLEN1, RWDLEN2, XWDLEN1, XWDLEN2) must therefore be set to 0, indicating an 8-bit wide serial data stream. If companding is enabled and either of the frame phases does not have an 8-bit word length, companding continues as if the word length is 8 bits.

Figure 34-53 illustrates the companding processes. When companding is chosen for the transmitter, compression occurs during the process of copying data from DXR1 to XSR1. The transmit data is encoded according to the specified companding law (A-law or μ-law). When companding is chosen for the receiver, expansion occurs during the process of copying data from RBR1 to DRR1. The receive data is decoded to twos-complement format.

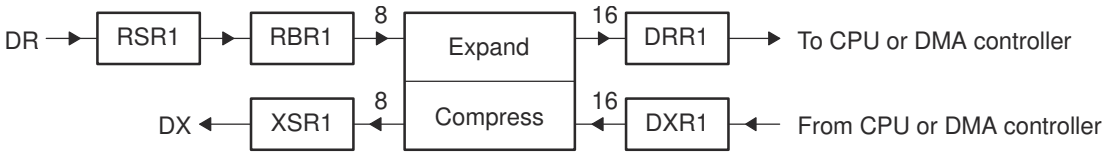


Figure 34-53. Companding Processes for Reception and for Transmission

34.9.12.2 Format for Data To Be Compressed

For transmission using μ-law compression, make sure the 14 data bits are left-justified in DXR1, with the remaining two low-order bits filled with 0s as shown in Figure 34-54.

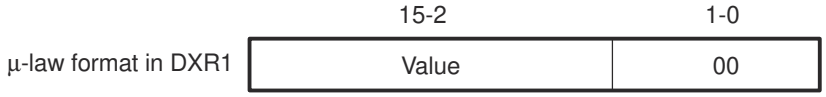


Figure 34-54. μ-Law Transmit Data Companding Format

For transmission using A-law compression, make sure the 13 data bits are left-justified in DXR1, with the remaining three low-order bits filled with 0s as shown in Figure 34-55.

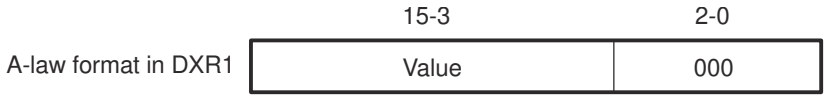


Figure 34-55. A-Law Transmit Data Companding Format

34.9.12.3 Capability to Compand Internal Data

If the McBSP is otherwise unused (the serial port transmit and receive sections are reset), the companding hardware can compand internal data. See Section 34.3.2.2.

34.9.12.4 Option to Transmit LSB First

Normally, the McBSP transmit or receives all data with the most significant bit (MSB) first. However, certain 8-bit data protocols (that do not use companded data) require the least significant bit (LSB) to be transferred first. If you set XCOMPAND = 01b in XCR2, the bit ordering of 8-bit words is reversed (LSB first) before being sent from the serial port. Similar to companding, this feature is enabled only if the appropriate word length bits are set to 0, indicating that 8-bit words are to be transferred serially. If either phase of the frame does not have an 8-bit word length, the McBSP assumes the word length is eight bits and LSB-first ordering is done.

34.9.13 Transmit Data Delay

The XDATDLY bits (see Table 34-60) determine the length of the data delay for the transmit frame.

Table 34-60. Register Bits Used to Set the Transmit Data Delay

Register	Bit	Name	Function		Type	Reset Value
XCR2	1-0	XDATDLY	Transmitter data delay		R/W	00
			XDATDLY = 00	0-bit data delay		
			XDATDLY = 01	1-bit data delay		
			XDATDLY = 10	2-bit data delay		
			XDATDLY = 11	Reserved		

34.9.13.1 Data Delay

The start of a frame is defined by the first clock cycle in which frame synchronization is found to be active. The beginning of actual data reception or transmission with respect to the start of the frame can be delayed if necessary. This delay is called data delay.

XDATDLY specifies the data delay for transmission. The range of programmable data delay is zero to two bit-clocks (XDATDLY = 00b-10b), as described in Table 34-60 and Figure 34-56. In this figure, the data transferred is an 8-bit value with bits labeled B7, B6, B5, and so on. Typically a 1-bit delay is selected, because data often follows a 1-cycle active frame-synchronization pulse.

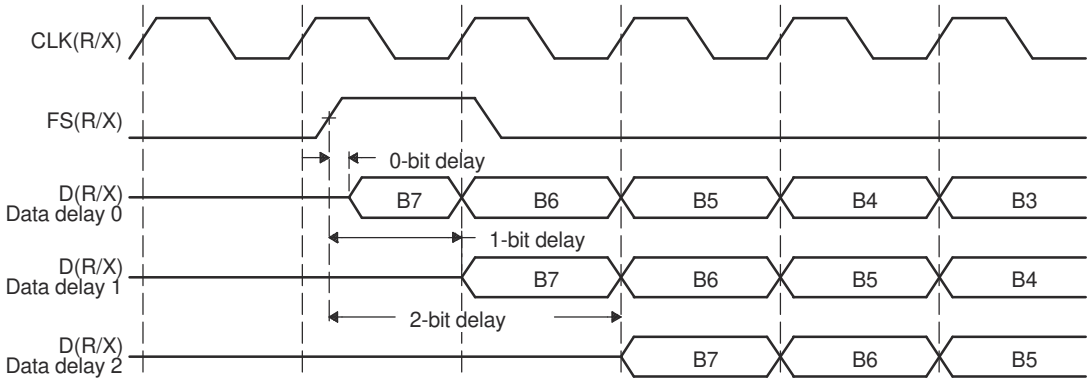


Figure 34-56. Range of Programmable Data Delay

34.9.13.2 0-Bit Data Delay

Normally, a frame-synchronization pulse is detected or sampled with respect to an edge of serial clock internal CLK(R/X). Thus, on the following cycle or later (depending on the data delay value), data can be received or transmitted. However, in the case of 0-bit data delay, the data must be ready for reception and/or transmission on the same serial clock cycle.

For reception this problem is solved because receive data is sampled on the first falling edge of MCLKR where an active-high internal FSR is detected. However, data transmission must begin on the rising edge of the internal CLKX clock that generated the frame synchronization. Therefore, the first data bit is assumed to be present in XSR1, and thus DX. The transmitter then asynchronously detects the frame synchronization, FSX, going active high and immediately starts driving the first bit to be transmitted on the DX pin.

34.9.13.3 2-Bit Data Delay

A data delay of two bit-periods allows the serial port to interface to different types of T1 framing devices where the data stream is preceded by a framing bit. During reception of such a stream with data delay of two bits (framing bit appears after a 1-bit delay and data appears after a 2-bit delay), the serial port essentially discards the framing bit from the data stream, as shown in Figure 34-57. In this figure, the data transferred is an 8-bit value with bits labeled B7, B6, B5, and so on.

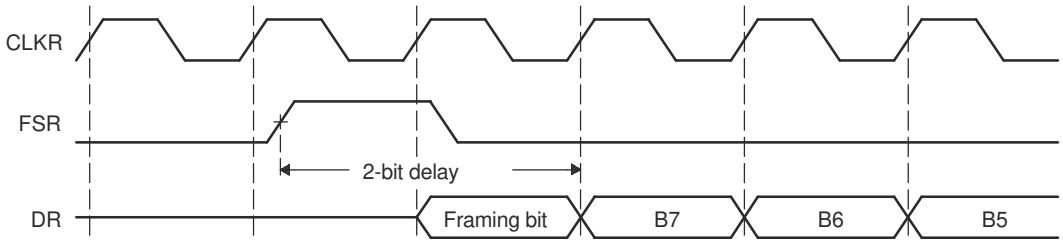


Figure 34-57. 2-Bit Data Delay Used to Skip a Framing Bit

34.9.14 Transmit DXENA Mode

Table 34-61 shows which register bit enables the Transmit DXENA (DX Delay Enabler) Mode.

Table 34-61. Register Bit Used to Set the Transmit DXENA (DX Delay Enabler) Mode

Register	Bit	Name	Function		Type	Reset Value
SPCR1	7	DXENA	DX delay enabler mode		R/W	0
			DXENA = 0	DX delay enabler is off.		
			DXENA = 1	DX delay enabler is on.		

The DXENA bit controls the delay enabler on the DX pin. Set DXENA to enable an extra delay for turn-on time. This bit does not control the data itself, so only the first bit is delayed.

If you tie together the DX pins of multiple McBSPs, make sure DXENA = 1 to avoid having more than one McBSP transmit on the data line at one time.

34.9.15 Transmit Interrupt Mode

The transmitter interrupt (XINT) signals the CPU of changes to the serial port status. Four options exist for configuring this interrupt. The options are set by the transmit interrupt mode bits, XINTM, in SPCR2.

Table 34-62. Register Bits Used to Set the Transmit Interrupt Mode

Register	Bit	Name	Function		Type	Reset Value
SPCR2	5-4	XINTM	Transmit interrupt mode		R/W	00
			XINTM = 00	XINT generated when XRDY changes from 0 to 1.		
			XINTM = 01	XINT generated by an end-of-block or end-of-frame condition in a transmit multichannel selection mode. In any of the transmit multichannel selection modes, interrupt after every 16-channel block boundary has been crossed within a frame and at the end of the frame. For details, see Section 34.6.8. In any other serial transfer case, this setting is not applicable and, therefore, no interrupts are generated.		
			XINTM = 10	XINT generated by a new transmit frame-synchronization pulse. Interrupt on detection of each transmit frame-synchronization pulse. This generates an interrupt even when the transmitter is in the reset state. This is done by synchronizing the incoming frame-synchronization pulse to the CPU clock and sending the pulse to the CPU using XINT.		
			XINTM = 11	XINT generated when XSYNCERR is set. Interrupt on frame-synchronization error. Regardless of the value of XINTM, XSYNCERR can be read to detect this condition. For more information on using XSYNCERR, see Section 34.5.6.		

34.9.16 Transmit Frame-Synchronization Mode

Table 34-63 shows which register bits enable the Transmit Frame-Synchronization Mode.

Table 34-63. Register Bits Used to Set the Transmit Frame-Synchronization Mode

Register	Bit	Name	Function		Type	Reset Value
PCR	11	FSXM	Transmit frame-synchronization mode		R/W	0
			FSXM = 0	Transmit frame synchronization is supplied by an external source via the FSX pin.		
			FSXM = 1	Transmit frame synchronization is supplied by the McBSP, as determined by the FSGM bit of SRGR2.		
SRGR2	12	FSGM	Sample rate generator transmit frame-synchronization mode		R/W	0
			Used when FSXM = 1 in PCR.			
			FSGM = 0	The McBSP generates a transmit frame-synchronization pulse when the content of DXR[1,2] is copied to XSR[1,2].		
			FSGM = 1	The transmitter uses frame-synchronization pulses generated by the sample rate generator. Program the FWID bits to set the width of each pulse. Program the FPER bits to set the frame-synchronization period.		

Table 34-64 shows how FSXM and FSGM select the source of transmit frame-synchronization pulses. The three choices are:

- External frame-synchronization input
- Sample rate generator frame-synchronization signal (FSG)
- Internal signal that indicates a DXR-to-XSR copy has been made

Table 34-64 also shows the effect of each bit setting on the FSX pin. The polarity of the signal on the FSX pin is determined by the FSXP bit.

Table 34-64. How FSXM and FSGM Select the Source of Transmit Frame-Synchronization Pulses

FSXM	FSGM	Source of Transmit Frame Synchronization	FSX Pin Status
0	0 or 1	An external frame-synchronization signal enters the McBSP through the FSX pin. The signal is then inverted by FSXP before being used as internal FSX.	Input
1	1	Internal FSX is driven by the sample rate generator frame-synchronization signal (FSG).	Output. FSG is inverted by FSXP before being driven out on FSX pin.
1	0	A DXR-to-XSR copy causes the McBSP to generate a transmit frame-synchronization pulse that is 1 cycle wide.	Output. The generated frame-synchronization pulse is inverted as determined by FSXP before being driven out on FSX pin.

34.9.16.1 Other Considerations

If the sample rate generator creates a frame-synchronization signal (FSG) that is derived from an external input clock, the GSYNC bit determines whether FSG is kept synchronized with pulses on the FSR pin. For more details, see Section 34.4.3.

In the clock stop mode (CLKSTP = 10b or 11b), the McBSP can act as a master or as a slave in the SPI protocol. If the McBSP is a master and must provide a slave-enable signal ($\overline{\text{SPISTE}}$) on the FSX pin, make sure that FSXM = 1 and FSGM = 0 so that FSX is an output and is driven active for the duration of each transmission. If the McBSP is a slave, make sure that FSXM = 0 so that the McBSP can receive the slave-enable signal on the FSX pin.

34.9.17 Transmit Frame-Synchronization Polarity

Table 34-65 shows which register bits enable the Transmit Frame-Synchronization Polarity.

Table 34-65. Register Bit Used to Set Transmit Frame-Synchronization Polarity

Register	Bit	Name	Function		Type	Reset Value
PCR	3	FSXP	Transmit frame-synchronization polarity		R/W	0
			FSXP = 0	Frame-synchronization pulse FSX is active high.		
			FSXP = 1	Frame-synchronization pulse FSX is active low.		

34.9.17.1 Frame Synchronization Pulses, Clock Signals, and Their Polarities

Transmit frame-synchronization pulses can be generated internally by the sample rate generator (see Section 34.4.2) or driven by an external source. The source of frame synchronization is selected by programming the mode bit, FSXM, in PCR. FSX is also affected by the FSGM bit in SRGR2. For information about the effects of FSXM and FSGM, see Section 34.9.16). Similarly, transmit clocks can be selected to be inputs or outputs by programming the mode bit, CLKXM, in the PCR (see Section 34.9.19).

When FSR and FSX are inputs (FSXM = FSRM= 0, external frame-synchronization pulses), the McBSP detects them on the internal falling edge of clock, internal MCLKR, and internal CLKX, respectively. The receive data arriving at the DR pin is also sampled on the falling edge of internal MCLKR. These internal clock signals are either derived from external source via CLK(R/X) pins or driven by the sample rate generator clock (CLKG) internal to the McBSP.

When FSR and FSX are outputs, implying that the outputs are driven by the sample rate generator, the outputs are generated (transition to their active state) on the rising edge of internal clock, CLK(R/X). Similarly, data on the DX pin is output on the rising edge of internal CLKX.

FSRP, FSXP, CLKRP, and CLKXP in the pin control register (PCR) configure the polarities of the FSR, FSX, MCLKR, and CLKX signals, respectively. All frame-synchronization signals (internal FSR, internal FSX) that are internal to the serial port are active high. If the serial port is configured for external frame synchronization (FSR/FSX are inputs to McBSP) and FSRP = FSXP = 1, the external active-low frame-synchronization signals are inverted before being sent to the receiver (internal FSR) and transmitter (internal FSX). Similarly, if internal synchronization (FSR/FSX are output pins and GSYNC = 0) is selected and the polarity bit FS(R/X)P = 1, the internal active-high frame-synchronization signals are inverted before being sent to the FS(R/X) pin.

On the transmit side, the transmit clock polarity bit, CLKXP, sets the edge used to shift and clock out transmit data. Data is always transmitted on the rising edge of internal CLKX. If CLKXP = 1 and external clocking is selected (CLKXM = 0 and CLKX is an input), the external falling-edge triggered input clock on CLKX is inverted to a rising-edge triggered clock before being sent to the transmitter. If CLKXP = 1, and internal clocking selected (CLKXM = 1 and CLKX is an output pin), the internal (rising-edge triggered) clock, internal CLKX, is inverted before being sent out on the MCLKX pin.

Similarly, the receiver can reliably sample data that is clocked with a rising edge clock (by the transmitter). The receive clock polarity bit, CLKRP, sets the edge used to sample received data. The receive data is always sampled on the falling edge of internal MCLKR. Therefore, if CLKRP = 1 and external clocking is selected (CLKRM = 0 and MCLKR is an input pin), the external rising-edge triggered input clock on MCLKR is inverted to a falling-edge triggered clock before being sent to the receiver. If CLKRP = 1 and internal clocking is selected (CLKRM = 1), the internal falling-edge triggered clock is inverted to a rising-edge triggered clock before being sent out on the MCLKR pin.

CLKRP = CLKXP in a system where the same clock (internal or external) is used to clock the receiver and transmitter. The receiver uses the opposite edge as the transmitter to make sure of valid setup and hold of data around this edge. Figure 34-58 shows how data clocked by an external serial device using a rising edge can be sampled by the McBSP receiver on the falling edge of the same clock.

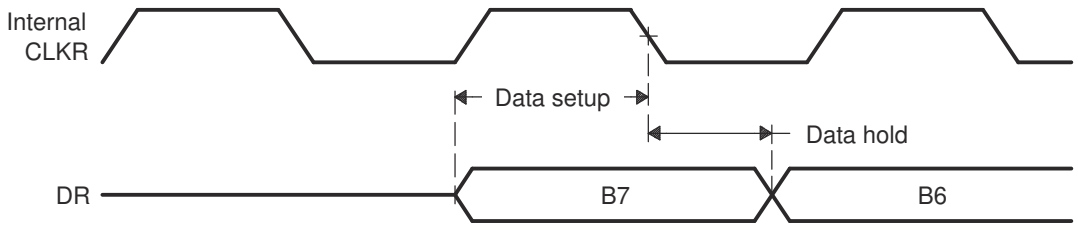


Figure 34-58. Data Clocked Externally Using a Rising Edge and Sampled by the McBSP Receiver on a Falling Edge

34.9.18 SRG Frame-Synchronization Period and Pulse Width

Table 34-66 shows which register bits set the SRG Frame-Synchronization Period and Pulse Width.

Table 34-66. Register Bits Used to Set SRG Frame-Synchronization Period and Pulse Width

Register	Bit	Name	Function	Type	Reset Value
SRGR2	11-0	FPER	Sample rate generator frame-synchronization period For the frame-synchronization signal FSG, (FPER + 1) determines the period from the start of a frame-synchronization pulse to the start of the next frame-synchronization pulse. Range for (FPER + 1): 1 to 4096 CLKG cycles.	R/W	0000 0000 0000
SRGR1	15-8	FWID	Sample rate generator frame-synchronization pulse width This field plus 1 determines the width of each frame-synchronization pulse on FSG. Range for (FWID + 1): 1 to 256 CLKG cycles.	R/W	0000 0000

34.9.18.1 Frame-Synchronization Period and Frame-Synchronization Pulse Width

The sample rate generator can produce a clock signal, CLKG, and a frame-synchronization signal, FSG. If the sample rate generator is supplying receive or transmit frame synchronization, you must program the bit fields FPER and FWID.

On FSG, the period from the start of a frame-synchronization pulse to the start of the next pulse is (FPER + 1) CLKG cycles. The 12 bits of FPER allow a frame-synchronization period of 1 to 4096 CLKG cycles, which allows up to 4096 data bits per frame. When GSYNC = 1, FPER is a don't care value.

Each pulse on FSG has a width of (FWID + 1) CLKG cycles. The eight bits of FWID allow a pulse width of 1 to 256 CLKG cycles. It is recommended that FWID be programmed to a value less than the programmed word length.

The values in FPER and FWID are loaded into separate down-counters. The 12-bit FPER counter counts down the generated clock cycles from the programmed value (4095 maximum) to 0. The 8-bit FWID counter counts down from the programmed value (255 maximum) to 0.

Figure 34-59 shows a frame-synchronization period of 16 CLKG periods (FPER = 15 or 00001111b) and a frame-synchronization pulse with an active width of 2 CLKG periods (FWID = 1).

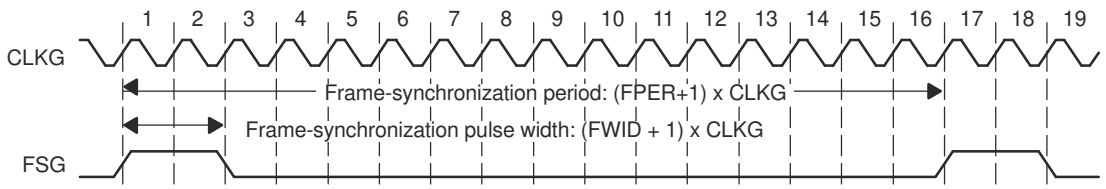


Figure 34-59. Frame of Period 16 CLKG Periods and Active Width of 2 CLKG Periods

When the sample rate generator comes out of reset, FSG is in the inactive state. Then, when GRST = 1 and FSGM = 1, a frame-synchronization pulse is generated. The frame width value (FWID + 1) is counted down on every CLKG cycle until the value reaches 0, at which time FSG goes low. At the same time, the frame period value (FPER + 1) is also counting down. When this value reaches 0, FSG goes high, indicating a new frame.

34.9.19 Transmit Clock Mode

Table 34-67 shows which register bits can set the Transmit Clock Mode.

Table 34-67. Register Bit Used to Set the Transmit Clock Mode

Register	Bit	Name	Function		Type	Reset Value
PCR	9	CLKXM	Transmit clock mode		R/W	0
			CLKXM = 0	The transmitter gets the clock signal from an external source via the MCLKX pin.		
			CLKXM = 1	The MCLKX pin is an output pin driven by the sample rate generator of the McBSP.		

34.9.19.1 Selecting a Source for the Transmit Clock and a Data Direction for the MCLKX pin

Table 34-68 shows how the CLKXM bit selects the transmit clock and the corresponding status of the MCLKX pin. The polarity of the signal on the MCLKX pin is determined by the CLKXP bit.

Table 34-68. How the CLKXM Bit Selects the Transmit Clock and the Corresponding Status of the MCLKX pin

CLKXM in PCR	Source of Transmit Clock	MCLKX pin Status
0	Internal CLKX is driven by an external clock on the MCLKX pin. CLKX is inverted as determined by CLKXP before being used.	Input
1	Internal CLKX is driven by the sample rate generator clock, CLKG.	Output. CLKG, inverted as determined by CLKXP, is driven out on CLKX.

34.9.19.2 Other Considerations

If the sample rate generator creates a clock signal (CLKG) that is derived from an external input clock, the GSYNC bit determines whether CLKG is kept synchronized with pulses on the FSR pin. For more details, see Section 34.4.3.

In the clock stop mode (CLKSTP = 10b or 11b), the McBSP can act as a master or as a slave in the SPI protocol. If the McBSP is a master, make sure that CLKXM = 1 so that CLKX is an output to supply the master clock to any slave devices. If the McBSP is a slave, make sure that CLKXM = 0 so that CLKX is an input to accept the master clock signal.

34.9.20 Transmit Clock Polarity

Table 34-69 shows which register bits set the Transmit Clock Polarity.

Table 34-69. Register Bit Used to Set Transmit Clock Polarity

Register	Bit	Name	Function		Type	Reset Value
PCR	1	CLKXP	Transmit clock polarity		R/W	0
			CLKXP = 0	Transmit data sampled on rising edge of CLKX.		
			CLKXP = 1	Transmit data sampled on falling edge of CLKX.		

34.9.20.1 Frame Synchronization Pulses, Clock Signals, and Their Polarities

Transmit frame-synchronization pulses can be either generated internally by the sample rate generator (see Section 34.4.2) or driven by an external source. The source of frame synchronization is selected by

programming the mode bit, FSXM, in PCR. FSX is also affected by the FSGM bit in SRGR2. For information about the effects of FSXM and FSGM, see Section 34.9.16). Similarly, transmit clocks can be selected to be inputs or outputs by programming the mode bit, CLKXM, in the PCR (see Section 34.9.19).

When FSR and FSX are inputs (FSXM = FSRM= 0, external frame-synchronization pulses), the McBSP detects them on the internal falling edge of clock, internal MCLKR, and internal CLKX, respectively. The receive data arriving at the DR pin is also sampled on the falling edge of internal MCLKR. These internal clock signals are either derived from external source via CLK(R/X) pins or driven by the sample rate generator clock (CLKG) internal to the McBSP.

When FSR and FSX are outputs, implying that the outputs are driven by the sample rate generator, the outputs are generated (transition to their active state) on the rising edge of internal clock, CLK(R/X). Similarly, data on the DX pin is output on the rising edge of internal CLKX.

FSRP, FSXP, CLKRP, and CLKXP in the pin control register (PCR) configure the polarities of the FSR, FSX, MCLKR, and CLKX signals, respectively. All frame-synchronization signals (internal FSR, internal FSX) that are internal to the serial port are active high. If the serial port is configured for external frame synchronization (FSR/FSX are inputs to McBSP), and FSRP = FSXP = 1, the external active-low frame-synchronization signals are inverted before being sent to the receiver (internal FSR) and transmitter (internal FSX). Similarly, if internal synchronization (FSR/FSX are output pins and GSYNC = 0) is selected, the internal active-high frame-synchronization signals are inverted, if the polarity bit FS(R/X)P = 1, before being sent to the FS(R/X) pin.

On the transmit side, the transmit clock polarity bit, CLKXP, sets the edge used to shift and clock out transmit data. Data is always transmitted on the rising edge of internal CLKX. If CLKXP = 1 and external clocking is selected (CLKXM = 0 and CLKX is an input), the external falling-edge triggered input clock on CLKX is inverted to a rising-edge triggered clock before being sent to the transmitter. If CLKXP = 1 and internal clocking is selected (CLKXM = 1 and CLKX is an output pin), the internal (rising-edge triggered) clock, internal CLKX, is inverted before being sent out on the MCLKX pin.

Similarly, the receiver can reliably sample data that is clocked with a rising edge clock (by the transmitter). The receive clock polarity bit, CLKRP, sets the edge used to sample received data. The receive data is always sampled on the falling edge of internal MCLKR. Therefore, if CLKRP = 1 and external clocking is selected (CLKRM = 0 and CLKR is an input pin), the external rising-edge triggered input clock on CLKR is inverted to a falling-edge triggered clock before being sent to the receiver. If CLKRP = 1 and internal clocking is selected (CLKRM = 1), the internal falling-edge triggered clock is inverted to a rising-edge triggered clock before being sent out on the MCLKR pin.

CLKRP = CLKXP in a system where the same clock (internal or external) is used to clock the receiver and transmitter. The receiver uses the opposite edge as the transmitter to make sure of valid setup and hold of data around this edge (see Figure 34-58).

Figure 34-60 shows how data clocked by an external serial device using a rising edge can be sampled by the McBSP receiver on the falling edge of the same clock.

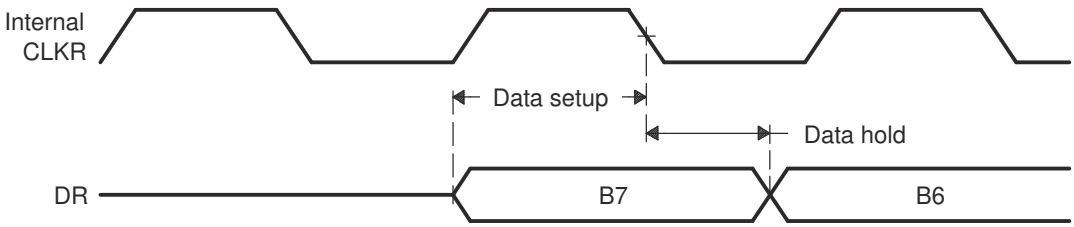


Figure 34-60. Data Clocked Externally Using a Rising Edge and Sampled by the McBSP Receiver on a Falling Edge

34.10 Emulation and Reset Considerations

This section covers the following topics:

- How to program McBSP response to a breakpoint in the high-level language debugger (see Section 34.10.1)
- How to reset and initialize the various parts of the McBSP (see Section 34.10.2)

34.10.1 McBSP Emulation Mode

FREE and SOFT are special emulation bits in SPCR2 that determine the state of the McBSP when a breakpoint is encountered in the high-level language debugger. If FREE = 1, the clock continues to run upon a software breakpoint and data is still shifted out. When FREE = 1, the SOFT bit is a *don't care.*

If FREE = 0, the SOFT bit takes effect. If SOFT = 0 when breakpoint occurs, the clock stops immediately, aborting a transmission. If SOFT = 1 and a breakpoint occurs while transmission is in progress, the transmission continues until completion of the transfer and then the clock halts. These options are listed in Table 34-70.

The McBSP receiver functions in a similar fashion. If a mode other than the immediate stop mode (SOFT = FREE = 0) is chosen, the receiver continues running and an overrun error is possible.

Table 34-70. McBSP Emulation Modes Selectable with FREE and SOFT Bits of SPCR2

FREE	SOFT	McBSP Emulation Mode
0	0	Immediate stop mode (reset condition). The transmitter or receiver stops immediately in response to a breakpoint.
0	1	Soft stop mode When a breakpoint occurs, the transmitter stops after completion of the current word. The receiver is not affected.
1	0 or 1	Free run mode The transmitter and receiver continue to run when a breakpoint occurs.

34.10.2 Resetting and Initializing McBSPs

This section discusses in greater depth the MCBSP Reset and Initialization configurations.

34.10.2.1 McBSP Pin States: DSP Reset Versus Receiver/Transmitter Reset

Table 34-71 shows the state of McBSP pins when the serial port is reset due to direct receiver or transmitter reset on the 2833x device.

Table 34-71. Reset State of Each McBSP Pin

Pin	Possible States[1]	State Forced by Device Reset	State Forced by Receiver/Transmitter Reset
Receiver reset (RRST = 0 and GRST = 1)			
MDRx	I	GPIO-input	Input
MCLKRx	I/O/Z	GPIO-input	Known state if input; MCLKR running if output
MFSRx	I/O/Z	GPIO-input	Known state if input; FSRP inactive state if output
Transmitter reset (XRST = 0 and GRST = 1)			
MDXx	O/Z	GPIO Input	High impedance
MCLKXx	I/O/Z	GPIO-input	Known state if input; CLKX running if output
MFSXx	I/O/Z	GPIO-input	Known state if input; FSXP inactive state if output

(1) In Possible States column, I = Input, O = Output, Z = High impedance. In the C28x family, at device reset, all I/Os default to GPIO function and generally as inputs.

34.10.2.2 Device Reset, McBSP Reset, and Sample Rate Generator Reset

When the McBSP is reset in either of the above two ways, the machine is reset to its initial state, including reset of all counters and status bits. The receive status bits include RFULL, RRDY, and RSYNCERR. The transmit status bits include XEMPTY, XRDY, and XSYNCERR.

- Device reset. When the whole DSP is reset ($\overline{\text{XRS}}$ signal is driven low), all McBSP pins are in GPIO mode. When the device is pulled out of reset, the clock to the McBSP modules remains disabled.
- McBSP reset. When the receiver and transmitter reset bits, RRST and XRST, are loaded with 0s, the respective portions of the McBSP are reset and activity in the corresponding section of the serial port stops.

Input-only pins such as MDRx, and all other pins that are configured as inputs are in a known state. The MFSRx and MFSXx pins are driven to their inactive state if they are not outputs. If the MCLKR and MCLKX pins are programmed as outputs, they are driven by CLKG, provided that GRST = 1. Lastly, the MDXx pin is in the high-impedance state when the transmitter and/or the device is reset.

During normal operation, the sample rate generator is reset if the GRST bit is cleared. GRST must be 0 only when neither the transmitter nor the receiver is using the sample rate generator. In this case, the internal sample rate generator clock (CLKG) and its frame-synchronization signal (FSG) are driven inactive low.

When the sample rate generator is not in the reset state (GRST = 1), pins MFSRx and MFSXx are in an inactive state when RRST = 0 and XRST = 0, respectively, even if they are outputs driven by FSG. This ensures that when only one portion of the McBSP is in reset, the other portion can continue operation when GRST = 1 and its frame synchronization is driven by FSG.

- Sample rate generator reset. The sample rate generator is reset when GRST is loaded with 0.

  When neither the transmitter nor the receiver is fed by CLKG and FSG, you can reset the sample rate generator by clearing GRST. In this case, CLKG and FSG are driven inactive low. If you then set GRST, CLKG starts and runs as programmed. Later, if GRST = 1, FSG pulses active high after the programmed number of CLKG cycles has elapsed.

34.10.2.3 McBSP Initialization Procedure

The serial port initialization procedure is as follows:

1. Make XRST = RRST = GRST = 0 in SPCR[1,2]. If coming out of a device reset, this step is not required.
2. While the serial port is in the reset state, program only the McBSP configuration registers (not the data registers) as required.
3. Wait for two clock cycles. This ensures proper internal synchronization.
4. Set up data acquisition as required (such as writing to DXR[1,2]).
5. Make XRST = RRST = 1 to enable the serial port. Make sure that as you set these reset bits, you do not modify any of the other bits in SPCR1 and SPCR2. Otherwise, you change the configuration you selected in step 2.
6. Set FRST = 1, if internally generated frame synchronization is required.
7. Wait two clock cycles for the receiver and transmitter to become active.

Alternatively, on either write (step 1 or 5), the transmitter and receiver can be placed in or taken out of reset individually by modifying the desired bit.

The above procedure for reset/initialization can be applied in general when the receiver or transmitter must be reset during its normal operation and when the sample rate generator is not used for either operation.

Note

1. The necessary duration of the active-low period of XRST or RRST is at least two MCLKR/CLKX cycles.
2. The appropriate bits in serial port configuration registers SPCR[1,2], PCR, RCR[1,2], XCR[1,2], and SRGR[1,2] must only be modified when the affected portion of the serial port is in its reset state.
3. In most cases, the data transmit registers (DXR[1,2]) must be loaded by the CPU or by the DMA controller only when the transmitter is enabled (XRST = 1). An exception to this rule is when these registers are used for companding internal data (see Section 34.3.2.2).
4. The bits of the channel control registers—MCR[1,2], RCER[A-H], XCER[A-H]—can be modified at any time as long as they are not being used by the current reception/transmission in a multichannel selection mode.

34.10.2.4 Resetting the Transmitter While the Receiver is Running

Example 34-1 shows values in the control registers that reset and configure the transmitter while the receiver is running.

Example 34-1. Resetting and Configuring McBSP Transmitter While McBSP Receiver Running

```
SPCR1 = 0001h SPCR2 = 0030h ;
The receiver is running with the receive interrupt (RINT) triggered by the receiver ready bit
(RRDY).;
The transmitter is in its reset state.;
The transmit interrupt (XINT) will be triggered by the transmit frame-sync error bit (XSYNCERR).;
PCR = 0900h ;
Transmit frame synchronization is generated internally according to the FSGM bit of SRGR2.;
The transmit clock is driven by an external source. ;
The receive clock continues to be driven by sample rate generator. The input clock ;
of the sample rate generator is supplied by the CPU clock. ;
SRGR1 = 0001h SRGR2 = 2000h ;
The CPU clock is the input clock for the sample rate generator. The sample ;
rate generator divides the CPU clock by 2 to generate its output clock (CLKG). ;
Transmit frame synchronization is tied to the automatic copying of data from ;
the DXR(s) to the XSR(s). ;
XCR1 = 0740h XCR2 = 8321h ;
The transmit frame has two phases. Phase 1 has eight 16-bit words. ;
Phase 2 has four 12-bit words. There is 1-bit data delay between the start of a ;
frame-sync pulse and the first data bit transmitted. ;
SPCR2 = 0031h ;
The transmitter is taken out of reset.
```

34.11 Data Packing Examples

This section shows two ways to implement data packing in the McBSP.

34.11.1 Data Packing Using Frame Length and Word Length

Frame length and word length can be manipulated to effectively pack data. For example, consider a situation where four 8-bit words are transferred in a single-phase frame as shown in Figure 34-61. In this case:

- (R/X)PHASE = 0: Single-phase frame
- (R/X)FRLEN1 = 0000011b: 4-word frame
- (R/X)WDLEN1 = 000b: 8-bit words

Four 8-bit data words are transferred to and from the McBSP by the CPU or by the DMA controller. Thus, four reads from DRR1 and four writes to DXR1 are necessary for each frame.

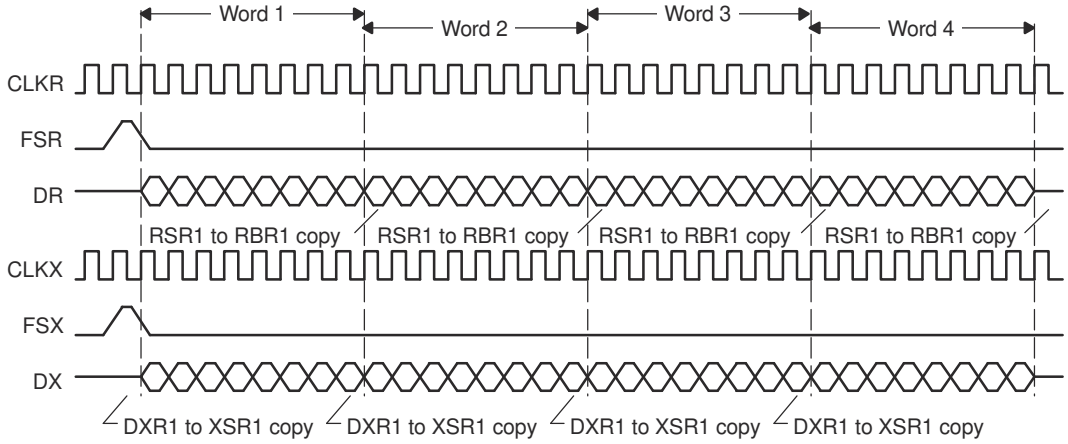


Figure 34-61. Four 8-Bit Data Words Transferred To/From the McBSP

This data can also be treated as a single-phase frame consisting of one 32-bit data word, as shown in Figure 34-62. In this case:

- (R/X)PHASE = 0: Single-phase frame
- (R/X)FRLEN1 = 0000000b: 1-word frame
- (R/X)WDLEN1 = 101b: 32-bit word

Two 16-bit data words are transferred to and from the McBSP by the CPU or DMA controller. Thus, two reads, from DRR2 and DRR1, and two writes, to DXR2 and DXR1, are necessary for each frame. This results in only half the number of transfers compared to the previous case. This manipulation reduces the percentage of bus time required for serial port data movement.

Note

When the word length is larger than 16 bits, make sure you access DRR2/DXR2 before you access DRR1/DXR1. McBSP activity is tied to accesses of DRR1/DXR1. During the reception of 24-bit or 32-bit words, read DRR2 and then read DRR1. Otherwise, the next RBR[1,2]-to-DRR[1,2] copy occurs before DRR2 is read. Similarly, during the transmission of 24-bit or 32-bit words, write to DXR2 and then write to DXR1. Otherwise, the next DXR[1,2]-to-XSR[1,2] copy occurs before DXR2 is loaded with new data.

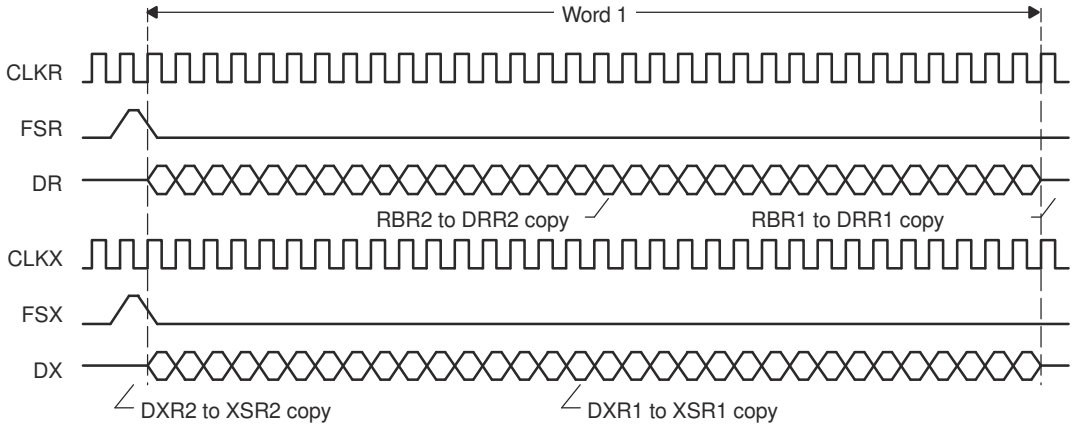


Figure 34-62. One 32-Bit Data Word Transferred To/From the McBSP

34.11.2 Data Packing Using Word Length and the Frame-Synchronization Ignore Function

When there are multiple words per frame, you can implement data packing by increasing the word length (defining a serial word with more bits) and by ignoring frame-synchronization pulses. First, consider Figure 34-63, which shows the McBSP operating at the maximum packet frequency. Here, each frame only has a single 8-bit word. Notice the frame-synchronization pulse that initiates each frame transfer for reception and for transmission. For reception, this configuration requires one read operation for each word. For transmission, this configuration requires one write operation for each word.

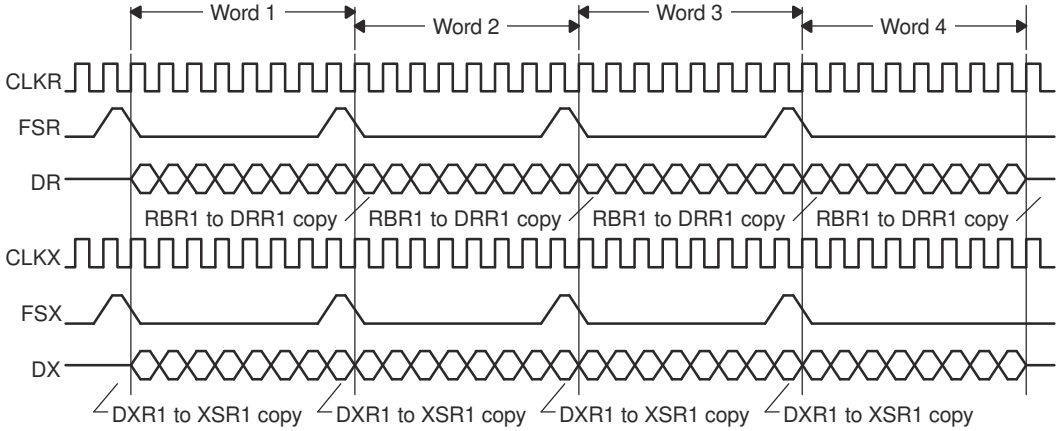


Figure 34-63. 8-Bit Data Words Transferred at Maximum Packet Frequency

Figure 34-64 shows the McBSP configured to treat this data stream as a continuous 32-bit word. In this example, the McBSP responds to an initial frame-synchronization pulse. However, (R/X)FIG = 1 so that the McBSP ignores subsequent pulses. Only two read transfers or two write transfers are needed every 32 bits. This configuration effectively reduces the required bus bandwidth to half the bandwidth needed to transfer four 8-bit words.

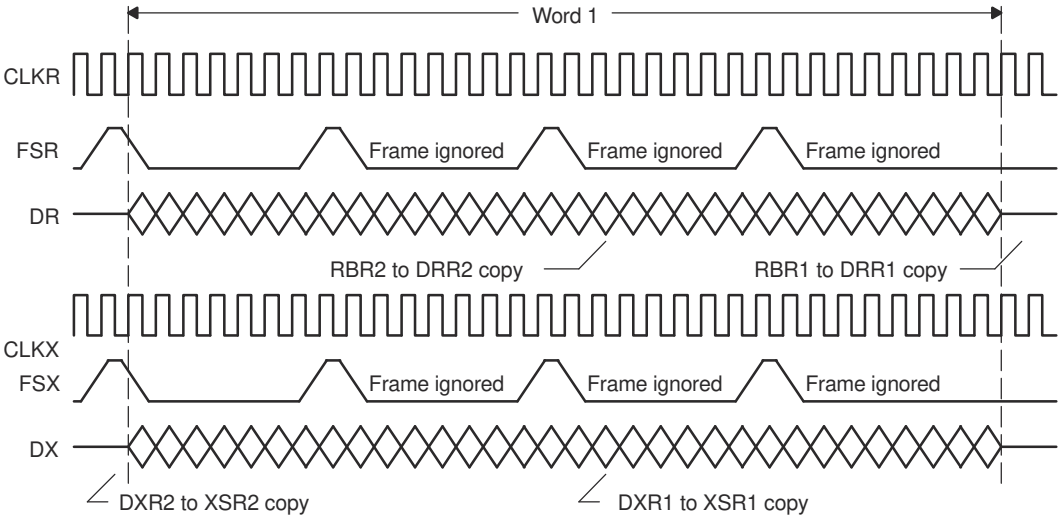


Figure 34-64. Configuring the Data Stream of Figure 34-63 as a Continuous 32-Bit Word

34.12 Interrupt Generation

McBSP registers can be programmed to receive and transmit data through DRR2/DRR1 and DXR2/DXR1 registers, respectively. The CPU can directly access these registers to move data from memory to these registers. Interrupt signals are based on these register pair contents and the related flags. MRINT/MXINT generates CPU interrupts for receive and transmit conditions.

34.12.1 McBSP Receive Interrupt Generation

In the McBSP module, data receive and error conditions generate two sets of interrupt signals. One set is used for the CPU and the other set is for DMA.

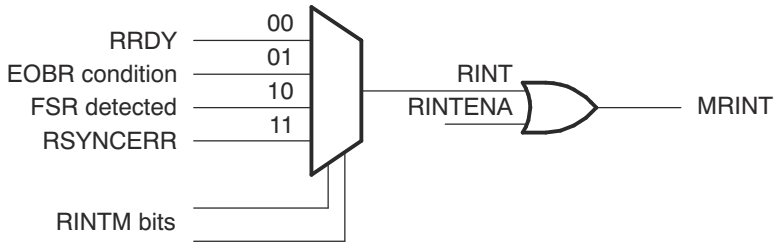


Figure 34-65. Receive Interrupt Generation

Table 34-72. Receive Interrupt Sources and Signals

McBSP Interrupt Signal	Interrupt Flags	Interrupt Enables in SPCR1 (RINTM Bits)	Interrupt Enables	Type of Interrupt	Interrupt Line
RINT	RRDY	00	RINTENA	Every word receive	MRINT
	EOBR	01	RINTENA	Every 16-channel block boundary	
	FSR	10	RINTENA	On every FSR	
	RSYNCERR	11	RINTENA	Frame sync error	

Note

Since X/RINT, X/REVTA, and X/RXFFINT share the same CPU interrupt, it is recommended that all applications use one of the above selections for interrupt generation. If multiple interrupt enables are selected at the same time, there is a likelihood of interrupts being masked or not recognized.

34.12.2 McBSP Transmit Interrupt Generation

McBSP module data transmit and error conditions generate two sets of interrupt signals. One set is used for the CPU and the other set is for DMA.

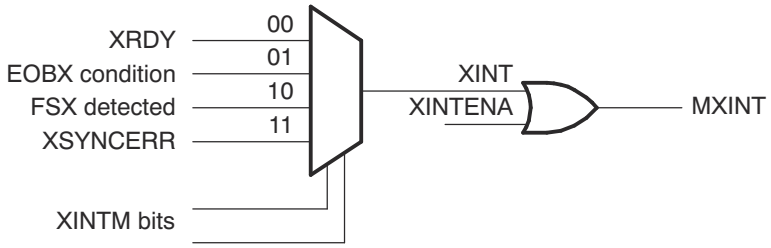


Figure 34-66. Transmit Interrupt Generation

Table 34-73. Transmit Interrupt Sources and Signals

McBSP Interrupt Signal	Interrupt Flags	Interrupt Enables in SPCR2 (XINTM Bits)	Interrupt Enables	Type of Interrupt	Interrupt Line
XINT	XRDY	00	XINTENA	Every word transmit	MXINT
	EOBX	01	XINTENA	Every 16-channel block boundary	
	FSX	10	XINTENA	On every FSX	
	XSYNCERR	11	XINTENA	Frame sync error	

34.12.3 Error Flags

The McBSP has several error flags both on receive and transmit channel. Table 34-74 explains the error flags and their meaning.

Table 34-74. Error Flags

Error Flags	Function
RFULL	Indicates DRR2/DRR1 are not read and RXR register is overwritten
RSYNCERR	Indicates unexpected frame-sync condition, current data reception will abort and restart. Use RINTM bit 11 for interrupt generation on this condition.
XSYNCERR	Indicates unexpected frame-sync condition, current data transmission will abort and restart. Use XINTM bit 11 for interrupt generation on this condition.

34.13 McBSP Modes

McBSP, in its normal mode, communicates with various types of Codecs with variable word size. Apart from this mode, the McBSP uses time-division multiplexed (TDM) data stream while communicating with other McBSPs or serial devices. The multichannel mode provides flexibility while transmitting/receiving selected channels or all the channels in a TDM stream.

Table 34-75 provides a quick reference to McBSP mode selection.

Table 34-75. McBSP Mode Selection

No.	McBSP Word Size	Register Bits Used for Mode Selection				Mode and Function Description
		MCR1 bit 9,0		MCR2 bit 9,1,0		
		RMCME	RMCM	XMCME	XMCM	
						Normal Mode
1	8/12/16/20/24/32 bit words	0	0	0	0	All types of Codec interface will use this selection
						Multichannel Mode
2	8-bit words					2 Partition or 32-channel Mode
		0	1	0	1	All channels are disabled, unless selected in X/RCERA/B
		0	1	0	10	All channels are enabled,but masked unless selected in X/RCERA/B
		0	1	0	11	Symmetric transmit, receive
						8 Partition or 128 Channel Mode Transmit/Receive
						Channels selected by X/RCERA to X/RCERH bits
						Multichannel Mode is ON
		1	1	1	1	All channels are disabled, unless selected in XCERs
		1	1	1	10	All channels are enabled, but masked unless selected in XCERs
		1	1	1	11	Symmetric transmit, receive
						Continuous mode, transmit
		1	0	1	0	**Multi-Channel Mode is OFF**
						All 128 channels are active and enabled

34.14 Special Case: External Device is the Transmit Frame Master

Care must be taken if the transmitter expects a frame sync from an external device. After the transmitter comes out of reset (XRST = 1), the transmitter waits for a frame sync from the external device. If the first frame sync arrives very shortly after the transmitter is enabled, the CPU or DMA controller cannot have a chance to service DXR. In this case, the transmitter shifts out the default data in XSR instead of the desired value, which has not yet arrived in DXR. This causes problems in some applications, as the first data element in the frame is invalid. The data stream appears element-shifted (the first data word can appear in the second channel instead of the first).

To make sure of proper operation when the external device is the frame master, you must verify that DXR is already serviced with the first word when a frame sync occurs. To do so, you can keep the transmitter in reset until the first frame sync is detected. Upon detection of the first frame sync, the McBSP generates an interrupt to the CPU. Within the interrupt service routine, the transmitter is taken out of reset (XRST = 1). This verifies that the transmitter does not begin data transfers at the data pin during the first frame sync period. This also provides almost an entire frame period for the DSP to service DXR with the first word before the second frame sync occurs. The transmitter only begins data transfers upon receiving the second frame sync. At this point, DXR is already serviced with the first word.

The interrupt service routine must first be setup, then follow this modified procedure for proper initialization:

1. Make sure that no portion of the McBSP is using the internal sample rate generator signal CLKG and the internal frame sync generator signal FSG (GRST = FRST = 0). The respective portion of the McBSP needs to be in reset (XRST = 0 and/or RRST = 0).
2. Program SRGR and other control registers as required. Make sure the internal sample rate generator and the internal frame sync generator are still in reset (GRST = FRST = 0). Also make sure the respective portion of the McBSP is still in reset in this step (XRST = 0 and/or RRST = 0).
3. Program the XINTM bits to 2h in SPCR to generate an interrupt to the CPU upon detection of a transmit frame sync. Do not enable the XINT interrupt in the interrupt enable register (IER) in this step.
4. Wait for proper internal synchronization. If the external device provides the bit clock, wait for two CLKR or CLKX cycles. If the McBSP generates the bit clock as a master clock, wait for two CLKSRG cycles. In this case, the clock source to the sample rate generator (CLKSRG) is selected by the CLKSM bit in SRGR.
5. Skip this step if the bit clock is provided by the external device. This step only applies if the McBSP is the bit master clock and the internal sample rate generator is used.
   a. Start the sample rate generator by setting the GRST bit to 1. Wait two CLKG bit clocks for synchronization. CLKG is the output of the sample rate generator.
   b. On the next rising edge of CLKSRG, CLKG transitions to 1 and starts clocking with a frequency equal to 1/(CLKGDV + 1) of the sample rate generator source clock CLKSRG.
6. A transmit sync error (XSYNCERR) can occur when enabled for the first time after device reset. The purpose of this step is to clear any potential XSYNCERR that occurs on the transmitter at this time:
   a. Set the XRST bit to 1 to enable the transmitter.
   b. Wait for any unexpected frame sync error to occur. If the external device provides the bit clock, wait for two CLKR or CLKX cycles. If the McBSP generates the bit clock as a master clock, wait for two CLKG cycles. The unexpected frame sync error (XSYNCERR), if any, occurs within this time period.
   c. Disable the transmitter (XRST = 0). This clears any outstanding XSYNCERR.
7. Setup data acquisition as required:
   a. If the DMA controller is used to service the McBSP, setup data acquisition as desired and start the DMA controller in this step, before the McBSP is taken out of reset.
   b. If CPU interrupt is used to service the McBSP, no action is required in this step.
   c. If CPU polling is used to service the McBSP, no action is required in this step.
8. Enable the XINT interrupt by setting the corresponding bit in the interrupt enable register (IER). In this step, the McBSP transmitter is still in reset. Upon detection of the first transmit frame sync from the external device, the McBSP generates an interrupt to the CPU and the DSP enters the interrupt service routine (ISR). The ISR needs to perform these tasks in this order:
   a. Modify the XINTM bits to the value desired for normal McBSP operations. If CPU interrupt is used to service the McBSP in normal operations, make sure that the XINTM bits are modified to 0 to detect the McBSP XRDY event. If no McBSP interrupt is desired in normal operations, disable future McBSP-to-CPU interrupt in the interrupt enable register (IER).
   b. Set the XRST bit and/or the RRST bit to 1 to enable the respective portion of the McBSP. The McBSP is now ready to transmit and/or receive.
9. Service the McBSP:
   a. If CPU polling is used to service the McBSP in normal operations, CPU polling can do so upon exit from the ISR.
   b. If CPU interrupt is used to service the McBSP in normal operations, upon XRDY interrupt service routine is entered. The ISR must be setup to verify that XRDY = 1 and service the McBSP accordingly.
   c. If DMA controller is used to service the McBSP in normal operations, the DMA controller services the McBSP automatically upon receiving the XEVT and/or REVT.
10. Upon detection of the second frame sync, DXR is already serviced and the transmitter is ready to transmit the valid data. The receiver is also serviced properly by the DSP.

34.15 Software

34.15.1 MCBSP Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/mcbsp

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

34.15.1.1 Pin Setup for McBSP module

FILE: mcbsp_pin_setup.c This file contains functions to configures pins for McBSPA and McBSPB.

34.15.1.2 McBSP loopback example

FILE: mcbsp_ex1_loopback.c

This example demonstrates the McBSP operation using internal loopback. This example does not use interrupts. Instead, a polling method is used to check the receive data. The incoming data is checked for accuracy.

Three different serial word sizes can be tested. Before compiling this project, select the serial word size of 8, 16 or 32 by using the #define statements at the beginning of the code.

This program will execute until terminated by the user.

8-bit word example:
The sent data looks like this:
00 01 02 03 04 05 06 07 .... FE FF
16-bit word example:
The sent data looks like this:
0000 0001 0002 0003 0004 0005 0006 0007 .... FFFE FFFF
32-bit word example:
The sent data looks like this:
FFFF0000 FFFE0001 FFFD0002 .... 0000FFFF
External Connections

- None

Watch Variables:

- *txData1* - Sent data word: 8 or 16-bit or low half of 32-bit
- *txData2* - Sent data word: upper half of 32-bit
- *rxData1* - Received data word: 8 or 16-bit or low half of 32-bit
- *rxData2* - Received data word: upper half of 32-bit
- *errCountGlobal* - Error counter

txData2 and rxData2 are not used for 8-bit or 16-bit word size.

34.15.1.3 McBSP loopback with DMA example.

FILE: mcbsp_ex2_loopback_dma.c

This example demonstrates the McBSP operation using internal loopback and utilizes the DMA to transfer data from one buffer to the McBSP and then from McBSP to another buffer.

Initially, txData[] is filled with values from 0x0000- 0x007F. The DMA moves the values in txData[] one by one to the DXRx registers of the McBSP. These values are transmitted and subsequently received by the McBSP. Then, the the DMA moves each data value to rxData[] as it is received by the McBSP.

The sent data buffer looks like this:
0000 0001 0002 0003 0004 0005 .... 007F
Three different serial word sizes can be tested. Before compiling this project, select the serial word size of 8, 16 or 32 by using the #define statements at the beginning of the code.

This example uses DMA channel 1 and 2 interrupts. The incoming data is checked for accuracy.

External Connections
- None

Watch Variables:
- *txData* - Sent data buffer
- *rxData* - Received data buffer
- *errCountGlobal* - Error counter

34.15.1.4 McBSP loopback with interrupts example

FILE: mcbsp_ex3_loopback_interrupts.c

This example demonstrates the McBSP operation using internal loopback. This example uses interrupts. Both Rx and Tx interrupts are enabled.

External Connections
- None

Watch Variables:
- *txData* - Sent data word
- *rxData* - Received data word
- *errCountGlobal* - Error counter

34.15.1.5 McBSP loopback with interrupts example

FILE: mcbsp_ex3_loopback_interrupts_sysconfig.c

This example demonstrates the McBSP operation using internal loopback. This example uses interrupts. Both Rx and Tx interrupts are enabled.

External Connections
- None

Watch Variables:
- *txData* - Sent data word
- *rxData* - Received data word
- *errCountGlobal* - Error counter

34.15.1.6 McBSP loopback example using SPI mode

FILE: mcbsp_ex4_spi_loopback.c

This example demonstrates the McBSP operation in SPI mode using internal loopback. This example demonstrates SPI master mode transfer of 32-bit word size with digital loopback enabled.

McBSP Signals - SPI equivalent
- MCLKX - SPICLK
- MFSX - SPISTE
- MDX - SPISIMO
- MDR - SPISOMI (not used for this example)

External Connections
- None

Watch Variables:
- *txData1* - Sent data word: 8 or 16-bit or low half of 32-bit
- *txData2* - Sent data word: upper half of 32-bit
- *rxData1* - Received data word: 8 or 16-bit or low half of 32-bit
- *rxData2* - Received data word: upper half of 32-bit
- *errCountGlobal* - Error counter

34.15.1.7 McBSP external loopback example

FILE: mcbsp_ex5_ext_loopback.c

This example demonstrates the McBSP operation using external loopback. This example does not use interrupts. Instead, a polling method is used to check the receive data. The incoming data is checked for accuracy.

Three different serial word sizes can be tested. Before compiling this project, select the serial word size of 8, 16 or 32 by using the #define statements at the beginning of the code.

This program will execute until terminated by the user.

8-bit word example:
The sent data looks like this:
00 01 02 03 04 05 06 07 .... FE FF
16-bit word example:
The sent data looks like this:
0000 0001 0002 0003 0004 0005 0006 0007 .... FFFE FFFF
32-bit word example:
The sent data looks like this:
FFFF0000 FFFE0001 FFFD0002 .... 0000FFFF
External Connections
McBSPA Signals - McBSPB signals

- MCLKXA - MCLKRB
- MFSXA - MFSRB
- MDXA - MDRB
- MCLKRA - MCLKXB
- MFSRA - MFSXB
- MDRA - MDXB

Watch Variables:

- *txData1A* - Sent data word by McBSPA Transmitter:8 or 16-bit or low half of 32-bit
- *txData2A* - Sent data word by McBSPA Transmitter:upper half of 32-bit
- *rxData1A* - Received data word by MCBSPA Receiver:8 or 16-bit or lower half of 32-bit
- *rxData2A* - Received data word by McBSPA Receiver:upper half of 32-bit
- *txData1B* - Sent data word by McBSPB Transmitter:8 or 16-bit or low half of 32-bit
- *txData2B* - Sent data word by McBSPB Transmitter:upper half of 32-bit
- *rxData1B* - Received data word by McBSPB Receiver:8 or 16-bit or lower half of 32-bit
- *rxData2B* - Received data word by McBSPB Receiver:upper half of 32-bit
- *errCountGlobal* - Error counter

txData2A, rxData2A, txData2B and rxData2B are not used for 8-bit or 16-bit word size.

34.15.1.8 McBSP external loopback example using SPI mode

FILE: mcbsp_ex6_spi_ext_loopback.c

This example demonstrates the McBSP operation in SPI mode using external loopback. This example configures McBSP instances available on the device as SPI master and slave and demonstrates transfer of 32-bit word size data with external loopback.

External Connections
SPI Master(McBSPA) *SPI* Slave(McBSPB)

- MCLKXA(SPICLK) (GPIO22) - MCLKXB(SPICLK) (GPIO26)
- MFSXA (SPISTE) (GPIO23) - MFSXB (SPISTE) (GPIO27)
- MDXA (SPISIMO)(GPIO20) - MDRB (SPISIMO)(GPIO25)
- MDRA (SPISOMI)(GPIO21) - MDXB (SPISOMI)(GPIO24)

Watch Variables:

- *txData1* - Sent data word: 8 or 16-bit or low half of 32-bit
- *txData2* - Sent data word: upper half of 32-bit
- *rxData1* - Received data word: 8 or 16-bit or low half of 32-bit
- *rxData2* - Received data word: upper half of 32-bit
- *errCountGlobal* - Error counter

34.15.1.9 McBSP TDM-8 Test

FILE: mcbsp_ex7_tdm8_test.c

For the detailed description of this example, please refer to: How to Implement Custom Serial Interfaces Using the Configurable Logic Block (CLB) Application Note (SPRAD62).

In this example a MCBSP is used to generate and receive a TDM-8 test stream. This example uses interrupts. Both RX and TX interrupts are enabled. The McBSP TDM stream is set to eight 32-bit channels per frame.

This example is specifically created for use with SPRAD62. To use the McBSP inputs and outputs, the following connections are needed:

External Connections
McBSP Output Pins GPIO pin Device Under Test (DUT) MCLKX GPIO22 BCLK_IN MFSX GPIO23 FSYNC_IN MDX GPIO20 DATA1_IN

McBSP Input Pins GPIO pin Device Under Test (DUT) MCLKR GPIO58 BCLK_OUT FSR GPIO59 FSYNC_OUT MDR GPIO21 DATA1_OUT

The McBSP TX and RX pins can be externally looped back to create self contained test. *Watch Variables:*
- *txData* - Sent data word by McBSP Transmitter
- *rxData* - Received data word by McBSP Receiver
- *testWordDetected* - Indicates when test has started
- *errCountGlobal* - Number of errors detected

34.16 McBSP Registers

This section describes the multichannel buffered serial port (McBSP) registers.

34.16.1 MCBSP Base Address Table (C28)

Table 34-76. MCBSP Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
McbspaRegs	McBSP_REGS	MCBSPA_BASE	0x0000_6000	YES	YES	YES	YES	YES
McbspbRegs	McBSP_REGS	MCBSPB_BASE	0x0000_6040	YES	YES	YES	YES	YES

34.16.2 McBSP_REGS Registers

Table 34-77 lists the memory-mapped registers for the McBSP_REGS registers. All register offset addresses not listed in Table 34-77 should be considered as reserved locations and the register contents should not be modified.

Table 34-77. MCBSP_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	DRR2	Data receive register bits 31-16		Go
1h	DRR1	Data receive register bits 15-0		Go
2h	DXR2	Data transmit register bits 31-16		Go
3h	DXR1	Data transmit register bits 15-0		Go
4h	SPCR2	Serial port control register 2		Go
5h	SPCR1	Serial port control register 1		Go
6h	RCR2	Receive Control register 2		Go
7h	RCR1	Receive Control register 1		Go
8h	XCR2	Transmit Control register 2		Go
9h	XCR1	Transmit Control register 1		Go
Ah	SRGR2	Sample rate generator register 2		Go
Bh	SRGR1	Sample rate generator register 1		Go
Ch	MCR2	Multi-channel control register 2		Go
Dh	MCR1	Multi-channel control register 1		Go
Eh	RCERA	Receive channel enable partition A		Go
Fh	RCERB	Receive channel enable partition B		Go
10h	XCERA	Transmit channel enable partition A		Go
11h	XCERB	Transmit channel enable partition B		Go
12h	PCR	Pin Control register		Go
13h	RCERC	Receive channel enable partition C		Go
14h	RCERD	Receive channel enable partition D		Go
15h	XCERC	Transmit channel enable partition C		Go
16h	XCERD	Transmit channel enable partition D		Go
17h	RCERE	Receive channel enable partition E		Go
18h	RCERF	Receive channel enable partition F		Go
19h	XCERE	Transmit channel enable partition E		Go
1Ah	XCERF	Transmit channel enable partition F		Go
1Bh	RCERG	Receive channel enable partition G		Go
1Ch	RCERH	Receive channel enable partition H		Go
1Dh	XCERG	Transmit channel enable partition G		Go
1Eh	XCERH	Transmit channel enable partition H		Go
23h	MFFINT	Interrupt enable		Go

Complex bit access types are encoded to fit into small table cells. Table 34-78 shows the codes that are used for access types in this section.

Table 34-78. McBSP_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write

Table 34-78. McBSP_REGS Access Type Codes (continued)

Access Type	Code	Description
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

34.16.2.1 DRR2 Register (Offset = 0h) [Reset = 0h]

DRR2 is shown in Figure 34-67 and described in Table 34-79.

Return to the Summary Table.

DRR2 contains the upper 16 bits of the received data to be read by the CPU or DMA. DRR2 is only used if the word length is greater than 16 bits.

Figure 34-67. DRR2 Register

15	14	13	12	11	10	9	8
HWHB							
R/W-0h							
7	6	5	4	3	2	1	0
HWLB							
R/W-0h							

Table 34-79. DRR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	HWHB	R/W	0h	High word high byte Reset type: SYSRSn
7-0	HWLB	R/W	0h	High word low byte Reset type: SYSRSn

34.16.2.2 DRR1 Register (Offset = 1h) [Reset = 0h]

DRR1 is shown in Figure 34-68 and described in Table 34-80.

Return to the Summary Table.

DRR1 contains the lower 16 bits of the received data to be read by either the CPU or DMA.

Figure 34-68. DRR1 Register

15	14	13	12	11	10	9	8
LWHB							
R/W-0h							
7	6	5	4	3	2	1	0
LWLB							
R/W-0h							

Table 34-80. DRR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	LWHB	R/W	0h	Low word high byte Reset type: SYSRSn
7-0	LWLB	R/W	0h	Low word low byte Reset type: SYSRSn

34.16.2.3 DXR2 Register (Offset = 2h) [Reset = 0h]

DXR2 is shown in Figure 34-69 and described in Table 34-81.

Return to the Summary Table.

DXR2 contains the upper 16 bits of the data to be transmitted after being written by the CPU or DMA. DXR2 is only used if the word length is greater than 16 bits.

Figure 34-69. DXR2 Register

15	14	13	12	11	10	9	8
HWHB							
R/W-0h							
7	6	5	4	3	2	1	0
HWLB							
R/W-0h							

Table 34-81. DXR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	HWHB	R/W	0h	Low word high byte Reset type: SYSRSn
7-0	HWLB	R/W	0h	Low word low byte Reset type: SYSRSn

34.16.2.4 DXR1 Register (Offset = 3h) [Reset = 0h]

DXR1 is shown in Figure 34-70 and described in Table 34-82.

Return to the Summary Table.

DXR1 contains the lower 16 bits of the data to be transmitted after being written by the CPU or DMA.

Figure 34-70. DXR1 Register

15	14	13	12	11	10	9	8
LWHB							
R/W-0h							
7	6	5	4	3	2	1	0
LWLB							
R/W-0h							

Table 34-82. DXR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	LWHB	R/W	0h	Low word high byte Reset type: SYSRSn
7-0	LWLB	R/W	0h	Low word low byte Reset type: SYSRSn

34.16.2.5 SPCR2 Register (Offset = 4h) [Reset = 0h]

SPCR2 is shown in Figure 34-71 and described in Table 34-83.

Return to the Summary Table.

SPCR2 conatins control and status bits for various McBSP functions such as emulation modes, transmit interrupt mode control, transmitter status bits, and transmitter and other internal reset controls.

Figure 34-71. SPCR2 Register

15	14	13	12	11	10	9	8
RESERVED						FREE	SOFT
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
FRST	GRST	XINTM		XSYNCERR	XEMPTY	XRDY	XRST
R/W-0h	R/W-0h	R/W-0h		R/W-0h	R-0h	R-0h	R/W-0h

Table 34-83. SPCR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9	FREE	R/W	0h	Free run bit. When a breakpoint is encountered in the high-level language debugger, FREE determines whether the McBSP transmit and receive clocks continue to run or whether they are affected as determined by the SOFT bit. When one of the clocks stops, the corresponding data transfer (transmission or reception) stops. Reset type: SYSRSn
8	SOFT	R/W	0h	Soft stop bit. When FREE = 0, SOFT determines the response of the McBSP transmit and receive clocks when a breakpoint is encountered in the high-level language debugger. When one of the clocks stops, the corresponding data transfer (transmission or reception) stops. Reset type: SYSRSn
7	FRST	R/W	0h	Frame-synchronization logic reset bit. The sample rate generator of the McBSP includes framesynchronization logic to generate an internal frame-synchronization signal. You can use FRST to take the frame-synchronization logic into and out of its reset state. This bit has a negative polarity FRST = 0 indicates the reset state. Reset type: SYSRSn 0h (R/W) = If you read a 0, the frame-synchronization logic is in its reset state. If you write a 0, you reset the frame-synchronization logic. In the reset state, the frame-synchronization logic does not generate a frame-synchronization signal (FSG). 1h (R/W) = If you read a 1, the frame-synchronization logic is enabled. If you write a 1, you enable the frame-synchronization logic by taking it out of its reset state. When the frame-synchronization logic is enabled (FRST = 1) and the sample rate generator as a whole is enabled (GRST = 1), the frame-synchronization logic generates the frame-synchronization signal FSG as programmed.

Table 34-83. SPCR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	GRST	R/W	0h	Sample rate generator reset bit. You can use GRST to take the McBSP sample rate generator into and out of its reset state. This bit has a negative polarity GRST = 0 indicates the reset state. Reset type: SYSRSn 0h (R/W) = If you read a 0, the sample rate generator is in its reset state. If you write a 0, you reset the sample rate generator. If GRST = 0 due to a reset, CLKG is driven by the CPU clock divided by 2, and FSG is driven low (inactive). If GRST = 0 due to program code, CLKG and FSG are both driven low (inactive). 1h (R/W) = If you read a 1, the sample rate generator is enabled. If you write a 1, you enable the sample rate generator by taking it out of its reset state. When enabled, the sample rate generator generates the clock signal CLKG as programmed in the sample rate generator registers. If FRST = 1, the generator also generates the frame-synchronization signal FSG as programmed in the sample rate generator registers.
5-4	XINTM	R/W	0h	Transmit interrupt mode bits. XINTM determines which event in the McBSP transmitter generates a transmit interrupt (XINT) request. If XINT is properly enabled, the CPU services the interrupt request otherwise, the CPU ignores the request. Reset type: SYSRSn 0h (R/W) = The McBSP sends a transmit interrupt (XINT) request to the CPU when the XRDY bit changes from 0 to 1, indicating that transmitter is ready to accept new data (the content of DXR[1,2] has been copied to XSR[1,2]). Regardless of the value of XINTM, you can check XRDY to determine whether a word transfer is complete. The McBSP sends an XINT request to the CPU when 16 enabled bits have been transmitted on the DX pin. 1h (R/W) = In the multichannel selection mode, the McBSP sends an XINT request to the CPU after every 16- channel block is transmitted in a frame. Outside of the multichannel selection mode, no interrupt request is sent. 2h (R/W) = The McBSP sends an XINT request to the CPU when each transmit frame-synchronization pulse is detected. The interrupt request is sent even if the transmitter is in its reset state. 3h (R/W) = The McBSP sends an XINT request to the CPU when the XSYNCERR bit is set, indicating a transmit frame-synchronization error. Regardless of the value of XINTM, you can check XSYNCERR to determine whether a transmit framesynchronization error occurred.
3	XSYNCERR	R/W	0h	Transmit frame-synchronization error bit. XSYNCERR is set when a transmit frame-synchronization error is detected by the McBSP. If XINTM = 11b, the McBSP sends a transmit interrupt (XINT) request to the CPU when XSYNCERR is set. The flag remains set until you write a 0 to it or reset the transmitter. Reset type: SYSRSn 0h (R/W) = No error 1h (R/W) = Transmit frame-synchronization error

Table 34-83. SPCR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	XEMPTY	R	0h	Transmitter empty bit. XEMPTY is cleared when the transmitter is ready to send new data but no new data is available (transmitter-empty condition). This bit has a negative polarity a transmitter-empty condition is indicated by XEMPTY = 0. Reset type: SYSRSn 0h (R/W) = Transmitter-empty condition Typically this indicates that all the bits of the current word have been transmitted but there is no new data in DXR1. XEMPTY is also cleared if the transmitter is reset and then restarted. 1h (R/W) = No transmitter-empty condition
1	XRDY	R	0h	Transmitter ready bit. XRDY is set when the transmitter is ready to accept new data in DXR[1,2]. Specifically, XRDY is set in response to a copy from DXR1 to XSR1. If the transmit interrupt mode is XINTM = 00b, the McBSP sends a transmit interrupt (XINT) request to the CPU when XRDY changes from 0 to 1. Also, when XRDY changes from 0 to 1, the McBSP sends a transmit synchronization event (XEVT) signal to the DMA controller. Reset type: SYSRSn 0h (R/W) = Transmitter not ready When DXR1 is loaded, XRDY is automatically cleared. 1h (R/W) = Transmitter ready: DXR[1,2] is ready to accept new data. If both DXRs are needed (word length larger than 16 bits), the CPU or the DMA controller must load DXR2 first and then load DXR1. As soon as DXR1 is loaded, the contents of both DXRs are copied to the transmit shift registers (XSRs), as described in the next step. If DXR2 is not loaded first, the previous content of DXR2 is passed to the XSR2
0	XRST	R/W	0h	Transmitter reset bit. You can use XRST to take the McBSP transmitter into and out of its reset state. This bit has a negative polarity XRST = 0 indicates the reset state. To read about the effects of a transmitter reset, see Section 15.10.2, Resetting and Initializing a McBSP. Reset type: SYSRSn 0h (R/W) = If you read a 0, the transmitter is in its reset state. If you write a 0, you reset the transmitter. 1h (R/W) = If you read a 1, the transmitter is enabled. If you write a 1, you enable the transmitter by taking it out of its reset state.

34.16.2.6 SPCR1 Register (Offset = 5h) [Reset = 0h]

SPCR1 is shown in Figure 34-72 and described in Table 34-84.

Return to the Summary Table.

SPCR1 contains control and status bits for various McBSP functions such as digital loopback, receive data justification, clock stop mode, receive interrupt mode, DX pin delay enabler, receiver status bits, and receiver reset control.

Figure 34-72. SPCR1 Register

15	14	13	12	11	10	9	8
DLB	RJUST		CLKSTP		RESERVED		
R/W-0h	R/W-0h		R/W-0h		R-0h		
7	6	5	4	3	2	1	0
DXENA	RESERVED	RINTM		RSYNCERR	RFULL	RRDY	RRST
R/W-0h	R/W-0h	R/W-0h		R/W-0h	R-0h	R-0h	R/W-0h

Table 34-84. SPCR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	DLB	R/W	0h	Digital loopback mode bit. DLB disables or enables the digital loopback mode of the McBSP: Reset type: SYSRSn 0h (R/W) = Disabled Internal DR is supplied by the MDRx pin. Internal FSR and internal MCLKR can be supplied by their respective pins or by the sample rate generator, depending on the mode bits FSRM and CLKRM. Internal DX is supplied by the MDXx pin. Internal FSX and internal CLKX are supplied by their respective pins or are generated internally, depending on the mode bits FSXM and CLKXM. 1h (R/W) = Enabled Internal receive signals are supplied by internal transmit signals: MDRx connected to MDXx MFSRx connected to MFSXx MCLKR connected to MCLKXx This mode allows you to test serial port code with a single DSP. The McBSP transmitter directly supplies data, frame synchronization, and clocking to the McBSP receiver.
14-13	RJUST	R/W	0h	Receive sign-extension and justification mode bits. During reception, RJUST determines how data is justified and bit filled before being passed to the data receive registers (DRR1, DRR2). RJUST is ignored if you enable a companding mode with the RCOMPAND bits. In a companding mode, the 8-bit compressed data in RBR1 is expanded to left-justified 16-bit data in DRR1. Reset type: SYSRSn 0h (R/W) = Right justify the data and zero fill the MSBs 1h (R/W) = Right justify the data and sign-extend the data into the MSBs 2h (R/W) = Left justify the data and zero fill the LSBs 3h (R/W) = Reserved (do not use)
12-11	CLKSTP	R/W	0h	Clock stop mode bits. CLKSTP allows you to use the clock stop mode to support the SPI masterslave protocol. If you will not be using the SPI protocol, you can clear CLKSTP to disable the clock stop mode. In the clock stop mode, the clock stops at the end of each data transfer. At the beginning of eachdata transfer, the clock starts immediately (CLKSTP = 10b) or after a half-cycle delay (CLKSTP = 11b). Reset type: SYSRSn 0h (R/W) = Clock stop mode is disabled. 1h (R/W) = Clock stop mode is disabled. 2h (R/W) = Clock stop mode, without clock delay 3h (R/W) = Clock stop mode, with half-cycle clock delay

Table 34-84. SPCR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10-8	RESERVED	R	0h	Reserved
7	DXENA	R/W	0h	DX delay enabler mode bit. DXENA controls the delay enabler for the DX pin. The enabler creates an extra delay for turn-on time (for the length of the delay, see the device-specific data sheet). Reset type: SYSRSn 0h (R/W) = DX delay enabler off 1h (R/W) = DX delay enabler on
6	RESERVED	R/W	0h	Reserved
5-4	RINTM	R/W	0h	Receive interrupt mode bits. RINTM determines which event in the McBSP receiver generates areceive interrupt (RINT) request. If RINT is properly enabled inside the CPU, the CPU services the interrupt request otherwise, the CPU ignores the request. Reset type: SYSRSn 0h (R/W) = The McBSP sends a receive interrupt (RINT) request to the CPU when the RRDY bit changes from 0 to 1, indicating that receive data is ready to be read (the content of RBR[1,2] has been copied to DRR[1,2]): Regardless of the value of RINTM, you can check RRDY to determine whether a word transfer is complete. The McBSP sends a RINT request to the CPU when 16 enabled bits have been received on the DR pin. 1h (R/W) = In the multichannel selection mode, the McBSP sends a RINT request to the CPU after every 16- channel block is received in a frame. Outside of the multichannel selection mode, no interrupt request is sent. 2h (R/W) = The McBSP sends a RINT request to the CPU when each receive frame-synchronization pulse is detected. The interrupt request is sent even if the receiver is in its reset state. 3h (R/W) = The McBSP sends a RINT request to the CPU when the RSYNCERR bit is set, indicating a receive frame-synchronization error. Regardless of the value of RINTM, you can check RSYNCERR to determine whether a receive frame-synchronization error occurred.
3	RSYNCERR	R/W	0h	Receive frame-sync error bit. RSYNCERR is set when a receive frame-sync error is detected by the McBSP. If RINTM = 11b, the McBSP sends a receive interrupt (RINT) request to the CPU when RSYNCERR is set. The flag remains set until you write a 0 to it or reset the receiver. Reset type: SYSRSn 0h (R/W) = No error 1h (R/W) = Receive frame-synchronization error.
2	RFULL	R	0h	Receiver full bit. RFULL is set when the receiver is full with new data and the previously received data has not been read (receiver-full condition). For more details about this condition, Reset type: SYSRSn 0h (R/W) = No receiver-full condition 1h (R/W) = Receiver-full condition: RSR[1,2] and RBR[1,2] are full with new data, but the previous data in DRR[1,2] has not been read

Table 34-84. SPCR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	RRDY	R	0h	Receiver ready bit. RRDY is set when data is ready to be read from DRR[1,2]. Specifically, RRDY is set in response to a copy from RBR1 to DRR1. If the receive interrupt mode is RINTM = 00b, the McBSP sends a receive interrupt request to the CPU when RRDY changes from 0 to 1. Also, when RRDY changes from 0 to 1, the McBSP sends a receive synchronization event (REVT) signal to the DMA controller. Reset type: SYSRSn 0h (R/W) = Receiver not ready When the content of DRR1 is read, RRDY is automatically cleared. 1h (R/W) = Receiver ready: New data can be read from DRR[1,2]. Important: If both DRRs are required (word length larger than 16 bits), the CPU or the DMA controller must read from DRR2 first and then from DRR1. As soon as DRR1 is read, the next RBR-to-DRR copy is initiated. If DRR2 is not read first, the data in DRR2 is lost.
0	RRST	R/W	0h	Receiver reset bit. You can use RRST to take the McBSP receiver into and out of its reset state. This bit has a negative polarity RRST = 0 indicates the reset state. Reset type: SYSRSn 0h (R/W) = If you read a 0, the receiver is in its reset state. If you write a 0, you reset the receiver. 1h (R/W) = If you read a 1, the receiver is enabled. If you write a 1, you enable the receiver by taking it out of its reset state.

34.16.2.7 RCR2 Register (Offset = 6h) [Reset = 0h]

RCR2 is shown in Figure 34-73 and described in Table 34-85.

Return to the Summary Table.

RCR2 contains control bits for the receiver such as number of phases in each frame, the serial word length and number of words for phase 2 of dual phase frames, receive companding mode, receive frame synchronization ignore function, and the receive data delay.

Figure 34-73. RCR2 Register

15	14	13	12	11	10	9	8
RPHASE	RFRLEN2						
R/W-0h	R/W-0h						
7	6	5	4	3	2	1	0
RWDLEN2			RCOMPAND		RFIG	RDATDLY	
R/W-0h			R/W-0h		R/W-0h	R/W-0h	

Table 34-85. RCR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RPHASE	R/W	0h	Receive phase number bit. RPHASE determines whether the receive frame has one phase or two phases. For each phase you can define the serial word length and the number of serial words in the phase. To set up phase 1, program RWDLEN1 (word length) and RFRLEN1 (number of words). To set up phase 2 (if there are two phases), program RWDLEN2 and RFRLEN2. Reset type: SYSRSn 0h (R/W) = Single-phase frame The receive frame has only one phase, phase 1. 1h (R/W) = Dual-phase frame The receive frame has two phases, phase 1 and phase 2.
14-8	RFRLEN2	R/W	0h	Receive frame length 2 (1 to 128 words). Each frame of receive data can have one or two phases, depending on value that you load into the RPHASE bit. If a single-phase frame is selected, RFRLEN1 in RCR1 selects the number of serial words (8, 12, 16, 20, 24, or 32 bits per word) in the frame. If a dual-phase frame is selected, RFRLEN1 determines the number of serial words in phase 1 of the frame, and RFRLEN2 in RCR2 determines the number of words in phase 2 of the frame. The 7-bit RFRLEN fields allow up to 128 words per phase. See Table 15-77 for a summary of how to determine the frame length. This length corresponds to the number of words or logical time slots or channels per frame-synchronization period. Program the RFRLEN fields with [w minus 1], where w represents the number of words per phase. For example, if you want a phase length of 128 words in phase 2, load 127 into RFRLEN2. Reset type: SYSRSn

Table 34-85. RCR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-5	RWDLEN2	R/W	0h	Receive word length 2. Each frame of receive data can have one or two phases, depending on the value that you load into the RPHASE bit. If a single-phase frame is selected, RWDLEN1 in RCR1 selects the length for every serial word received in the frame. If a dual-phase frame is selected, RWDLEN1 determines the length of the serial words in phase 1 of the frame, and RWDLEN2 in RCR2 determines the word length in phase 2 of the frame. Reset type: SYSRSn 0h (R/W) = 8 bits 1h (R/W) = 12 bits 2h (R/W) = 16 bits 3h (R/W) = 20 bits 4h (R/W) = 24 bits 5h (R/W) = 32 bits 6h (R/W) = Reserved (do not use) 7h (R/W) = Reserved (do not use)
4-3	RCOMPAND	R/W	0h	Receive companding mode bits. Companding (COMpress and exPAND) hardware allows compression and expansion of data in either u-law or A-law format. RCOMPAND allows you to choose one of the following companding modes for the McBSP receiver: For more details about these companding modes, see Section 15.1.5, Companding (Compressing and Expanding) Data. Reset type: SYSRSn 0h (R/W) = No companding, any size data, MSB received first 1h (R/W) = No companding, 8-bit data, LSB received first 2h (R/W) = u-law companding, 8-bit data, MSB received first 3h (R/W) = A-law companding, 8-bit data, MSB received first
2	RFIG	R/W	0h	Receive frame-synchronization ignore bit. If a frame-synchronization pulse starts the transfer of a new frame before the current frame is fully received, this pulse is treated as an unexpected framesynchronization pulse. Unexpected Receive Frame-Synchronization Pulse. Setting RFIG causes the serial port to ignore unexpected frame-synchronization signals during reception. Reset type: SYSRSn 0h (R/W) = Frame-synchronization detect. An unexpected FSR pulse causes the receiver to discard the contents of RSR[1,2] in favor of the new incoming data. The receiver: 1. Aborts the current data transfer 2. Sets RSYNCERR in SPCR1 3. Begins the transfer of a new data word 1h (R/W) = Frame-synchronization ignore. An unexpected FSR pulse is ignored. Reception continues uninterrupted.
1-0	RDATDLY	R/W	0h	Receive data delay bits. RDATDLY specifies a data delay of 0, 1, or 2 receive clock cycles after framesynchronization and before the reception of the first bit of the frame. Reset type: SYSRSn 0h (R/W) = 0-bit data delay 1h (R/W) = 1-bit data delay 2h (R/W) = 2-bit data delay 3h (R/W) = Reserved (do not use)

34.16.2.8 RCR1 Register (Offset = 7h) [Reset = 0h]

RCR1 is shown in Figure 34-74 and described in Table 34-86.

Return to the Summary Table.

RCR1 contains control bits for the receiver such as the serial word length and number of words for single phase transmissions, or phase 1 if dual phase frames are used.

Figure 34-74. RCR1 Register

15	14	13	12	11	10	9	8
RESERVED	RFRLEN1						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
RWDLEN1			RESERVED				
R/W-0h			R-0h				

Table 34-86. RCR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14-8	RFRLEN1	R/W	0h	Receive frame length 1 (1 to 128 words). Each frame of receive data can have one or two phases,depending on value that you load into the RPHASE bit. If a single-phase frame is selected, RFRLEN1 inRCR1 selects the number of serial words (8, 12, 16, 20, 24, or 32 bits per word) in the frame. If a dualphase frame is selected, RFRLEN1 determines the number of serial words in phase 1 of the frame, and RFRLEN2 in RCR2 determines the number of words in phase 2 of the frame. The 7-bit RFRLEN fields allow up to 128 words per phase. See Table 15-75 for a summary of how you determine the frame length. This length corresponds to the number of words or logical time slots or channels per framesynchronization period. Program the RFRLEN fields with [w minus 1], where w represents the number of words per phase. For example, if you want a phase length of 128 words in phase 1, load 127 into RFRLEN1. Note: When operating in SPI mode, the frame length can only be 1 word. Reset type: SYSRSn
7-5	RWDLEN1	R/W	0h	Receive word length 1. Each frame of receive data can have one or two phases, depending on the value that you load into the RPHASE bit. If a single-phase frame is selected, RWDLEN1 in RCR1 selects the length for every serial word received in the frame. If a dual-phase frame is selected, RWDLEN1 determines the length of the serial words in phase 1 of the frame, and RWDLEN2 in RCR2 determines the word length in phase 2 of the frame. Reset type: SYSRSn 0h (R/W) = 8 bits 1h (R/W) = 12 bits 2h (R/W) = 16 bits 3h (R/W) = 20 bits 4h (R/W) = 24 bits 5h (R/W) = 32 bits 6h (R/W) = Reserved (do not use) 7h (R/W) = Reserved (do not use)
4-0	RESERVED	R	0h	Reserved

34.16.2.9 XCR2 Register (Offset = 8h) [Reset = 0h]

XCR2 is shown in Figure 34-75 and described in Table 34-87.

Return to the Summary Table.

XCR2 contains control bits for the transmitter such as number of phases in each frame, the serial word length and number of words for phase 2 of dual phase frames, transmit companding mode, transmit frame synchronization ignore function, and the transmit data delay control.

Figure 34-75. XCR2 Register

15	14	13	12	11	10	9	8
XPHASE	XFRLEN2						
R/W-0h	R/W-0h						
7	6	5	4	3	2	1	0
XWDLEN2			XCOMPAND		XFIG	XDATDLY	
R/W-0h			R/W-0h		R/W-0h	R/W-0h	

Table 34-87. XCR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	XPHASE	R/W	0h	Transmit phase number bit. XPHASE determines whether the transmit frame has one phase or two phases. For each phase you can define the serial word length and the number of serial words in the phase. To set up phase 1, program XWDLEN1 (word length) and XFRLEN1 (number of words). To set up phase 2 (if there are two phases), program XWDLEN2 and XFRLEN2. Reset type: SYSRSn
14-8	XFRLEN2	R/W	0h	Transmit frame length 2 (1 to 128 words). Each frame of transmit data can have one or two phases, depending on value that you load into the XPHASE bit. If a single-phase frame is selected, XFRLEN1 in XCR1 selects the number of serial words (8, 12, 16, 20, 24, or 32 bits per word) in the frame. If a dualphase frame is selected, XFRLEN1 determines the number of serial words in phase 1 of the frame and XFRLEN2 in XCR2 determines the number of words in phase 2 of the frame. The 7-bit XFRLEN fields allow up to 128 words per phase. See Table 15-81 for a summary of how to determine the frame length. This length corresponds to the number of words or logical time slots or channels per framesynchronization period. Program the XFRLEN fields with [w minus 1], where w represents the number of words per phase. For example, if you want a phase length of 128 words in phase 1, load 127 into XFRLEN1. Reset type: SYSRSn
7-5	XWDLEN2	R/W	0h	Transmit word length 2. Each frame of transmit data can have one or two phases, depending on the value that you load into the XPHASE bit. If a single-phase frame is selected, XWDLEN1 in XCR1 selects the length for every serial word transmitted in the frame. If a dual-phase frame is selected, XWDLEN1 determines the length of the serial words in phase 1 of the frame and XWDLEN2 in XCR2 determines the word length in phase 2 of the frame. Reset type: SYSRSn 0h (R/W) = 8 bits 1h (R/W) = 12 bits 2h (R/W) = 16 bits 3h (R/W) = 20 bits 4h (R/W) = 24 bits 5h (R/W) = 32 bits 6h (R/W) = Reserved (do not use) 7h (R/W) = Reserved (do not use)

Table 34-87. XCR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	XCOMPAND	R/W	0h	Transmit companding mode bits. Companding (COMpress and exPAND) hardware allows compression and expansion of data in either u-law or A-law format. Reset type: SYSRSn 0h (R/W) = No companding, any size data, MSB received first 1h (R/W) = No companding, 8-bit data, LSB received first 2h (R/W) = u-law companding, 8-bit data, MSB received first 3h (R/W) = A-law companding, 8-bit data, MSB received first
2	XFIG	R/W	0h	Transmit frame-synchronization ignore bit. If a frame-synchronization pulse starts the transfer of a new frame before the current frame is fully transmitted, this pulse is treated as an unexpected framesynchronization pulse. Setting XFIG causes the serial port to ignore unexpected frame-synchronization pulses during transmission. Reset type: SYSRSn 0h (R/W) = Frame-synchronization detect. An unexpected FSX pulse causes the transmitter to discard the content of XSR[1,2]. The transmitter: 1. Aborts the present transmission 2. Sets XSYNCERR in SPCR2 3. Begins a new transmission from DXR[1,2]. If new data was written to DXR[1,2] since the last DXR[1,2]-to-XSR[1,2] copy, the current value in XSR[1,2] is lost. Otherwise, the same data is transmitted. 1h (R/W) = Frame-synchronization ignore. An unexpected FSX pulse is ignored. Transmission continues uninterrupted.
1-0	XDATDLY	R/W	0h	Transmit data delay bits. XDATDLY specifies a data delay of 0, 1, or 2 transmit clock cycles after frame synchronization and before the transmission of the first bit of the frame. Reset type: SYSRSn 0h (R/W) = 0-bit data delay 1h (R/W) = 1-bit data delay 2h (R/W) = 2-bit data delay 3h (R/W) = Reserved (do not use)

34.16.2.10 XCR1 Register (Offset = 9h) [Reset = 0h]

XCR1 is shown in Figure 34-76 and described in Table 34-88.

Return to the Summary Table.

XCR1 contains control bits for the transmitter such as the serial word length and number of words for single phase transmissions, or phase 1 if dual phase frames are used.

Figure 34-76. XCR1 Register

15	14	13	12	11	10	9	8
RESERVED	XFRLEN1						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
XWDLEN1			RESERVED				
R/W-0h			R-0h				

Table 34-88. XCR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14-8	XFRLEN1	R/W	0h	Transmit frame length 1 (1 to 128 words). Each frame of transmit data can have one or two phases, depending on value that you load into the XPHASE bit. If a single-phase frame is selected, XFRLEN1 in XCR1 selects the number of serial words (8, 12, 16, 20, 24, or 32 bits per word) in the frame. If a dual-phase frame is selected, XFRLEN1 determines the number of serial words in phase 1 of the frame and XFRLEN2 in XCR2 determines the number of words in phase 2 of the frame. The 7-bit XFRLEN fields allow up to 128 words per phase. See Table 15-79 for a summary of how you determine the frame length. This length corresponds to the number of words or logical time slots or channels per frame-synchronization period. Program the XFRLEN fields with [w minus 1], where w represents the number of words per phase. For example, if you want a phase length of 128 words in phase 1, load 127 into XFRLEN1. Note: When operating in SPI mode, the frame length can only be 1 word. Reset type: SYSRSn
7-5	XWDLEN1	R/W	0h	Transmit word length 1. Each frame of transmit data can have one or two phases, depending on the value that you load into the XPHASE bit. If a single-phase frame is selected, XWDLEN1 in XCR1 selects the length for every serial word transmitted in the frame. If a dual-phase frame is selected, XWDLEN1 determines the length of the serial words in phase 1 of the frame and XWDLEN2 in XCR2 determines the word length in phase 2 of the frame. Reset type: SYSRSn 0h (R/W) = 8 bits 1h (R/W) = 12 bits 2h (R/W) = 16 bits 3h (R/W) = 20 bits 4h (R/W) = 24 bits 5h (R/W) = 32 bits 6h (R/W) = Reserved (do not use) 7h (R/W) = Reserved (do not use)
4-0	RESERVED	R	0h	Reserved

34.16.2.11 SRGR2 Register (Offset = Ah) [Reset = 0h]

SRGR2 is shown in Figure 34-77 and described in Table 34-89.

Return to the Summary Table.

SRGR2 contains control bits for the sample rate generator such as input clock selection, internal tranmist frame-synchronization source selection, and the period between frame-synchronization pulses.
If an external source provides the input clock source for the sample rate generator, a control bit is provided to make the CLKG synchronized to an external frame synchronization pulse on the FSR pin so that CLKG is kept in phase with the input clock.

Figure 34-77. SRGR2 Register

15	14	13	12	11	10	9	8
GSYNC	RESERVED	CLKSM	FSGM	FPER			
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h			
7	6	5	4	3	2	1	0
FPER							
R/W-0h							

Table 34-89. SRGR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	GSYNC	R/W	0h	Clock synchronization mode bit for CLKG. GSYNC is used only when the input clock source for the sample rate generator is external?on the MCLKR pin. When GSYNC = 1, the clock signal (CLKG) and the frame-synchronization signal (FSG) generated by the sample rate generator are made dependent on pulses on the FSR pin. Reset type: SYSRSn 0h (R/W) = No clock synchronization CLKG oscillates without adjustment, and FSG pulses every (FPER + 1) CLKG cycles. 1h (R/W) = Clock synchronization - CLKG is adjusted as necessary so that it is synchronized with the input clock on the MCLKR pin. - FSG pulses. FSG only pulses in response to a pulse on the FSR pin. The frame-synchronization period defined in FPER is ignored.
14	RESERVED	R/W	0h	Reserved

Table 34-89. SRGR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	CLKSM	R/W	0h	Sample rate generator mode Reset type: SYSRSn 0h (R/W) = Sample rate generator input clock mode bit. The sample rate generator can accept an input clock signal and divide it down according to CLKGDV to produce an output clock signal, CLKG. The frequency of CLKG is:) CLKG frequency = (input clock frequency)/ (CLKGDV + 1 CLKSM is used in conjunction with the SCLKME bit to determine the source for the input clock. A reset selects the CPU clock as the input clock and forces the CLKG frequency to half the LSPCLK frequency. The input clock for the sample rate generator is taken from the MCLKR pin, depending on the value of the SCLKME bit of PCR: SCLKME CLKSM Input Clock For Sample Rate Generator 0 0 Reserved 1 0 Signal on MCLKR pin 1h (R/W) = The input clock for the sample rate generator is taken from the LSPCLK or from the MCLKX pin, depending on the value of the SCLKME bit of PCR: SCLKME CLKSM Input Clock For Sample Rate Generator 0 1 LSPCLK 1 1 Signal on MCLKX pin
12	FSGM	R/W	0h	Sample rate generator transmit frame-synchronization mode bit. The transmitter can get frame synchronization from the FSX pin (FSXM = 0) or from inside the McBSP (FSXM = 1). When FSXM = 1, the FSGM bit determines how the McBSP supplies frame-synchronization pulses. Reset type: SYSRSn 0h (R/W) = If FSXM = 1, the McBSP generates a transmit frame-synchronization pulse when the content of DXR[1,2] is copied to XSR[1,2]. 1h (R/W) = If FSXM = 1, the transmitter uses frame-synchronization pulses generated by the sample rate generator. Program the FWID bits to set the width of each pulse. Program the FPER bits to set the period between pulses.
11-0	FPER	R/W	0h	Frame-synchronization period bits for FSG. The sample rate generator can produce a clock signal, CLKG, and a frame-synchronization signal, FSG. The period between framesynchronization pulses on FSG is (FPER + 1) CLKG cycles. The 12 bits of FPER allow a frame-synchronization period of 1 to 4096 CLKG cycles: Reset type: SYSRSn

34.16.2.12 SRGR1 Register (Offset = Bh) [Reset = 1h]

SRGR1 is shown in Figure 34-78 and described in Table 34-90.

Return to the Summary Table.

SRGR1 contains control bits for the sample rate generator functions such as the divide down frequency, and the width fo the frame-synchronization pulses on FSG.

Figure 34-78. SRGR1 Register

15	14	13	12	11	10	9	8
FWID							
R/W-0h							
7	6	5	4	3	2	1	0
CLKGDV							
R/W-1h							

Table 34-90. SRGR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	FWID	R/W	0h	Divide-down value for CLKG. The sample rate generator can accept an input clock signal and divide it down according to CLKGDV to produce an output clock signal, CLKG. The frequency of CLKG is: CLKG frequency = (Input clock frequency)/ (CLKGDV + 1) The input clock is selected by the SCLKME and CLKSM bits: SCLKME CLKSM Input Clock For Sample Rate Generator 0 0 Reserved 0 1 LSPCLK 1 0 Signal on MCLKR pin 1 1 Signal on MCLKX pin Reset type: SYSRSn
7-0	CLKGDV	R/W	1h	Frame-synchronization pulse width bits for FSG The sample rate generator can produce a clock signal, CLKG, and a frame-synchronization signal, FSG. For frame-synchronization pulses on FSG, (FWID + 1) is the pulse width in CLKG cycles. The eight bits of FWID allow a pulse width of 1 to 256 CLKG cycles: 0 <= FWID <= 255 1 <= (FWID + 1) <= 256 CLKG cycles The period between the frame-synchronization pulses on FSG is defined by the FPER bits. Reset type: SYSRSn

34.16.2.13 MCR2 Register (Offset = Ch) [Reset = 0h]

MCR2 is shown in Figure 34-79 and described in Table 34-91.

Return to the Summary Table.

MCR2 contains control bits for the transmitter multi-channel functions such as channel enable mode selection, channel partition modes, channel block assignments, and active channel status bits.

Figure 34-79. MCR2 Register

15	14	13	12	11	10	9	8
RESERVED						XMCME	XPBBLK
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
XPBBLK	XPABLK		XCBLK			XMCM	
R/W-0h	R/W-0h		R-0h			R/W-0h	

Table 34-91. MCR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9	XMCME	R/W	0h	Transmit multichannel partition mode bit. XMCME determines whether only 32 channels or all 128 channels are to be individually selectable. XMCME is only applicable if channels can be individually disabled/enabled or masked/unmasked for transmission (XMCM is nonzero). Reset type: SYSRSn
8-7	XPBBLK	R/W	0h	Transmit partition B block bits XPBBLK is only applicable if channels can be individually disabled/ enabled and masked/unmasked (XMCM is nonzero) and the 2-partition mode is selected (XMCME = 0). Under these conditions, the McBSP transmitter can transmit or withhold data in any of the 32 channels that are assigned to partitions A and B of the transmitter. The 128 transmit channels of the McBSP are divided equally among 8 blocks (0 through 7). When XPBBLK is applicable, use XPBBLK to assign one of the odd-numbered blocks (1, 3, 5, or 7) to partition B, as shown in the following table. Use the PABLK bit to assign one of the evennumbered blocks (0, 2, 4, or 6) to partition A. If you want to use more than 32 channels, you can change block assignments dynamically. You can assign a new block to one partition while the transmitter is handling activity in the other partition. For example, while the block in partition A is active, you can change which block is assigned to partition B. The XCBLK bits are regularly pdated to indicate which block is active. When XMCM = 11b (for symmetric transmission and reception), the transmitter uses the receive block bits (RPABLK and RPBBLK) rather than the transmit block bits (XPABLK and XPBBLK). Reset type: SYSRSn 0h (R/W) = Block 1: channels 16 through 31 1h (R/W) = Block 3: channels 48 through 63 2h (R/W) = Block 5: channels 80 through 95 3h (R/W) = Block 7: channels 112 through 127

Table 34-91. MCR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-5	XPABLK	R/W	0h	Transmit partition A block bits. XPABLK is only applicable if channels can be individually disabled/enabled and masked/unmasked (XMCM is nonzero) and the 2-partition mode is selected (XMCME = 0). Under these conditions, the McBSP transmitter can transmit or withhold data in any of the 32 channels that are assigned to artitions A and B of the transmitter. See the Descriptionfor XPBBLK (bits 8-7) for more information about assigning blocks to partitions A and B. Reset type: SYSRSn 0h (R/W) = Block 0: channels 0 through 15 1h (R/W) = Block 2: channels 32 through 47 2h (R/W) = Block 4: channels 64 through 79 3h (R/W) = Block 6: channels 96 through 111
4-2	XCBLK	R	0h	Transmit current block indicator. XCBLK indicates which block of 16 channels is involved in the current McBSP transmission: Reset type: SYSRSn 0h (R/W) = Block 0: channels 0 through 15 1h (R/W) = Block 1: channels 16 through 31 2h (R/W) = Block 2: channels 32 through 47 3h (R/W) = Block 3: channels 48 through 63 4h (R/W) = Block 4: channels 64 through 79 5h (R/W) = Block 5: channels 80 through 95 6h (R/W) = Block 6: channels 96 through 111 7h (R/W) = Block 7: channels 112 through 127
1-0	XMCM	R/W	0h	Transmit multichannel selection mode bits. XMCM determines whether all channels or only selected channels are enabled and unmasked for transmission. Reset type: SYSRSn 0h (R/W) = No transmit multichannel selection mode is on. All channels are enabled and unmasked. No channels can be disabled or masked 1h (R/W) = All channels are disabled unless they are selected in the appropriate transmit channel enable registers (XCERs). If enabled, a channel in this mode is also unmasked. The XMCME bit determines whether 32 channels or 128 channels are selectable in XCERs. 2h (R/W) = All channels are enabled, but they are masked unless they are selected in the appropriate transmit channel enable registers (XCERs). The XMCME bit determines whether 32 channels or 128 channels are selectable in XCERs. 3h (R/W) = This mode is used for symmetric transmission and reception. All channels are disabled for transmission unless they are enabled for reception in the appropriate receive channel enable registers (RCERs). Once enabled, they are masked unless they are also selected in the appropriate transmit channel enable registers (XCERs). The XMCME bit determines whether 32 channels or 128 channels are selectable in RCERs and XCERs.

34.16.2.14 MCR1 Register (Offset = Dh) [Reset = 0h]

MCR1 is shown in Figure 34-80 and described in Table 34-92.

Return to the Summary Table.

MCR1 contains control bits for the receiver multi-channel functions such as channel enable mode selection, channel partition modes, channel block assignments, and active channel status bits.

Figure 34-80. MCR1 Register

15	14	13	12	11	10	9	8
RESERVED						RMCME	RPBBLK
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RPBBLK	RPABLK		RCBLK			RESERVED	RMCM
R/W-0h	R/W-0h		R-0h			R/W-0h	R/W-0h

Table 34-92. MCR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9	RMCME	R/W	0h	Receive multichannel partition mode bit. RMCME is only applicable if channels can be individually enabled or disabled for reception (RMCM = 1). RMCME determines whether only 32 channels or all 128 channels are to be individually selectable. Reset type: SYSRSn 0h (R/W) = 2-partition mode Only partitions A and B are used. You can control up to 32 channels in the receive multichannel selection mode (RMCM = 1). Assign 16 channels to partition A with the RPABLK bits. Assign 16 channels to partition B with the RPBBLK bits. You control the channels with the appropriate receive channel enable registers: RCERA: Channels in partition A RCERB: Channels in partition B 1h (R/W) = 8-partition mode All partitions (A through H) are used. You can control up to 128 channels in the receive multichannel selection mode. You control the channels with the appropriate receive channel enable registers: RCERA: Channels 0 through 15 RCERB: Channels 16 through 31 RCERC: Channels 32 through 47 RCERD: Channels 48 through 63 RCERE: Channels 64 through 79 RCERF: Channels 80 through 95 RCERG: Channels 96 through 111 RCERH: Channels 112 through 127

Table 34-92. MCR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8-7	RPBBLK	R/W	0h	Receive partition B block bits RPBBLK is only applicable if channels can be individually enabled or disabled (RMCM = 1) and the 2-partition mode is selected (RMCME = 0). Under these conditions, the McBSP receiver can accept or ignore data in any of the 32 channels that are assigned to partitions A and B of the receiver. The 128 receive channels of the McBSP are divided equally among 8 blocks (0 through 7). When RPBBLK is applicable, use RPBBLK to assign one of the odd-numbered blocks (1, 3, 5, or 7) to partition B. Use the RPABLK bits to assign one of the even-numbered blocks (0, 2, 4, or 6) to partition A. If you want to use more than 32 channels, you can change block assignments dynamically. You can assign a new block to one partition while the receiver is handling activity in the other partition. For example, while the block in partition A is active, you can change which block is assigned to partition B. The RCBLK bits are regularly updated to indicate which block is active. When XMCM = 11b (for symmetric transmission and reception), the transmitter uses the receive block bits (RPABLK and RPBBLK) rather than the transmit block bits (XPABLK and XPBBLK). Reset type: SYSRSn 0h (R/W) = Block 1: channels 16 through 31 1h (R/W) = Block 3: channels 48 through 63 2h (R/W) = Block 5: channels 80 through 95 3h (R/W) = Block 7: channels 112 through 127
6-5	RPABLK	R/W	0h	Receive partition A block bits RPABLK is only applicable if channels can be individually enabled or disabled (RMCM = 1) and the 2-partition mode is selected (RMCME = 0). Under these conditions, the McBSP receiver can accept or ignore data in any of the 32 channels that are assigned to partitions A and B of the receiver. See the Descriptionfor RPBBLK (bits 8-7) for more information about assigning blocks to partitions A and B. Reset type: SYSRSn 0h (R/W) = Block 0: channels 0 through 15 1h (R/W) = Block 2: channels 32 through 47 2h (R/W) = Block 4: channels 64 through 79 3h (R/W) = Block 6: channels 96 through 111
4-2	RCBLK	R	0h	Receive current block indicator. RCBLK indicates which block fo 16 channels is involved in the current McBSP reception Reset type: SYSRSn 0h (R/W) = Block 0: channels 0 through 15 1h (R/W) = Block 1: channels 16 through 31 2h (R/W) = Block 2: channels 32 through 47 3h (R/W) = Block 3: channels 48 through 63 4h (R/W) = Block 4: channels 64 through 79 5h (R/W) = Block 5: channels 80 through 95 6h (R/W) = Block 6: channels 96 through 111 7h (R/W) = Block 7: channels 112 through 127
1	RESERVED	R/W	0h	Reserved
0	RMCM	R/W	0h	Receive multichannel selection mode bit. RMCM determines whether all channels or only selected channels are enabled for reception: Reset type: SYSRSn 0h (R/W) = All 128 channels are enabled. 1h (R/W) = Multichanneled selection mode. Channels can be individually enabled or disabled. The only channels enabled are those selected in the appropriate receive channel enable registers (RCERs). The way channels are assigned to the RCERs depends on the number of receive channel partitions (2 or 8), as defined by the RMCME bit.

34.16.2.15 RCERA Register (Offset = Eh) [Reset = 0h]

RCERA is shown in Figure 34-81 and described in Table 34-93.

Return to the Summary Table.

RCERA contains the receive channel enable registers for the A partition. This register is only used when the receiver is configured to allow individual disabling/enabling and masking/unmasking of the channels (e.g. RMCM is nonzero).

Figure 34-81. RCERA Register

15	14	13	12	11	10	9	8
RCEA							
R/W-0h							
7	6	5	4	3	2	1	0
RCEA							
R/W-0h							

Table 34-93. RCERA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RCEA	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1): Reset type: SYSRSn 0h (R/W) = Disable the channel that is mapped to RCEx. 1h (R/W) = Enable the channel that is mapped to RCEx.

34.16.2.16 RCERB Register (Offset = Fh) [Reset = 0h]

RCERB is shown in Figure 34-82 and described in Table 34-94.

Return to the Summary Table.

RCERB contains the receive channel enable registers for the B partition. This register is only used when the receiver is configured to allow individual disabling/enabling and masking/unmasking of the channels (e.g. RMCM is nonzero).

Figure 34-82. RCERB Register

15	14	13	12	11	10	9	8
RCEB							
R/W-0h							
7	6	5	4	3	2	1	0
RCEB							
R/W-0h							

Table 34-94. RCERB Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RCEB	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1): Reset type: SYSRSn 0h (R/W) = Disable the channel that is mapped to RCEx. 1h (R/W) = Enable the channel that is mapped to RCEx.

34.16.2.17 XCERA Register (Offset = 10h) [Reset = 0h]

XCERA is shown in Figure 34-83 and described in Table 34-95.

Return to the Summary Table.

XCERA contains the transmit channel enable registers for the A partition. This register is only used when the transmitter is configured to allow individual disabling/enabling and masking/unmasking of the channels (e.g. XMCM is nonzero).

Figure 34-83. XCERA Register

15	14	13	12	11	10	9	8
XCERA							
R/W-0h							
7	6	5	4	3	2	1	0
XCERA							
R/W-0h							

Table 34-95. XCERA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	XCERA	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel selection mode is selected with the XMCM bits. Reset type: SYSRSn 0h (R/W) = For multichannel selection when XMCM = 01b (all channels disabled unless selected): Disable and mask the channel that is mapped to XCEx. For multichannel selection when XMCM = 10b (all channels enabled but masked unless selected): Mask the channel that is mapped to XCEx. For multichannel selection when XMCM = 11b (all channels masked unless selected): Mask the channel that is mapped to XCEx. Even if the channel is enabled by the corresponding receive channel enable bit, this channel's data cannot appear on the DX pin. 1h (R/W) = For multichannel selection when XMCM = 01b (all channels disabled unless selected): Enable and unmask the channel that is mapped to XCEx. For multichannel selection when XMCM = 10b (all channels enabled but masked unless selected): Unmask the channel that is mapped to XCEx. For multichannel selection when XMCM = 11b (all channels masked unless selected): Unmask the channel that is mapped to XCEx. If the channel is also enabled by the corresponding receive channel enable bit, full transmission can occur.

34.16.2.18 XCERB Register (Offset = 11h) [Reset = 0h]

XCERB is shown in Figure 34-84 and described in Table 34-96.

Return to the Summary Table.

XCERB contains the transmit channel enable registers for the B partition. This register is only used when the transmitter is configured to allow individual disabling/enabling and masking/unmasking of the channels (e.g. XMCM is nonzero).

Figure 34-84. XCERB Register

15	14	13	12	11	10	9	8
XCERB							
R/W-0h							
7	6	5	4	3	2	1	0
XCERB							
R/W-0h							

Table 34-96. XCERB Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	XCERB	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel selection mode is selected with the XMCM bits. Reset type: SYSRSn 0h (R/W) = For multichannel selection when XMCM = 01b (all channels disabled unless selected): Disable and mask the channel that is mapped to XCEx. For multichannel selection when XMCM = 10b (all channels enabled but masked unless selected): Mask the channel that is mapped to XCEx. For multichannel selection when XMCM = 11b (all channels masked unless selected): Mask the channel that is mapped to XCEx. Even if the channel is enabled by the corresponding receive channel enable bit, this channel's data cannot appear on the DX pin. 1h (R/W) = For multichannel selection when XMCM = 01b (all channels disabled unless selected): Enable and unmask the channel that is mapped to XCEx. For multichannel selection when XMCM = 10b (all channels enabled but masked unless selected): Unmask the channel that is mapped to XCEx. For multichannel selection when XMCM = 11b (all channels masked unless selected): Unmask the channel that is mapped to XCEx. If the channel is also enabled by the corresponding receive channel enable bit, full transmission can occur.

34.16.2.19 PCR Register (Offset = 12h) [Reset = 0h]

PCR is shown in Figure 34-85 and described in Table 34-97.

Return to the Summary Table.

PCR contains control bits for the McBSP pins such as frame synchronization modes, clock modes, input clock source selection for the sample rate generator, frame synchronization pulse active polarity, and transmit/receive active edge selection.

Figure 34-85. PCR Register

15	14	13	12	11	10	9	8
RESERVED				FSXM	FSRM	CLKXM	CLKRM
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
SCLKME	RESERVED	RESERVED	RESERVED	FSXP	FSRP	CLKXP	CLKRP
R/W-0h	R-0h	R-0h	R-0h	R/W-0h	R-0h	R-0h	R/W-0h

Table 34-97. PCR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	FSXM	R/W	0h	Transmit frame-synchronization mode bit. FSXM determines whether transmit framesynchronization pulses are supplied externally or internally. The polarity of the signal on the FSX pin is determined by the FSXP bit. Reset type: SYSRSn 0h (R/W) = Transmit frame synchronization is supplied by an external source via the FSX pin. 1h (R/W) = Transmit frame synchronization is generated internally by the Sample Rate generator, as determined by the FSGM bit of SRGR2
10	FSRM	R/W	0h	Receive frame-synchronization mode bit. FSRM determines whether receive framesynchronization pulses are supplied externally or internally. The polarity of the signal on the FSR pin is determined by the FSRP bit. Reset type: SYSRSn 0h (R/W) = Receive frame synchronization is supplied by an external source via the FSR pin. 1h (R/W) = Receive frame synchronization is supplied by the sample rate generator. FSR is an output pin reflecting internal FSR, except when GSYNC = 1 in SRGR2

Table 34-97. PCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	CLKXM	R/W	0h	Transmit clock mode bit. CLKXM determines whether the source for the transmit clock is external or internal, and whether the MCLKX pin is an input or an output. The polarity of the signal on the MCLKX pin is determined by the CLKXP bit. In the clock stop mode (CLKSTP = 10b or 11b), the McBSP can act as a master or as a slave in the SPI protocol. If the McBSP is a master, make sure that CLKX is an output. If the McBSP is a slave, make sure that CLKX is an input. Reset type: SYSRSn 0h (R/W) = Not in clock stop mode (CLKSTP = 00b or 01b): The transmitter gets its clock signal from an external source via the MCLKX pin. In clock stop mode (CLKSTP = 10b or 11b): The McBSP is a slave in the SPI protocol. The internal transmit clock (CLKX) is driven by the SPI master via the MCLKX pin. The internal receive clock (MCLKR) is driven internally by CLKX, so that both the transmitter and the receiver are controlled by the external master clock. 1h (R/W) = Not in clock stop mode (CLKSTP = 00b or 01b): Internal CLKX is driven by the sample rate generator of the McBSP. The MCLKX pin is an output pin that reflects internal CLKX. In clock stop mode (CLKSTP = 10b or 11b): The McBSP is a master in the SPI protocol. The sample rate generator drives the internal transmit clock (CLKX). Internal CLKX is reflected on the MCLKX pin to drive the shift clock of the SPI-compliant slaves in the system. Internal CLKX also drives the internal receive clock (MCLKR), so that both the transmitter and the receiver are controlled by the internal master clock
8	CLKRM	R/W	0h	Receive clock mode bit. The role of CLKRM and the resulting effect on the MCLKR pin depend on whether the McBSP is in the digital loopback mode (DLB = 1). The polarity of the signal on the MCLKR pin is determined by the CLKRP bit. Reset type: SYSRSn 0h (R/W) = Not in digital loopback mode (DLB = 0): The MCLKR pin is an input pin that supplies the internal receive clock (MCLKR). In digital loopback mode (DLB = 1): The MCLKR pin is in the high impedance state. The internal receive clock (MCLKR) is driven by the internal transmit clock (CLKX). CLKX is derived according to the CLKXM bit. 1h (R/W) = Not in digital loopback mode (DLB = 0): Internal MCLKR is driven by the sample rate generator of the McBSP. The MCLKR pin is an output pin that reflects internal MCLKR. In digital loopback mode (DLB = 1): Internal MCLKR is driven by internal CLKX. The MCLKR pin is an output pin that reflects internal MCLKR. CLKX is derived according to the CLKXM bit.

Table 34-97. PCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	SCLKME	R/W	0h	Sample rate generator input clock mode bit. The sample rate generator can produce a clock signal, CLKG. The frequency of CLKG is: CLKG freq. = (Input clock frequency) / (CLKGDV + 1) SCLKME is used in conjunction with the CLKSM bit to select the input clock. SCLKME CLKSM Input Clock For Sample Rate Generator 0 0 Reserved 0 1 LSPCLK The input clock for the sample rate generator is taken from the MCLKR pin or from the MCLKX pin, depending on the value of the CLKSM bit of SRGR2: SCLKME CLKSM Input Clock For Sample Rate Generator 1 0 Signal on MCLKR pin 1 1 Signal on MCLKX pin Reset type: SYSRSn
6	RESERVED	R	0h	Reserved
5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	FSXP	R/W	0h	Transmit frame-synchronization polarity bit. FSXP determines the polarity of FSX as seen on the FSX pin. Reset type: SYSRSn 0h (R/W) = Transmit frame-synchronization pulses are active high. 1h (R/W) = Transmit frame-synchronization pulses are active low.
2	FSRP	R	0h	Receive frame-synchronization polarity bit. FSRP determines the polarity of FSR as seen on the FSR pin. Reset type: SYSRSn 0h (R/W) = Receive frame-synchronization pulses are active high. 1h (R/W) = Receive frame-synchronization pulses are active low.
1	CLKXP	R	0h	Transmit clock polarity bit. CLKXP determines the polarity of CLKX as seen on the MCLKX pin. Reset type: SYSRSn 0h (R/W) = Transmit data is sampled on the rising edge of CLKX. 1h (R/W) = Transmit data is sampled on the falling edge of CLKX.
0	CLKRP	R/W	0h	Receive clock polarity bit. CLKRP determines the polarity of CLKR as seen on the MCLKR pin. Reset type: SYSRSn 0h (R/W) = Receive data is sampled on the falling edge of MCLKR. 1h (R/W) = Receive data is sampled on the rising edge of MCLKR.

34.16.2.20 RCERC Register (Offset = 13h) [Reset = 0h]

RCERC is shown in Figure 34-86 and described in Table 34-98.

Return to the Summary Table.

RCERC contains the receive channel enable registers for the C partition. This register is only used when the receiver is configured to allow individual disabling/enabling and masking/unmasking of the channels (e.g. RMCM is nonzero).

Figure 34-86. RCERC Register

15	14	13	12	11	10	9	8
RCEC							
R/W-0h							
7	6	5	4	3	2	1	0
RCEC							
R/W-0h							

Table 34-98. RCERC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RCEC	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1): Reset type: SYSRSn 0h (R/W) = Disable the channel that is mapped to RCEx. 1h (R/W) = Enable the channel that is mapped to RCEx.

34.16.2.21 RCERD Register (Offset = 14h) [Reset = 0h]

RCERD is shown in Figure 34-87 and described in Table 34-99.

Return to the Summary Table.

RCERD contains the receive channel enable registers for the D partition. This register is only used when the receiver is configured to allow individual disabling/enabling and masking/unmasking of the channels (e.g. RMCM is nonzero).

Figure 34-87. RCERD Register

15	14	13	12	11	10	9	8
RCED							
R/W-0h							
7	6	5	4	3	2	1	0
RCED							
R/W-0h							

Table 34-99. RCERD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RCED	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1): Reset type: SYSRSn 0h (R/W) = Disable the channel that is mapped to RCEx. 1h (R/W) = Enable the channel that is mapped to RCEx.

34.16.2.22 XCERC Register (Offset = 15h) [Reset = 0h]

XCERC is shown in Figure 34-88 and described in Table 34-100.

Return to the Summary Table.

XCERC contains the transmit channel enable registers for the C partition. This register is only used when the transmitter is configured to allow individual disabling/enabling and masking/unmasking of the channels (e.g. XMCM is nonzero).

Figure 34-88. XCERC Register

15	14	13	12	11	10	9	8
XCERC							
R/W-0h							
7	6	5	4	3	2	1	0
XCERC							
R/W-0h							

Table 34-100. XCERC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	XCERC	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel selection mode is selected with the XMCM bits. Reset type: SYSRSn 0h (R/W) = For multichannel selection when XMCM = 01b (all channels disabled unless selected): Disable and mask the channel that is mapped to XCEx. For multichannel selection when XMCM = 10b (all channels enabled but masked unless selected): Mask the channel that is mapped to XCEx. For multichannel selection when XMCM = 11b (all channels masked unless selected): Mask the channel that is mapped to XCEx. Even if the channel is enabled by the corresponding receive channel enable bit, this channel's data cannot appear on the DX pin. 1h (R/W) = For multichannel selection when XMCM = 01b (all channels disabled unless selected): Enable and unmask the channel that is mapped to XCEx. For multichannel selection when XMCM = 10b (all channels enabled but masked unless selected): Unmask the channel that is mapped to XCEx. For multichannel selection when XMCM = 11b (all channels masked unless selected): Unmask the channel that is mapped to XCEx. If the channel is also enabled by the corresponding receive channel enable bit, full transmission can occur.

34.16.2.23 XCERD Register (Offset = 16h) [Reset = 0h]

XCERD is shown in Figure 34-89 and described in Table 34-101.

Return to the Summary Table.

XCERD contains the transmit channel enable registers for the D partition. This register is only used when the transmitter is configured to allow individual disabling/enabling and masking/unmasking of the channels (e.g. XMCM is nonzero).

Figure 34-89. XCERD Register

15	14	13	12	11	10	9	8
XCERD							
R/W-0h							
7	6	5	4	3	2	1	0
XCERD							
R/W-0h							

Table 34-101. XCERD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	XCERD	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel selection mode is selected with the XMCM bits. Reset type: SYSRSn 0h (R/W) = For multichannel selection when XMCM = 01b (all channels disabled unless selected): Disable and mask the channel that is mapped to XCEx. For multichannel selection when XMCM = 10b (all channels enabled but masked unless selected): Mask the channel that is mapped to XCEx. For multichannel selection when XMCM = 11b (all channels masked unless selected): Mask the channel that is mapped to XCEx. Even if the channel is enabled by the corresponding receive channel enable bit, this channel's data cannot appear on the DX pin. 1h (R/W) = For multichannel selection when XMCM = 01b (all channels disabled unless selected): Enable and unmask the channel that is mapped to XCEx. For multichannel selection when XMCM = 10b (all channels enabled but masked unless selected): Unmask the channel that is mapped to XCEx. For multichannel selection when XMCM = 11b (all channels masked unless selected): Unmask the channel that is mapped to XCEx. If the channel is also enabled by the corresponding receive channel enable bit, full transmission can occur.

34.16.2.24 RCERE Register (Offset = 17h) [Reset = 0h]

RCERE is shown in Figure 34-90 and described in Table 34-102.

Return to the Summary Table.

RCERE contains the receive channel enable registers for the E partition. This register is only used when the receiver is configured to allow individual disabling/enabling and masking/unmasking of the channels (e.g. RMCM is nonzero).

Figure 34-90. RCERE Register

15	14	13	12	11	10	9	8
RCEE							
R/W-0h							
7	6	5	4	3	2	1	0
RCEE							
R/W-0h							

Table 34-102. RCERE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RCEE	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1): Reset type: SYSRSn 0h (R/W) = Disable the channel that is mapped to RCEx. 1h (R/W) = Enable the channel that is mapped to RCEx.

34.16.2.25 RCERF Register (Offset = 18h) [Reset = 0h]

RCERF is shown in Figure 34-91 and described in Table 34-103.

Return to the Summary Table.

RCERF contains the receive channel enable registers for the F partition. This register is only used when the receiver is configured to allow individual disabling/enabling and masking/unmasking of the channels (e.g. RMCM is nonzero).

Figure 34-91. RCERF Register

15	14	13	12	11	10	9	8
RCEF							
R/W-0h							
7	6	5	4	3	2	1	0
RCEF							
R/W-0h							

Table 34-103. RCERF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RCEF	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1): Reset type: SYSRSn 0h (R/W) = Disable the channel that is mapped to RCEx. 1h (R/W) = Enable the channel that is mapped to RCEx.

34.16.2.26 XCERE Register (Offset = 19h) [Reset = 0h]

XCERE is shown in Figure 34-92 and described in Table 34-104.

Return to the Summary Table.

XCERE contains the transmit channel enable registers for the E partition. This register is only used when the transmitter is configured to allow individual disabling/enabling and masking/unmasking of the channels (e.g. XMCM is nonzero).

Figure 34-92. XCERE Register

15	14	13	12	11	10	9	8
XCERE							
R/W-0h							
7	6	5	4	3	2	1	0
XCERE							
R/W-0h							

Table 34-104. XCERE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	XCERE	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel selection mode is selected with the XMCM bits. Reset type: SYSRSn 0h (R/W) = For multichannel selection when XMCM = 01b (all channels disabled unless selected): Disable and mask the channel that is mapped to XCEx. For multichannel selection when XMCM = 10b (all channels enabled but masked unless selected): Mask the channel that is mapped to XCEx. For multichannel selection when XMCM = 11b (all channels masked unless selected): Mask the channel that is mapped to XCEx. Even if the channel is enabled by the corresponding receive channel enable bit, this channel's data cannot appear on the DX pin. 1h (R/W) = For multichannel selection when XMCM = 01b (all channels disabled unless selected): Enable and unmask the channel that is mapped to XCEx. For multichannel selection when XMCM = 10b (all channels enabled but masked unless selected): Unmask the channel that is mapped to XCEx. For multichannel selection when XMCM = 11b (all channels masked unless selected): Unmask the channel that is mapped to XCEx. If the channel is also enabled by the corresponding receive channel enable bit, full transmission can occur.

34.16.2.27 XCERF Register (Offset = 1Ah) [Reset = 0h]

XCERF is shown in Figure 34-93 and described in Table 34-105.

Return to the Summary Table.

XCERF contains the transmit channel enable registers for the F partition. This register is only used when the transmitter is configured to allow individual disabling/enabling and masking/unmasking of the channels (e.g. XMCM is nonzero).

Figure 34-93. XCERF Register

15	14	13	12	11	10	9	8
XCERF							
R/W-0h							
7	6	5	4	3	2	1	0
XCERF							
R/W-0h							

Table 34-105. XCERF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	XCERF	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel selection mode is selected with the XMCM bits. Reset type: SYSRSn 0h (R/W) = For multichannel selection when XMCM = 01b (all channels disabled unless selected): Disable and mask the channel that is mapped to XCEx. For multichannel selection when XMCM = 10b (all channels enabled but masked unless selected): Mask the channel that is mapped to XCEx. For multichannel selection when XMCM = 11b (all channels masked unless selected): Mask the channel that is mapped to XCEx. Even if the channel is enabled by the corresponding receive channel enable bit, this channel's data cannot appear on the DX pin. 1h (R/W) = For multichannel selection when XMCM = 01b (all channels disabled unless selected): Enable and unmask the channel that is mapped to XCEx. For multichannel selection when XMCM = 10b (all channels enabled but masked unless selected): Unmask the channel that is mapped to XCEx. For multichannel selection when XMCM = 11b (all channels masked unless selected): Unmask the channel that is mapped to XCEx. If the channel is also enabled by the corresponding receive channel enable bit, full transmission can occur.

34.16.2.28 RCERG Register (Offset = 1Bh) [Reset = 0h]

RCERG is shown in Figure 34-94 and described in Table 34-106.

Return to the Summary Table.

RCERG contains the receive channel enable registers for the G partition. This register is only used when the receiver is configured to allow individual disabling/enabling and masking/unmasking of the channels (e.g. RMCM is nonzero).

Figure 34-94. RCERG Register

15	14	13	12	11	10	9	8
RCEG							
R/W-0h							
7	6	5	4	3	2	1	0
RCEG							
R/W-0h							

Table 34-106. RCERG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RCEG	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1): Reset type: SYSRSn 0h (R/W) = Disable the channel that is mapped to RCEx. 1h (R/W) = Enable the channel that is mapped to RCEx.

34.16.2.29 RCERH Register (Offset = 1Ch) [Reset = 0h]

RCERH is shown in Figure 34-95 and described in Table 34-107.

Return to the Summary Table.

RCERH contains the receive channel enable registers for the H partition. This register is only used when the receiver is configured to allow individual disabling/enabling and masking/unmasking of the channels (e.g. RMCM is nonzero).

Figure 34-95. RCERH Register

15	14	13	12	11	10	9	8
RCEH							
R/W-0h							
7	6	5	4	3	2	1	0
RCEH							
R/W-0h							

Table 34-107. RCERH Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RCEH	R/W	0h	Receive channel enable bit. For receive multichannel selection mode (RMCM = 1): Reset type: SYSRSn 0h (R/W) = Disable the channel that is mapped to RCEx. 1h (R/W) = Enable the channel that is mapped to RCEx.

34.16.2.30 XCERG Register (Offset = 1Dh) [Reset = 0h]

XCERG is shown in Figure 34-96 and described in Table 34-108.

Return to the Summary Table.

XCERG contains the transmit channel enable registers for the G partition. This register is only used when the transmitter is configured to allow individual disabling/enabling and masking/unmasking of the channels (e.g. XMCM is nonzero).

Figure 34-96. XCERG Register

15	14	13	12	11	10	9	8
XCERG							
R/W-0h							
7	6	5	4	3	2	1	0
XCERG							
R/W-0h							

Table 34-108. XCERG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	XCERG	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel selection mode is selected with the XMCM bits. Reset type: SYSRSn 0h (R/W) = For multichannel selection when XMCM = 01b (all channels disabled unless selected): Disable and mask the channel that is mapped to XCEx. For multichannel selection when XMCM = 10b (all channels enabled but masked unless selected): Mask the channel that is mapped to XCEx. For multichannel selection when XMCM = 11b (all channels masked unless selected): Mask the channel that is mapped to XCEx. Even if the channel is enabled by the corresponding receive channel enable bit, this channel's data cannot appear on the DX pin. 1h (R/W) = For multichannel selection when XMCM = 01b (all channels disabled unless selected): Enable and unmask the channel that is mapped to XCEx. For multichannel selection when XMCM = 10b (all channels enabled but masked unless selected): Unmask the channel that is mapped to XCEx. For multichannel selection when XMCM = 11b (all channels masked unless selected): Unmask the channel that is mapped to XCEx. If the channel is also enabled by the corresponding receive channel enable bit, full transmission can occur.

34.16.2.31 XCERH Register (Offset = 1Eh) [Reset = 0h]

XCERH is shown in Figure 34-97 and described in Table 34-109.

Return to the Summary Table.

XCERH contains the transmit channel enable registers for the H partition. This register is only used when the transmitter is configured to allow individual disabling/enabling and masking/unmasking of the channels (e.g. XMCM is nonzero).

Figure 34-97. XCERH Register

15	14	13	12	11	10	9	8
XCERH							
R/W-0h							
7	6	5	4	3	2	1	0
XCERH							
R/W-0h							

Table 34-109. XCERH Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	XCERH	R/W	0h	Transmit channel enable bit. The role of this bit depends on which transmit multichannel selection mode is selected with the XMCM bits. Reset type: SYSRSn 0h (R/W) = For multichannel selection when XMCM = 01b (all channels disabled unless selected): Disable and mask the channel that is mapped to XCEx. For multichannel selection when XMCM = 10b (all channels enabled but masked unless selected): Mask the channel that is mapped to XCEx. For multichannel selection when XMCM = 11b (all channels masked unless selected): Mask the channel that is mapped to XCEx. Even if the channel is enabled by the corresponding receive channel enable bit, this channel's data cannot appear on the DX pin. 1h (R/W) = For multichannel selection when XMCM = 01b (all channels disabled unless selected): Enable and unmask the channel that is mapped to XCEx. For multichannel selection when XMCM = 10b (all channels enabled but masked unless selected): Unmask the channel that is mapped to XCEx. For multichannel selection when XMCM = 11b (all channels masked unless selected): Unmask the channel that is mapped to XCEx. If the channel is also enabled by the corresponding receive channel enable bit, full transmission can occur.

34.16.2.32 MFFINT Register (Offset = 23h) [Reset = 0h]

MFFINT is shown in Figure 34-98 and described in Table 34-110.

Return to the Summary Table.

MFFINT contains the enable bits for both the transmitter and receiver interupts.

Figure 34-98. MFFINT Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					RINT	RESERVED	XINT
R-0h					R/W-0h	R-0h	R/W-0h

Table 34-110. MFFINT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	Reserved
2	RINT	R/W	0h	Enable for Receive Interrupt Reset type: SYSRSn 0h (R/W) = Receive interrupt on RRDY is disabled. 1h (R/W) = Receive interrupt on RRDY is enabled.
1	RESERVED	R	0h	Reserved
0	XINT	R/W	0h	Enable for Transmit Interrupt Reset type: SYSRSn 0h (R/W) = Transmit interrupt on XRDY is disabled. 1h (R/W) = Transmit interrupt on XRDY is enabled.

34.16.3 MCBSP Registers to Driverlib Functions

Table 34-111. MCBSP Registers to Driverlib Functions

File	Driverlib Function
DRR2	
mcbsp.h	McBSP_read32bitData
DRR1	
mcbsp.h	McBSP_read16bitData
mcbsp.h	McBSP_read32bitData
DXR2	
mcbsp.h	McBSP_write32bitData
DXR1	
mcbsp.h	McBSP_write16bitData
mcbsp.h	McBSP_write32bitData
SPCR2	
mcbsp.h	McBSP_setEmulationMode
mcbsp.h	McBSP_resetFrameSyncLogic
mcbsp.h	McBSP_enableFrameSyncLogic
mcbsp.h	McBSP_resetSampleRateGenerator
mcbsp.h	McBSP_enableSampleRateGenerator
mcbsp.h	McBSP_setTxInterruptSource
mcbsp.h	McBSP_getTxErrorStatus
mcbsp.h	McBSP_clearTxFrameSyncError
mcbsp.h	McBSP_isTxReady

Table 34-111. MCBSP Registers to Driverlib Functions (continued)

File	Driverlib Function
mcbsp.h	McBSP_resetTransmitter
mcbsp.h	McBSP_enableTransmitter
SPCR1	
mcbsp.h	McBSP_disableLoopback
mcbsp.h	McBSP_enableLoopback
mcbsp.h	McBSP_setRxSignExtension
mcbsp.h	McBSP_setClockStopMode
mcbsp.h	McBSP_disableDxPinDelay
mcbsp.h	McBSP_enableDxPinDelay
mcbsp.h	McBSP_setRxInterruptSource
mcbsp.h	McBSP_clearRxFrameSyncError
mcbsp.h	McBSP_getRxErrorStatus
mcbsp.h	McBSP_isRxReady
mcbsp.h	McBSP_resetReceiver
mcbsp.h	McBSP_enableReceiver
RCR2	
mcbsp.c	McBSP_setRxDataSize
mcbsp.h	McBSP_disableTwoPhaseRx
mcbsp.h	McBSP_enableTwoPhaseRx
mcbsp.h	McBSP_setRxCompandingMode
mcbsp.h	McBSP_disableRxFrameSyncErrorDetection
mcbsp.h	McBSP_enableRxFrameSyncErrorDetection
mcbsp.h	McBSP_setRxDataDelayBits
RCR1	
mcbsp.c	McBSP_setRxDataSize
XCR2	
mcbsp.c	McBSP_setTxDataSize
mcbsp.h	McBSP_disableTwoPhaseTx
mcbsp.h	McBSP_enableTwoPhaseTx
mcbsp.h	McBSP_setTxCompandingMode
mcbsp.h	McBSP_disableTxFrameSyncErrorDetection
mcbsp.h	McBSP_enableTxFrameSyncErrorDetection
mcbsp.h	McBSP_setTxDataDelayBits
XCR1	
mcbsp.c	McBSP_setTxDataSize
SRGR2	
mcbsp.h	McBSP_setFrameSyncPulsePeriod
mcbsp.h	McBSP_disableSRGSyncFSR
mcbsp.h	McBSP_enableSRGSyncFSR
mcbsp.h	McBSP_setRxSRGClockSource
mcbsp.h	McBSP_setTxSRGClockSource
mcbsp.h	McBSP_setTxInternalFrameSyncSource
SRGR1	
mcbsp.h	McBSP_setFrameSyncPulseWidthDivider
mcbsp.h	McBSP_setSRGDataClockDivider

Table 34-111. MCBSP Registers to Driverlib Functions (continued)

File	Driverlib Function
MCR2	
mcbsp.h	McBSP_setTxMultichannelPartition
mcbsp.h	McBSP_setTxTwoPartitionBlock
mcbsp.h	McBSP_getTxActiveBlock
mcbsp.h	McBSP_setTxChannelMode
MCR1	
mcbsp.h	McBSP_setRxMultichannelPartition
mcbsp.h	McBSP_setRxTwoPartitionBlock
mcbsp.h	McBSP_getRxActiveBlock
mcbsp.h	McBSP_setRxChannelMode
RCERA	
mcbsp.c	McBSP_disableRxChannel
mcbsp.c	McBSP_enableRxChannel
RCERB	
-	See RCERA
XCERA	
mcbsp.c	McBSP_disableTxChannel
mcbsp.c	McBSP_enableTxChannel
XCERB	
-	See XCERA
PCR	
mcbsp.h	McBSP_setRxSRGClockSource
mcbsp.h	McBSP_setTxSRGClockSource
mcbsp.h	McBSP_setTxFrameSyncSource
mcbsp.h	McBSP_setRxFrameSyncSource
mcbsp.h	McBSP_setTxClockSource
mcbsp.h	McBSP_setRxClockSource
mcbsp.h	McBSP_setTxFrameSyncPolarity
mcbsp.h	McBSP_setRxFrameSyncPolarity
mcbsp.h	McBSP_setTxClockPolarity
mcbsp.h	McBSP_setRxClockPolarity
RCERC	
-	See RCERA
RCERD	
-	See RCERA
XCERC	
-	See XCERA
XCERD	
-	See XCERA
RCERE	
-	See RCERA
RCERF	
-	See RCERA
XCERE	
-	See XCERA

Table 34-111. MCBSP Registers to Driverlib Functions (continued)

File	Driverlib Function
XCERF	
-	See XCERA
RCERG	
-	See RCERA
RCERH	
-	See RCERA
XCERG	
-	See XCERA
XCERH	
-	See XCERA
MFFINT	
mcbsp.h	McBSP_enableRxInterrupt
mcbsp.h	McBSP_disableRxInterrupt
mcbsp.h	McBSP_enableTxInterrupt
mcbsp.h	McBSP_disableTxInterrupt

Chapter 35

Power Management Bus Module (PMBus)

This chapter describes the features and operation of the Power Management Bus (PMBus) module.

35.1 Introduction

The PMBus module provides an interface between the microcontroller and devices compliant with the SMI Forum PMBus Specification Part I version 1.0 and Part II version 1.1. The PMBus is based on SMBus, which uses a similar physical layer to the I2C. This chapter assumes you are familiar with the PMBus, SMBus, and I2C bus specifications.

35.1.1 PMBUS Related Collateral

Foundational Materials

- C2000 Academy - PMBUS
- Seven things to know about PMBus (Video)

Getting Started Materials

- C28x PMBus Communications Stack User's Guide Application Report
- *Software Implementation of PMBus over I2C for TMS320F2803x Application Report*

Expert Materials

- 9 things you need to know about PMBus Point-of-Load Power (Video)

35.1.2 Features

The PMBus module has the following features:

- Compliance with the SMI Forum PMBus Specification (Part I v1.0 and Part II v1.1)
- Support for master and slave modes
- Support for I2C modes
- Support for three speeds:
  - Standard Mode: Up to 100 kHz
  - Fast Mode: Up to 400 kHz
- Packet error checking
- CONTROL and ALERT signals
- Clock high and low time-outs
- Four-byte transmit and receive buffers
- One maskable interrupt, which can be generated by several conditions:
  - Receive data ready
  - Transmit buffer empty
  - Slave address received
  - End of message
  - ALERT input asserted
  - Clock low time-out
  - Clock high time-out
  - Bus free

35.1.3 Block Diagram

Figure 35-1 shows the block diagram for PMBus. The PMBus module handles the lower levels of the PMBus protocol. In addition to controlling signal levels and timing, parsing addresses, and buffering data, the PMBus module also directly supports complex transactions such as Read Word and Process Call.

There are four PMBus signals:

- **SCL** is the bus clock. SCL is normally controlled by the master, but can be held low by a slave to delay a transaction and allow more time for processing.
- **SDA** is the bidirectional data line.
- **CONTROL** is a slave input that can trigger an interrupt. CONTROL can be used to shut down a slave device.
- **ALERT** is a slave output/master input that allows a slave to request attention from the master.

The SDA and SCL timings produced by the module are derived from SYSCLK. To comply with the PMBus timing specs, the bit clock divider must be set by way of the PMBTIMCLK register to provide a bit clock of 10 MHz or less.

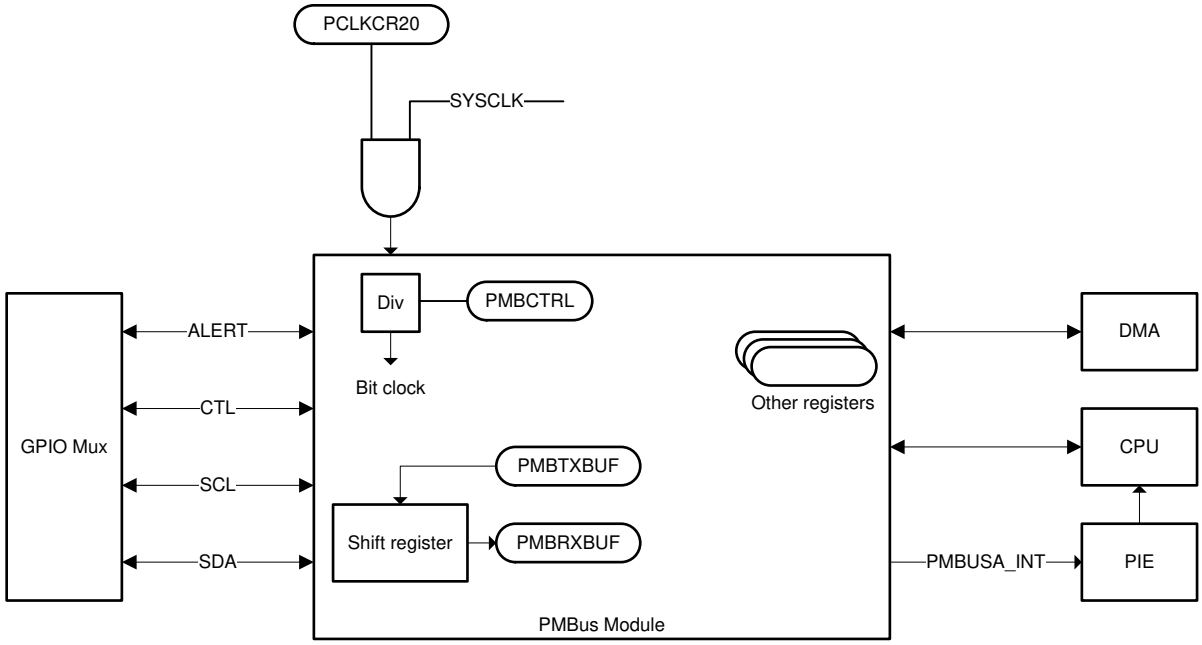


Figure 35-1. PMBus Module Block Diagram

35.2 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

Some IO functionality is defined by GPIO register settings independent of this peripheral. For input signals, the GPIO input qualification is set to asynchronous mode by setting the appropriate GPxQSELn register bits to 11b. The internal pullups are configured in the GPyPUD register.

Note

The GPIO configuration register GPyODR must be set to normal mode when the PMBus is used. The open-drain operation for PMBus is managed by the PMBus module

See the *General-Purpose Input/Output (GPIO)* chapter for more details on GPIO mux and settings.

35.3 Slave Mode Operation

This section describes the configuration and operation of the PMBus module in slave mode.

35.3.1 Configuration

To configure the module, write a clock divider to the PMBCTRL register's CLKDIV field to produce a bit clock frequency of less than 10 MHz. To activate slave mode, set the SLAVE_EN bit in the PMBCTRL register. Next, set up the PMBSC register. The following options are configurable:

- Slave address and mask (SLAVE_ADDR and SLAVE_MASK): Sets the slave address and mask for message acceptance.
- Manual slave address acknowledgment (MAN_SLAVE_ACK): When enabled, allows software to decide whether to acknowledge (ACK) an address. When disabled, the decision to ACK is made automatically based on the slave address and mask.
- PEC enable (PEC_ENA): Set this bit if Packet Error Checking (PEC) is used on the bus.
- Manual command byte acknowledgment (MAN_CMD): Similar to manual slave acknowledgment, setting this bit allows software to decide whether to acknowledge (ACK) a command byte.
- Number of bytes to acknowledge automatically (RX_BYTE_ACK_CNT): This is normally set to the maximum value, which allows the entire receive buffer to be used. However, smaller values can be used if the application requires that erroneous messages be detected and not acknowledged (NACKed) as soon as possible.

Manual acknowledgment is done by writing a one to the PMBACK register. Even with automatic acknowledgment, some writes to PMBACK are required. If the message (not including the address) is longer than 4 bytes, each packet of 4 bytes much be acknowledged. The PMBus module stretches the clock (hold the clock low) until an ACK is issued. The module then pulls the data line low and releases the clock, providing the ACK signal to the master.

If the complete message or the last part of the message is less than 4 bytes (or the RX_BYTE_ACK_CNT limit), do not write to PMBACK.

Writing a zero to the PMBACK bit sends a NACK. This can only be done when the module is waiting for an acknowledgment. If a zero is written at any other time, the NACK is issued during the next message.

35.3.2 Message Handling

This section describes some of the message types for PMBus and how to determine which message type is being received in slave mode. This section is oriented toward the most efficient mode of operation – with automatic address and command acknowledgment and is also oriented toward having Packet Error Checking (PEC) enabled.

If automatic address acknowledgment is disabled, all messages start by setting the SLAVE_ADDR_READY bit high. Read commands have two instructions one for the read and one for the write. If automatic command acknowledgment is enabled, the DATA READY bit is set high, as well. If the message has no PEC, the number of bytes available is n-1. For example with PEC, a QUICK COMMAND has one byte. With no PEC, a QUICK COMMAND has zero bytes.

Note that the byte count does not increment as bytes arrive. No bits are set in the PMBST register until a stop message is received, the receive buffer is full, or a fault occurs. Then, all appropriate bit values are placed in the register together. All that is necessary to receive a quick command is to ACK the message by writing a one to the PMBACK register.

35.3.2.1 Quick Command

Quick Commands (Figure 35-2) received by the PMBus module in slave mode require a simple acknowledgment of the received device address. In automatic address acknowledge mode, the module processes the quick command without firmware interaction. Upon receipt of the end of message, the firmware has the option to read the received address in the PMBHSA register. In manual address acknowledge mode, the address is acknowledged by writing to the PMBACK register.

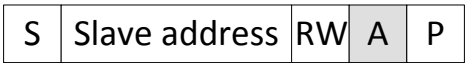


Figure 35-2. Quick Command Message

35.3.2.2 Send Byte

A Send Byte message (Figure 35-3) consists of the device address, a single data byte, and an optional PEC byte. To process the PEC byte correctly, PEC processing must be enabled in the PMBSC register. In automatic address acknowledge mode, the data and optional PEC byte are acknowledged without firmware interaction. The module generates an End of Message interrupt, reads the status register and finds the data ready indication bit set. In manual mode, the address is acknowledged by the firmware, while the remaining data and PEC bytes are acknowledged by the module.

The PMBus module stores Data Byte #0 into the PMBRXBUF register. The data byte is stored into bits 7-0. In non-PEC mode, the RX Byte Count in the PMBSTS register indicates one byte received. If PEC processing is enabled, the PEC byte is also stored into the PMBRXBUF register, with the PEC byte residing in bits 15-8. The RX Byte Count in the PMBSTS register indicates two bytes received. The PEC Valid bit in the PMBSTS register indicates the validity of the received PEC byte.

When a Send Byte message is received, the Data Ready bit is set along with the EOM and, assuming that the PEC is valid, the PEC valid bit. The read byte count (RD_BYTE_COUNT) register contains a 2. All that is necessary to receive a send byte command is to ACK the message by writing a 1 to the PMBACK register. Before doing the ACK, read the byte from the lowest byte of the PMBRXBUF register.

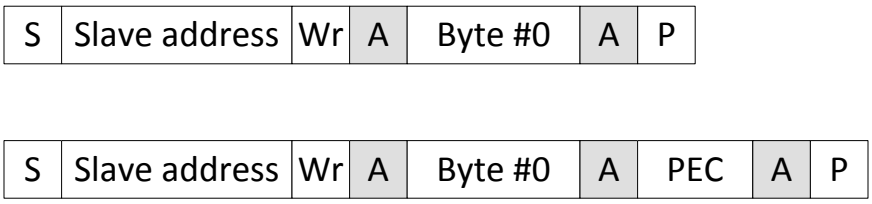


Figure 35-3. Send Byte Message With and Without PEC

35.3.2.3 Receive Byte

A Receive Byte message (Figure 35-4) consists of the device address, a single data byte, and an optional PEC byte. In automatic address acknowledge mode, the firmware receives a data request interrupt following reception of the slave address. The data byte to be sent to the master is stored into bits 7-0 of the PMBTXBUF register and Transmit Byte Count bits within the PMBSC register are set to a value of 1. If PEC processing is enabled, the Transmit PEC bit (bit 19) within the PMBSC register is set to 1, along with the Enable PEC bit (bit 15). The module automatically appends the calculated PEC byte at the completion of the message.

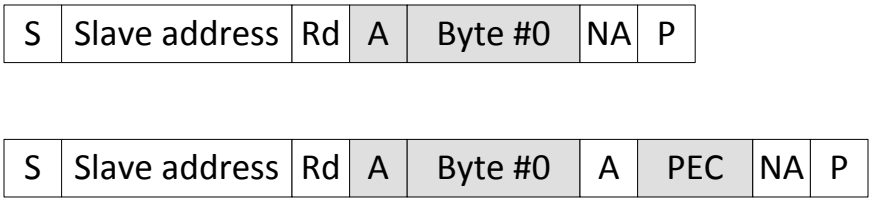


Figure 35-4. Receive Byte Message With and Without PEC

35.3.2.4 Write Byte and Write Word

The Write Byte and Write Word messages (Figure 35-5) consist of a slave address, a command word, transmitted data bytes and an optional PEC byte. In automatic address acknowledge mode, the data bytes and optional PEC byte are acknowledged without firmware interaction. The acknowledgment of the command word is configured through the PMBSC register. The firmware receives an End of Message interrupt in all cases except for Write Word with PEC message, reads the status register and finds the data ready indication bit set.

In the case of a Write Word with PEC byte message, the data ready interrupt is enabled after receiving 4 bytes (command byte, the 2 data bytes and the PEC byte). The firmware reads the data from the PMBRXBUF register and must write the PMBACK register to acknowledge back to the master. The PMBus module holds SCL low until the firmware responds to the received data.

In all other cases, the EOM interrupt is received and data can be read from the PMBRXBUF register. The firmware is not required to send an acknowledgment back to the master.

The Write Byte message looks exactly the same as the Send Byte, except the RD_BYTE_COUNT register contains a 3. The Write Word message has a RD_BYTE_COUNT of 4.

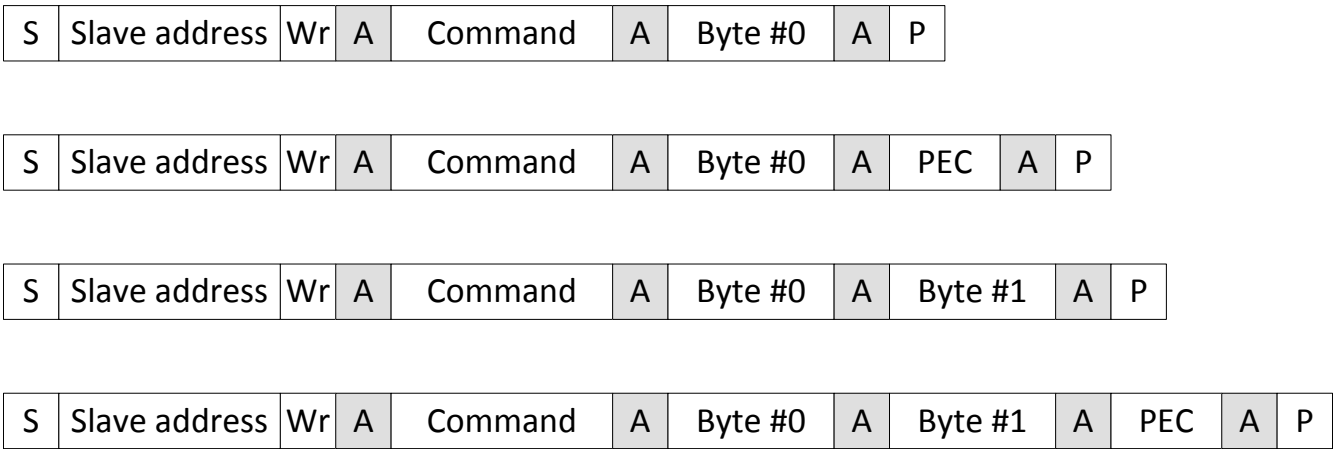


Figure 35-5. Write Byte and Write Word Messages With and Without PEC

35.3.2.5 Read Byte and Read Word

The Read Byte and Read Word messages (Figure 35-6) consist of a slave address, a command word, received data bytes from a slave, and an optional PEC byte. Address and command acknowledgment is configured through the PMBSC register. In automatic mode, the PMBus module provides a data ready and data request interrupt following receipt of a repeated start and slave address. The received command byte is found in bits 7-0 of the PMBRXBUF register. The firmware responds to the data request by programming the data bytes into the PMBTXBUF register and the TX Byte Count bits in the PMBSC register. If PEC processing is enabled, the Transmit PEC bit should also be asserted. An EOM interrupt indicates completion of the message to the Master.

When the repeated start (Sr) signal is received, the Data Ready bit is asserted with a RD_BYTE_COUNT of 1. At this point, the operation cannot be distinguished from a group command Send Byte message. When the same device address is sent out with a read, the Data Request bit is asserted. If data has already been written to the PMTXBUF register before the Device Address is received, the Data Request bit is not asserted. So if group commands are also expected, it is necessary to read the Data Ready with a RD_BYTE_COUNT of 1, and then wait and see whether the next event is an EOM or a Data Request. If the event is an EOM, the command should be processed as a group send byte. If the event is a Data Request, the command should be processed as a read. Depending on the command, the event can be a read byte, word, or block. If the PMBus module is polled, both the Data Ready and the Data Request bits could possibly be set between polling intervals. This should be considered in the design of the firmware.

Once the read command is recognized, you must respond by writing data to the PMBTXBUF register. It is also necessary to make sure that the values in the PMBSC register are correct. The transmit byte count and PEC bit must be set appropriately. For a read byte, the transmit byte count can be loaded with a 1. If the transmission of a PEC byte is desired, the TX_PEC bit must be set. After this, the data can be written to PMBTXBUF, which starts the transmission. All bytes should be written to PMBTXBUF at the same time. After the master receives the message, it NACKs the last byte to indicate that the correct number of bytes have been received. This causes the EOM bit to be set in the PMBSTS register, indicating to the firmware that the Read Byte message is complete.

Figure 35-6. Read Byte and Read Word Messages With and Without PEC

35.3.2.6 Process Call

The Process Call (Figure 35-7) protocol consists of a Write Word message, followed by a Read Word message, without a stop condition between the two messages. Address and command acknowledgment is configured through the PMBSC register. In automatic mode, following receipt of the repeated start and slave address, the PMBus module provides a data ready and a data request interrupt. The repeated start bit is set in the PMBSTS register to indicate the receipt of the first part of the Process Call message. The received command byte is found in bits 7-0 of the PMBRXBUF register, while the two data bytes received from the master can be found in bits 23-8. Upon receipt of the repeated start and a data request from the module, the firmware programs the PMBTXBUF with the 2 data bytes to be sent to the master. If PEC processing is enabled, the Transmit PEC bit within the PMBSC register is asserted. The EOM interrupt indicates the read word portion of the Process Call message has been completed by the module.

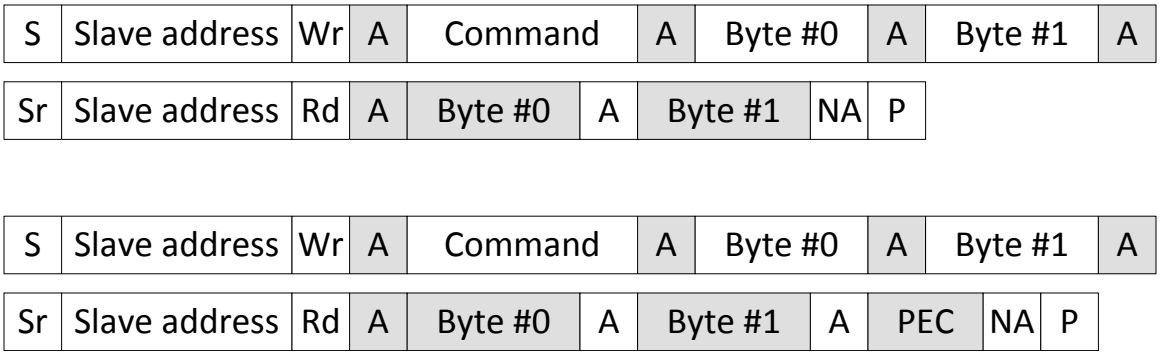


Figure 35-7. Process Call Message With and Without PEC

35.3.2.7 Block Write

The Block Write (Figure 35-8) protocol is similar to Write Word in structure, except that there are more than 2 data bytes in the message. Following the receipt of the command byte, the block length and 2 data bytes, the PMBus module provides a data ready interrupt. The module waits for the firmware to read the received data and program the acknowledge register. While waiting for an ACK from the firmware, the module drives the clock line low, stalling the bus. The data ready interrupts continue for the duration of the message at a frequency of every 4 data bytes. The number of bytes received can be found within the PMBSTS register. At the end of the message, less than 4 bytes can be stored in the PMBRXBUF register. The PEC Valid bit can be checked to determine if the received PEC value is accurate.

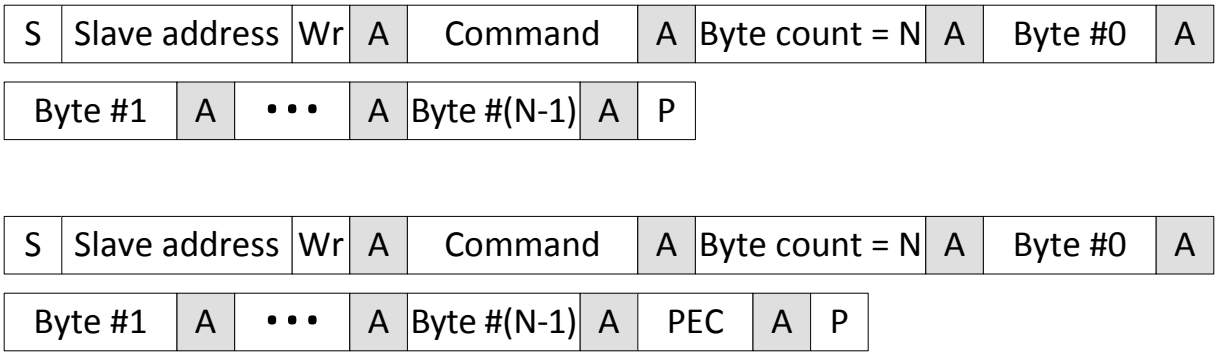


Figure 35-8. Block Write Message With and Without PEC

35.3.2.8 Block Read

The Block Read (Figure 35-9) protocol is similar to a Read Word in structure, except that there are more than 2 data bytes in the message. Following the receipt of the repeated slave address, a data ready and data request interrupt is generated by the PMBus module. The command byte received from the master can be found in bits 7-0 of the PMBRXBUF register. The SCL line is held low until the firmware programs data bytes into the PMBTXBUF register. The firmware is required to load the block length into bits 7-0 of the PMBTXBUF register during the initial programming of the register. After 4 bytes have been transmitted, the module issues a data request interrupt and holds SCL low again until the firmware has programmed additional data into the PMBTXBUF register.

Block read starts the same as Read Word or Read Byte, but TX_COUNT is loaded with a 4 the first time, and TX_PEC is not set. Instead of waiting for an EOM after the first transmission, the firmware instead waits for a Data Request, indicating that the master is ready for more data. Until the last 4 or less bytes, the firmware simply writes a 4 to TX_COUNT and then writes the 4 bytes to PMBTXBUF. TX_PEC is left cleared. Then when the last 4 or fewer bytes are to be transmitted, the firmware writes out the appropriate byte count, sets the TX_PEC bit, and writes the data to PMBTXBUF. The PMBus module writes out the data, followed by the PEC, and then the EOM bit is set when the master NACKs the PEC.

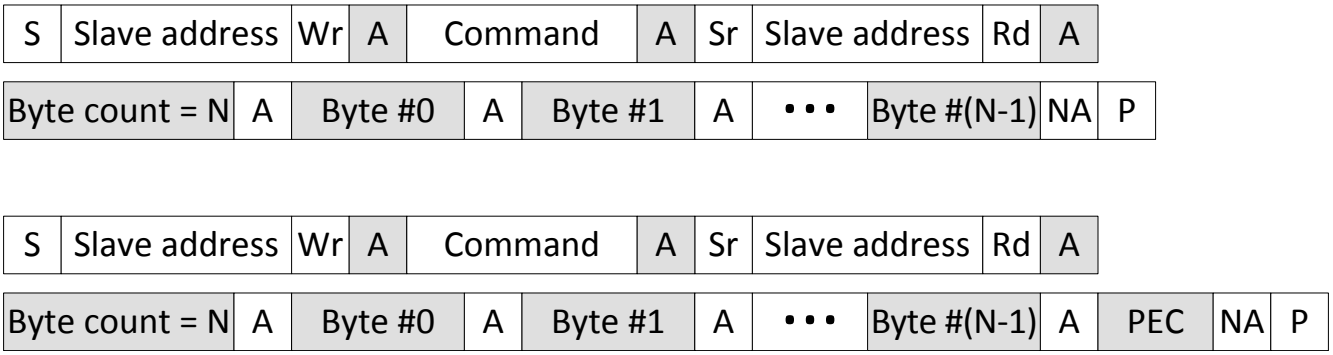


Figure 35-9. Block Read Message With and Without PEC

35.3.2.9 Block Write-Block Read Process Call

The Block Write-Block Read Process Call (Figure 35-10) protocol combines the Block Write and Block Read protocols, removing the stop condition between the two messages. The processing of the Block Read-Block Write Process Call message is similar to the mode of operation for the Process Call message. After acknowledgment of the address and command bytes, the PMBus module generates a data ready interrupt upon detection of 4 data bytes or a repeated start condition. After receiving the repeated start, the firmware is required to load transmit data to send to the master. Bits 7-0 of the initial programming of the PMBTXBUF register must represent the byte count of the block data sent to the master.

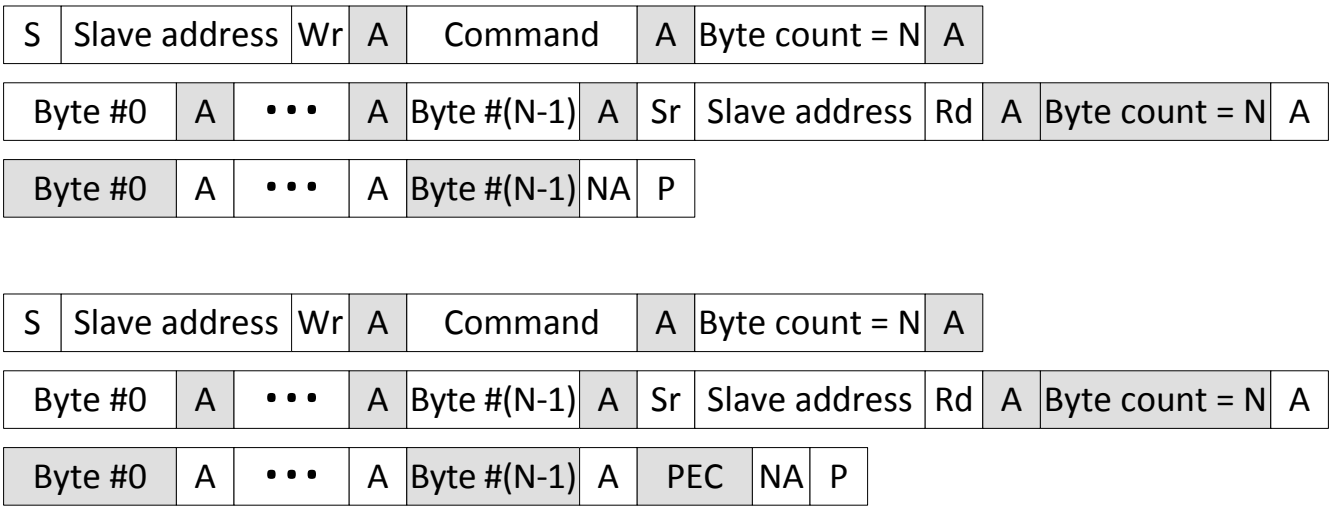


Figure 35-10. Block Write-Block Read Process Call Message With and Without PEC

35.3.2.10 Alert Response

The Alert Response Message (Figure 35-11) is utilized when the master detects an alert condition from a slave on the PMBus. In automatic address acknowledge mode, upon detection of the Alert Response Address, the PMBus module provides an acknowledgment to the master and sends the programmed slave address within the PMBSC register. The module only responds to the message if the Alert En bit within PMBCTRL register has been previously set. After receiving the Alert Response message, the module clears the alert condition and the enable bit within the PMBCTRL register.

In manual address acknowledge mode, the firmware must read the received address from the PMBRXBUF register and transmit the desired slave address back to the master. The PMBCTRL register must be reprogrammed to disable the Alert En bit used to initiate the Alert Response message from the master.

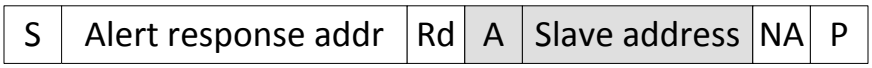


Figure 35-11. Alert Response Message

35.3.2.11 Extended Command

The PMBus module provides support for extended commands that allow for an extra 256 command codes. Both command bytes are stored in the PMBRXBUF register along with the data bytes. In recognizing the extended command messages, the Repeated Start bit and the Rd Byte Count Bits within the PMBSTS register are utilized. For Extended Command Write Byte and Write Word messages (Figure 35-12), the two command bytes are stored in bits 15-0 of the PMBRXBUF register. The initial command byte should hold the command extension code, representing utilization of the extended command protocol. The Repeated Start bit is also set, received after the retransmission of the device address. The Rd Byte Count equals 3 for an Ext Cmd Write Byte message and 4 for an Ext Cmd Write Word message.

For the Extended Command Read Byte and Read Word messages (Figure 35-13), the module generates a data ready and data request interrupt following reception of the repeated device address. The two command bytes are found in bits 15-0 of the PMBRXBUF register, with the initial command byte matching the command extension code. The firmware is required to load transmit data to complete the message back to the master.

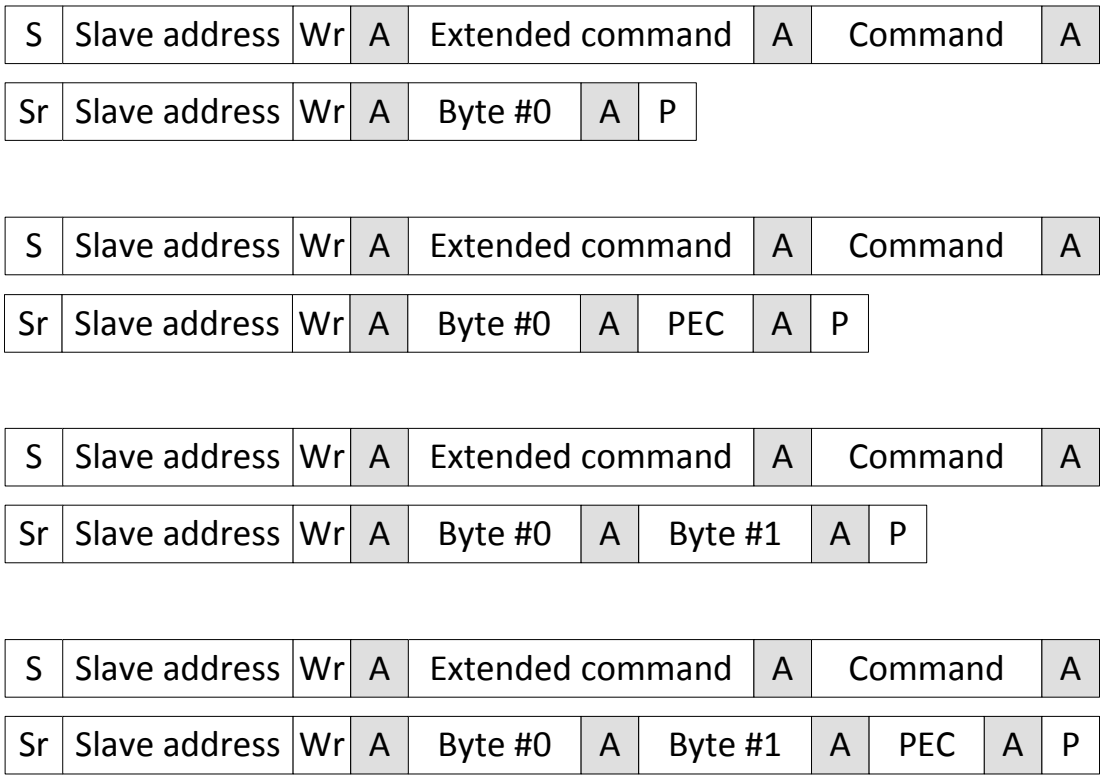


Figure 35-12. Extended Command Write Byte and Write Word Messages With and Without PEC

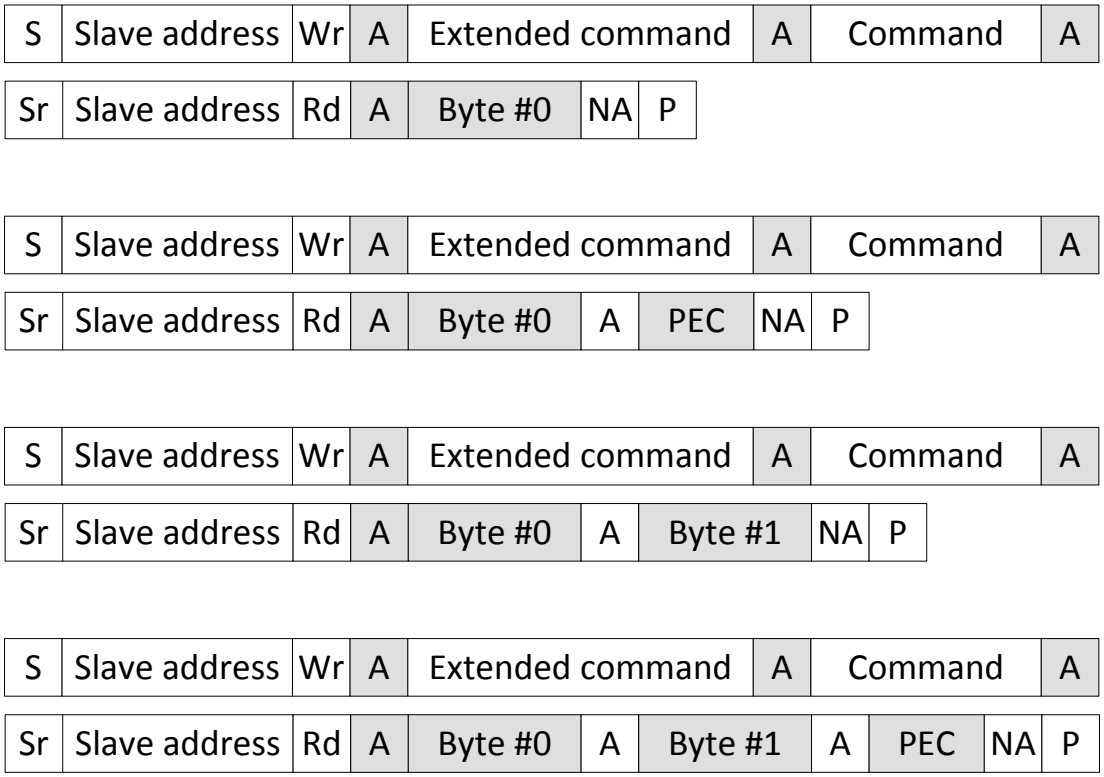


Figure 35-13. Extended Command Read Byte and Read Word Messages With and Without PEC

35.3.2.12 Group Command

The PMBus module supports the Group Command protocol. The Group Command (Figure 35-14) protocol is used to send commands to more than one device within the same message. When devices on the bus detect the stop condition at the conclusion of the Group Command message, the received commands are executed concurrently. Following address and command acknowledgment, the module provides a data ready interrupt upon detection of 4 data bytes or the transmission of a repeated start on the bus. The firmware should wait for the EOM interrupt before processing the received command, as required by the use of the Group Command message.

For Group Commands, the data ready bit is set as soon as the repeated start is received. The data can then be read into memory. But the data should not be acted upon until the EOM bit is set, which occurs when all of the messages have been received. Other than this delayed EOM, there is no difference for the slave firmware in receiving a Group Command than any other write message.

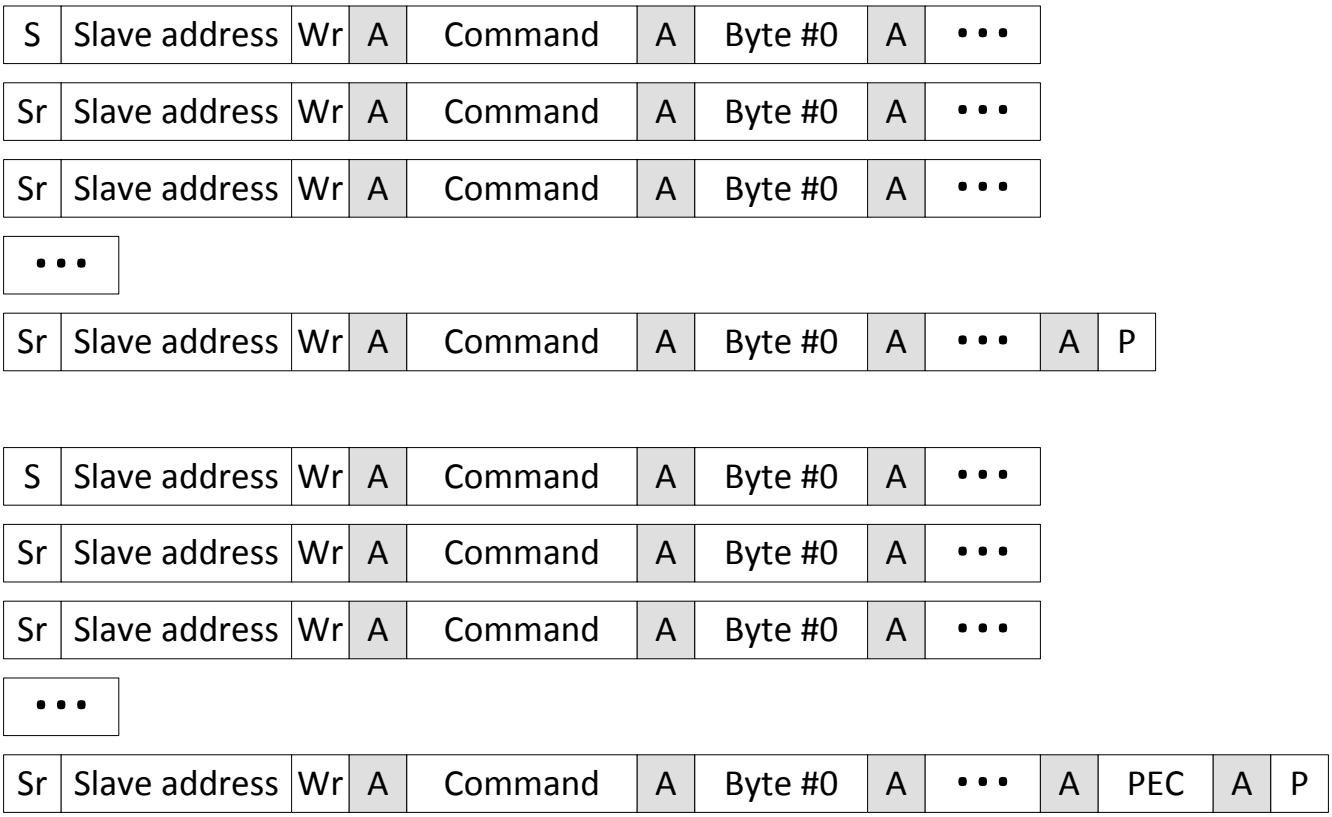


Figure 35-14. Group Command Message With and Without PEC

35.4 Master Mode Operation

This section describes the configuration and operation of the PMBus module in master mode.

35.4.1 Configuration

First, write a clock divider to the PMBCTRL register CLKDIV field to produce a bit clock frequency of less than 10 MHz. To activate master mode, set the MASTER_EN bit and clear the SLAVE_EN bit in the PMBCTRL register. For each transaction, set up the PMBMC register. The following options are configurable:

- Slave address (SLAVE_ADDR): Sets the slave address for the next transaction.
- PEC enable (PEC_ENA): If Packet Error Checking (PEC) is used on the bus, set this bit.
- Extended command code enable (EXT_CMD): When set, uses two bytes for commands.
- Command code enable (CMD_ENA): When set, sends a command byte at the start of the transaction.
- Byte count (BYTE_COUNT): Determines the number of data bytes to transfer. This does not include the block length byte, which is generated automatically when needed.
- Special command enables (GRP_CMD and PRC_CALL): Enables special behavior for group commands and process calls.

Writing to the PMBMC register starts a transfer.

Manual acknowledgment of received data is not needed.

35.4.2 Message Handling

This section describes the behavior and required configuration for each command type.

35.4.2.1 Quick Command

Quick commands (Figure 35-15) are initiated in master mode by simply programming the desired slave device address into the PMBMC. The byte count within the PMBMC register is configured to 0 bytes by writing all zeros to bits 15-8. Upon transmission of the device address, the PMBus module monitors the slave acknowledgment of the address. If the address is not acknowledged, the NACK bit within the status register is enabled and the PMBus module automatically sends a stop condition on the bus to terminate the message. If the address is acknowledged, a data request is issued to the processor. The firmware writes a zero to the PMBACK to terminate the message, forcing the PMBus modules to write a stop condition onto the bus.

Figure 35-15. Quick Command Message

35.4.2.2 Send Byte

A Send Byte message (Figure 35-16) consists of the device address, a single data byte, and an optional PEC byte. To initiate a Send Byte message, the data byte to be transmitted to the slave is loaded into bits 7-0 of the PMBTXBUF register. The PMBMC register is configured with the device address. To transmit a PEC byte with the message, the PEC_EN bit within the PMBMC register is asserted high when the address is programmed.

After programming the PMBMC register, the PMBus module transmits the Send Byte message. The firmware can wait for an End of Message interrupt from the PMBus module. Upon receipt of the EOM interrupt, the PMBSTS register is read to verify the slave properly acknowledged the transmitted data.

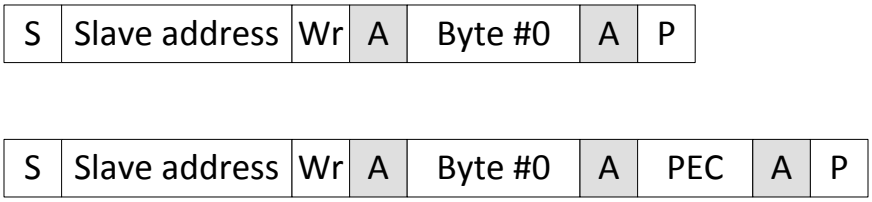


Figure 35-16. Send Byte Message With and Without PEC

35.4.2.3 Receive Byte

A Receive Byte message (Figure 35-17) consists of the device address, a single data byte, and an optional PEC byte. Data is being read from the slave in a Receive Byte message. To initiate a Receive Byte message, the firmware programs the device address, the R/W bit and the optional PEC_EN into the PMBMC register. The R/W bit is enabled high to indicate a read message type (data transmitted from slave to master).

After programming the PMBMC register, the PMBus module transmits the Receive Byte message. The firmware can wait for an End of Message interrupt from the PMBus module to verify the accuracy of the message transmission. Upon receipt of the EOM interrupt, the PMBSTS register is read to verify proper slave acknowledgment of the device address and to determine if any data is available for reading in the PMBRXBUF register. If PEC_EN was asserted in the PMBMC register, the PEC_VALID bit in the PMBSTS register is also checked to make sure a proper PEC byte was received from the slave with the received data.

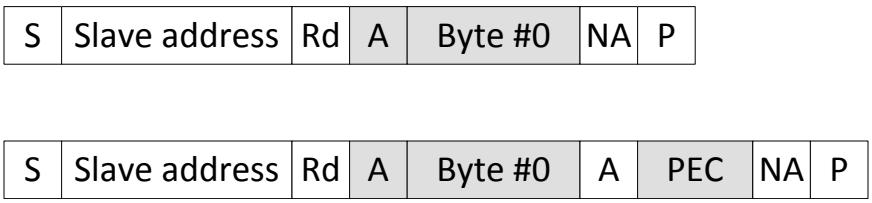


Figure 35-17. Receive Byte Message With and Without PEC

35.4.2.4 Write Byte and Write Word

The Write Byte and Write Word messages (Figure 35-18) consist of a device address, a command byte, transmitted data bytes, and an optional PEC byte. Write Byte messages include a single byte, while the Write Word messages support transmission of 2 bytes to the corresponding slave module. Similar to the Send Byte protocol, the PMBMC register is configured to send 1 or 2 bytes, the CMD_EN bit is set to enable command byte transmission and the optional PEC_EN bit is set.

With the command byte transmission enabled, the format of the PMBTXBUF register differs from the Send Byte protocol. In bits 7-0, the firmware must program the command byte to be sent to the slave. The data bytes are programmed into bits 15-8 and bits 23-16.

After programming the PMBMC register, the PMBus module transmits the Write Byte/Word message. The firmware can wait for an End of Message interrupt from the module to verify the accuracy of the message transmission. The PMBSTS register indicates if the slave acknowledged the message properly.

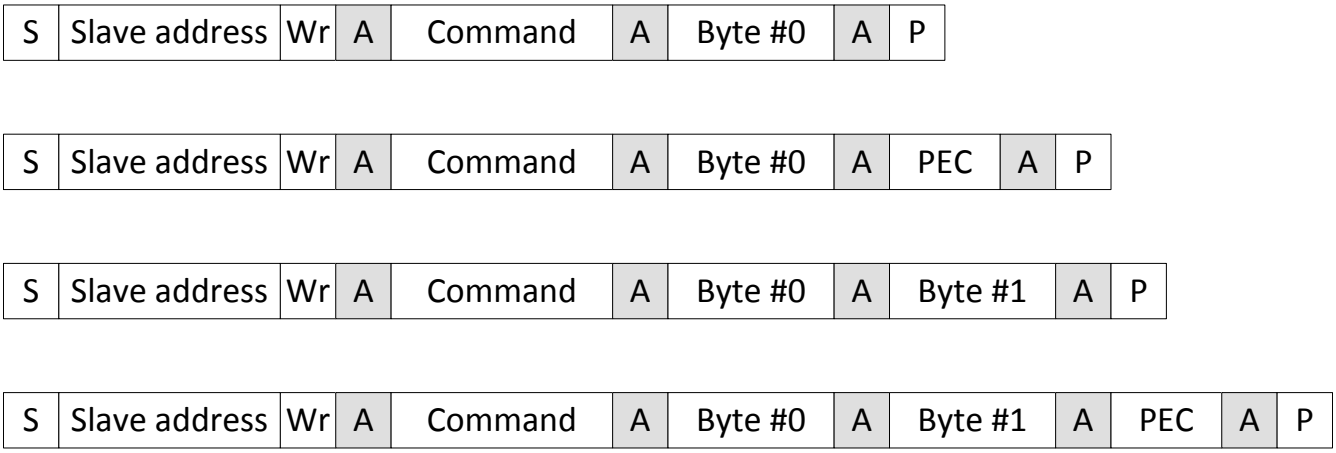


Figure 35-18. Write Byte and Write Word Messages With and Without PEC

35.4.2.5 Read Byte and Read Word

The Read Byte and Read Word messages (Figure 35-19) consist of a device address, a command byte, received data bytes from a slave, and an optional PEC byte. Read Byte messages include a single byte, while the Read Word message protocol supports receipt of 2 bytes from the slave. Similar to the Receive Byte Protocol, the PMBMC register is configured to receive 1 or 2 bytes, the CMD_EN bit is set and the PEC_EN is configured to expect or not expect a PEC byte appended to the message. The PMBus module automatically terminates the message after the expected number of bytes is received from the slave or if the slave does not properly acknowledge any portion of the message.

In addition to programming the PMBMC register, the firmware is expected to load the command byte into bits 7-0 of the PMBTXBUF register. Any data received from the slave is found in the PMBRXBUF register.

Figure 35-19. Read Byte and Read Word Messages With and Without PEC

35.4.2.6 Process Call

The Process Call (Figure 35-20) protocol consists of a Write Word message, followed by a Read Word message, without a stop condition between the two messages. A PEC byte can be appended to the read data from the slave as an option to the message protocol. The PMBMC register includes a PRC_CALL bit, which enables the transmission of a Process Call message onto the PMBus. The PMBus module automatically generates a repeated start condition and initiates the Read Word portion of the message when the process call bit is enabled.

To complete the Write Word portion of the Process Call, the PMBTXBUF register is loaded with the command byte in bits 7-0 and the data bytes are loaded into bits 23-8 of the register.

After programming the PMBMC register, the PMBus module transmits the Process Call Message. The firmware can wait for an End of Message interrupt from the module to determine the validity of the message. Upon the receipt of the EOM, the PMBSTS register can indicate the receipt of 2 bytes from the Read Word portion of the Process Call message and the status of the slave acknowledgment of the transmit data. If PEC processing is enabled, the PEC_VAL bit within the PMBSTS register indicates the accuracy of the PEC byte received from the slave during the Read Word part of the message.

The PRC_CALL bit within the PMBMC register must be disabled for the next non-Process Call message. Note that any write to the PMBMC register initiates a message, so reconfiguration of the master is not recommended until the firmware requires a new message to be transmitted.

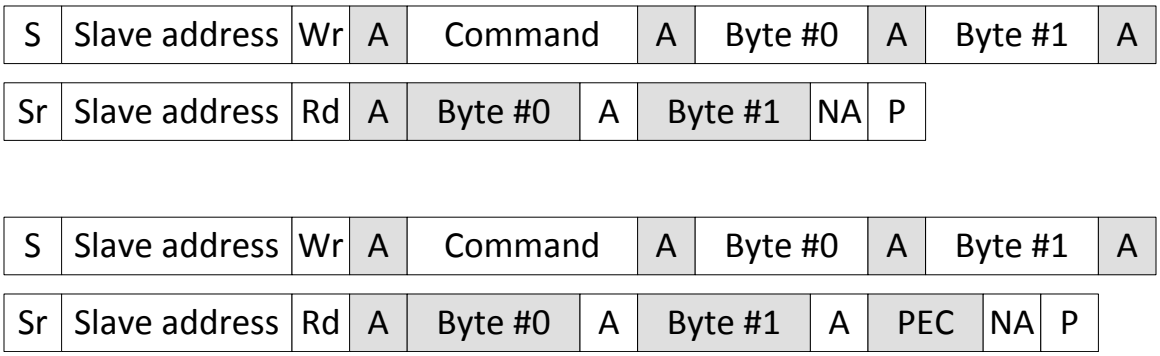


Figure 35-20. Process Call Message With and Without PEC

35.4.2.7 Block Write

The Block Write (Figure 35-21) protocol is similar to a Write Word in structure, with the exception of transmission of more than 2 data bytes in the message. Additionally, the first data byte following the command byte specifies the length of the block of data bytes. As with a majority of the message protocols, the PEC byte can be appended to the end of the write data to the slave.

To initiate a Block Write message on the bus, the PMBMC register is programmed with the block length in the Byte Count bits. The block length is the number of data bytes, excluding the command byte and the first data byte that contains the block length. The PMBus module automatically inserts the block length into the message, if the number of data bytes specified by the firmware exceeds 2. The initial write data is loaded into the PMBTXBUF register. With bits 7-0 representing the command byte, the remaining 3 bytes represent the first 3 data bytes following the block length.

Following programming of the PMBMC register, the Block Write message is transmitted. If the block length exceeds 3 bytes, the PMBus module provides a data request interrupt, indicating the need for additional data bytes in the PMBTXBUF register. The PMBus module assumes that if more than 4 bytes are needed to complete the message, the firmware utilizes all 4 bytes when programming the PMBTXBUF register. If less than 4 bytes are needed to finish the Block Write message, the firmware only needs to program the appropriate bits of the PMBTXBUF register.

Upon completion of the message, the PMBus module issues an EOM interrupt. The PMBSTS register can be checked to verify the slave accepted the block of write data.

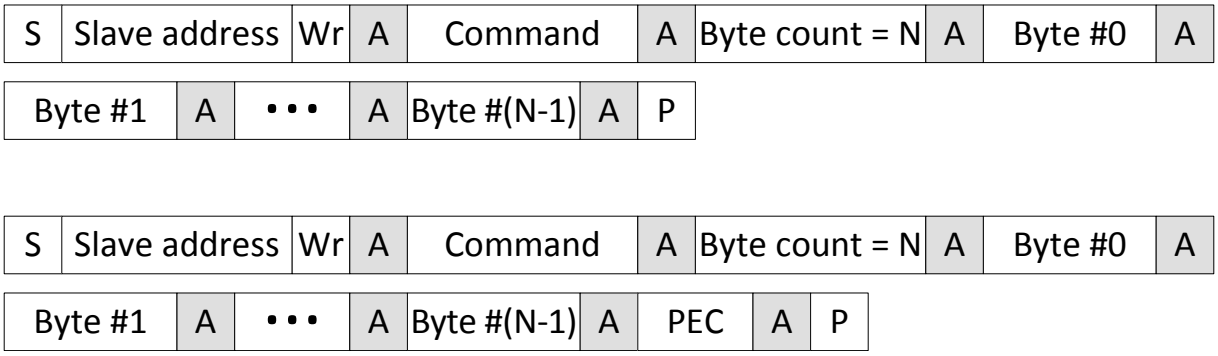


Figure 35-21. Block Write Message With and Without PEC

35.4.2.8 Block Read

The Block Read (Figure 35-22) protocol is similar to a Read Word in structure, with the exception that there are more than 2 data bytes received from the slave. The first data byte transmitted by the slave represents the block length of the data being written by the slave. If PEC processing is enabled, the slave appends a PEC byte to the end of the message.

To initiate a Block Read message on the PMBus, the PMBMC register is programmed with the block length in the Byte Count bits. This count excludes the command byte, any slave address and the block length bytes in the message. The command byte to be transmitted to the slave is written into bits 7-0 of the PMBTXBUF register prior to the programming of the PMBMC register.

After configuring the PMBMC register, the Block Read message is transmitted. The module interrupts the firmware upon receipt of 4 data bytes from the slave. If the block length is 3, the EOM interrupt is received concurrently with the data ready interrupt. Otherwise, only a data ready interrupt is asserted, indicating 4 bytes are ready for reading by the firmware. At the end of the message, less than 4 bytes can be stored in the PMBRXBUF register. The RX Byte Count bits in the PMBSTS register indicate the number of bytes available in the final data transfer. The firmware can verify the received PEC upon detection of the End of Message interrupt.

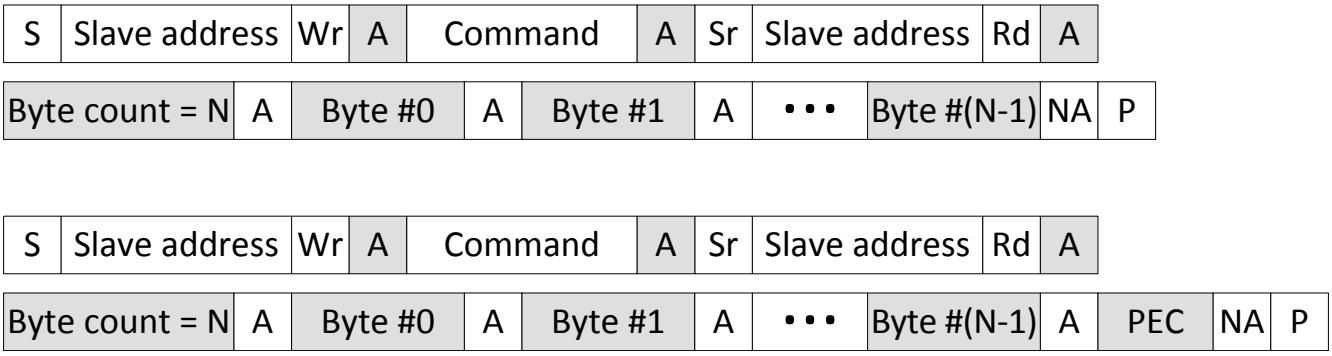


Figure 35-22. Block Read Message With and Without PEC

35.4.2.9 Block Write-Block Read Process Call

The Block Write-Block Read Process Call (Figure 35-23) protocol combines the Block Write and Block Read protocols, removing the stop condition between the two messages. The operation of the master is similar to a Block Write operation. Loading the block length into the byte count bits of the PMBMC register provides the length of the Block Write portion of the message. In addition, the PRC_CALL bit within the PMBMC register must be enabled. Upon completion of the Block Write part of the message, the PMBus module automatically issues a Repeated Start condition on the PMBus and starts transmission of the Block Read portion of the message. Operation of the PMBus module after the Repeated Start condition is the same as a simple Block Read Message.

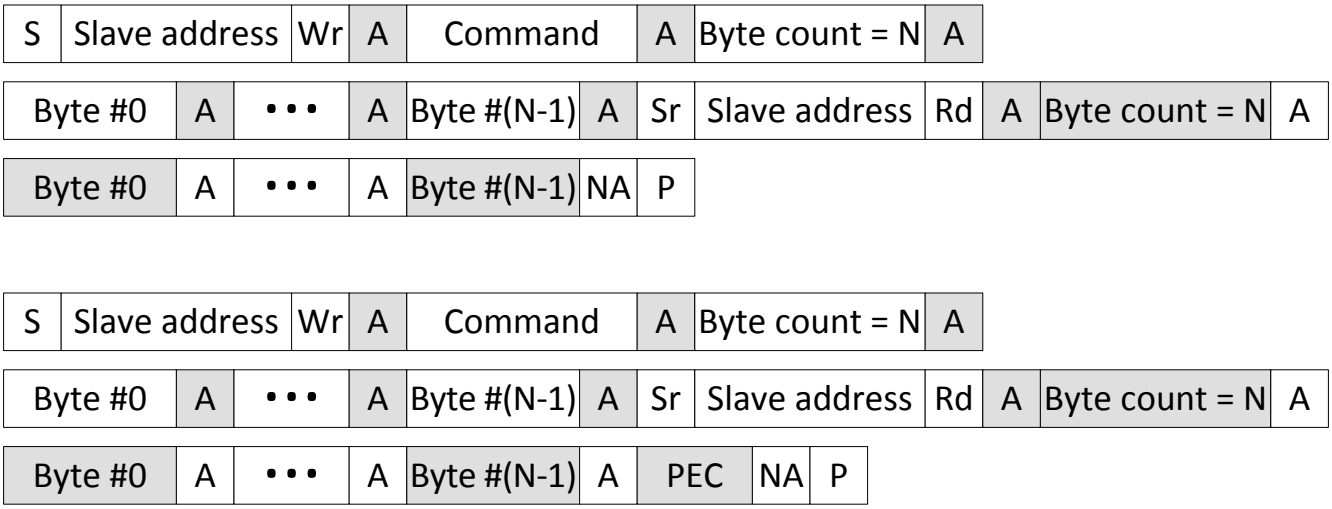


Figure 35-23. Block Write-Block Read Process Call Message With and Without PEC

35.4.2.10 Alert Response

The Alert Response Message (Figure 35-24) is utilized when the master detects an alert condition from a slave. In master mode, the Alert Response Message is simply a Receive Byte message with PEC disabled and the slave address set to 0xC (Alert Response address). The PMBus module detects the alert condition on an input and interrupts the firmware indicating the assertion of an alert condition (slave desires to communicate with the master). Programming the PMBMC register with the Alert Response address initiates the Alert Response message and provides the device address of the slave requesting service. The device address is found in the PMBRXBUF register following receipt of the EOM interrupt.

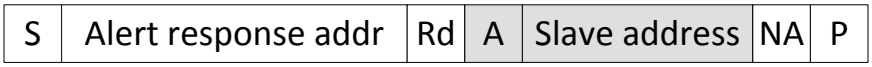


Figure 35-24. Alert Response Message

35.4.2.11 Extended Command

The PMBus module provides support for extended commands which allow for an extra 256 command codes. By asserting the EXT_CMD bit within the PMBMC register, two command bytes are transmitted on the message protocol. Extended commands can be added to the Write Byte and Write Word (Figure 35-25) and the Read Byte and Read Word (Figure 35-26) protocols. Operation of the PMBus module in extended command mode is similar to these formats. In programming the write data or first part of the read message, the second command byte is loaded into bits 15-8 of the PMBTXBUF register with the remaining data bytes. The remaining operation of the module is identical to the previous protocols, except for the inclusion of a Repeated Start condition and slave address in the write messages. No support is required by firmware for these additional bytes in the write messages. The module interprets the EXT_CMD bit and makes the appropriate format changes.

S	Slave address	Wr	A	Extended command	A	Command	A
Sr	Slave address	Wr	A	Byte #0	A	P	

S	Slave address	Wr	A	Extended command	A	Command	A
Sr	Slave address	Wr	A	Byte #0	A	PEC	A	P

S	Slave address	Wr	A	Extended command	A	Command	A
Sr	Slave address	Wr	A	Byte #0	A	Byte #1	A	P

S	Slave address	Wr	A	Extended command	A	Command	A
Sr	Slave address	Wr	A	Byte #0	A	Byte #1	A	PEC	A	P

Figure 35-25. Extended Command Write Byte and Write Word Messages With and Without PEC

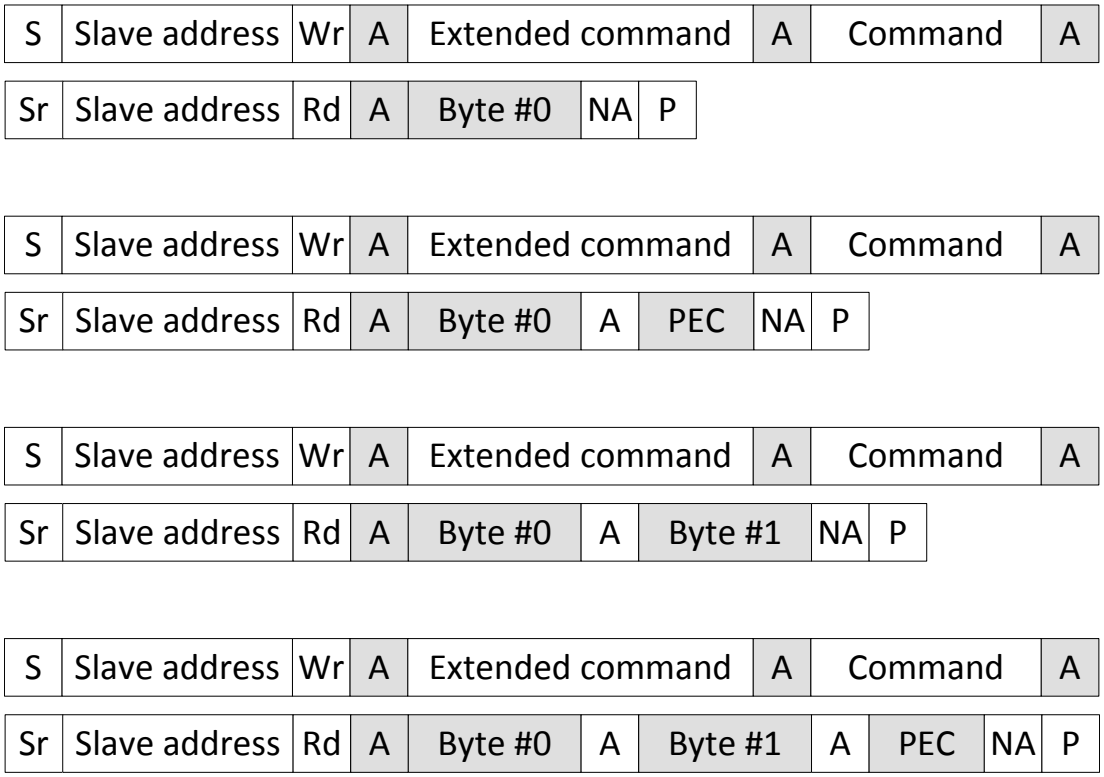


Figure 35-26. Extended Command Read Byte and Read Word Messages With and Without PEC

35.4.2.12 Group Command

The Group Command (Figure 35-27) protocol is used to send commands to more than one device within the same message. When devices on the bus detect the stop condition at the conclusion of the Group Command message, the received commands are executed concurrently. To initiate a Group Command, the GRP_CMD bit within the PMBMC register must be set when programming the slave address for the first device in the message. The rest of the message is processed as a write byte/word message. At the conclusion of the first part of the Group Command message, the firmware programs the next device address in the PMBMC register. The PMBus module sends a repeated start on the bus and begins the next part of the message. When programming the last device address of the Group Command message, the firmware must disable the GRP_CMD bit when programming the PMBMC register.

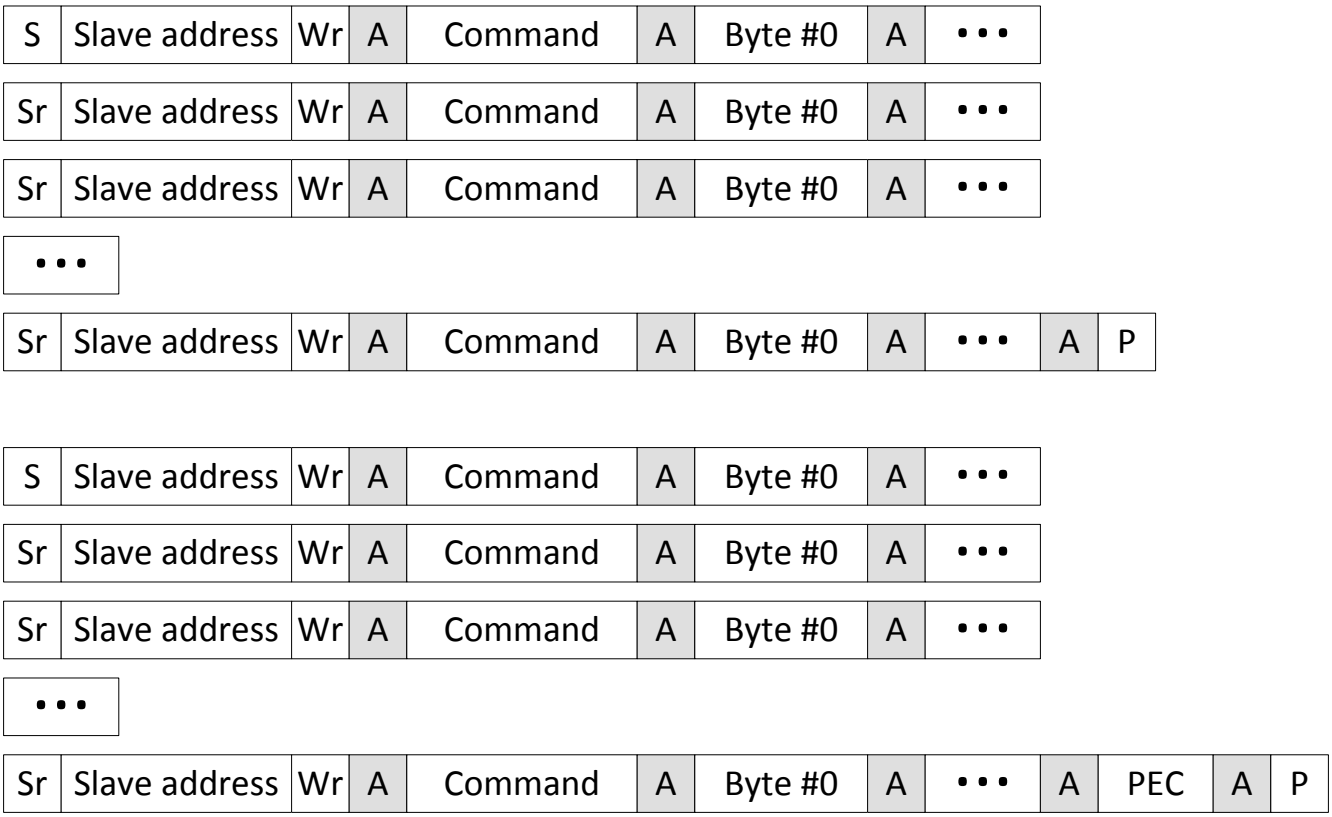


Figure 35-27. Group Command Message With and Without PEC

35.5 PMBus Registers

This section describes the Power-Management Bus module Registers.

35.5.1 PMBUS Base Address Table (C28)

Table 35-1. PMBUS Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
PmbusaRegs	PMBUS_REGS	PMBUSA_BASE	0x0000_6400	YES	YES	YES	YES	YES

35.5.2 PMBUS_REGS Registers

Table 35-2 lists the memory-mapped registers for the PMBUS_REGS registers. All register offset addresses not listed in Table 35-2 should be considered as reserved locations and the register contents should not be modified.

Table 35-2. PMBUS_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	PMBMC	PMBUS Master Mode Control Register	EALLOW	Go
2h	PMBTXBUF	PMBUS Transmit Buffer		Go
4h	PMBRXBUF	PMBUS Receive buffer		Go
6h	PMBACK	PMBUS Acknowledge Register		Go
8h	PMBSTS	PMBUS Status Register		Go
Ah	PMBINTM	PMBUS Interrupt Mask Register	EALLOW	Go
Ch	PMBSC	PMBUS Slave Mode Configuration Register	EALLOW	Go
Eh	PMBHSA	PMBUS Hold Slave Address Register		Go
10h	PMBCTRL	PMBUS Control Register	EALLOW	Go
12h	PMBTIMCTL	PMBUS Timing Control Register	EALLOW	Go
14h	PMBTIMCLK	PMBUS Clock Timing Register	EALLOW	Go
16h	PMBTIMSTSETUP	PMBUS Start Setup Time Register	EALLOW	Go
18h	PMBTIMBIDLE	PMBUS Bus Idle Time Register	EALLOW	Go
1Ah	PMBTIMLOWTIMOUT	PMBUS Clock Low Timeout Value Register	EALLOW	Go
1Ch	PMBTIMHIGHTIMOUT	PMBUS Clock High Timeout Value Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 35-3 shows the codes that are used for access types in this section.

Table 35-3. PMBUS_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
RC	R C	Read to Clear
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

35.5.2.1 PMBMC Register (Offset = 0h) [Reset = 0h]

PMBMC is shown in Figure 35-28 and described in Table 35-4.

Return to the Summary Table.

PMBUS Master Mode Control Register

Figure 35-28. PMBMC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED			PRC_CALL	GRP_CMD	PEC_ENA	EXT_CMD	CMD_ENA
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
BYTE_COUNT							
R/W-0h							
7	6	5	4	3	2	1	0
SLAVE_ADDR							RW
R/W-0h							R/W-0h

Table 35-4. PMBMC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-21	RESERVED	R	0h	Reserved
20	PRC_CALL	R/W	0h	0 = Default state for all messages besides Process Call message 1 = Enables transmission of Process Call message Reset type: SYSRSn
19	GRP_CMD	R/W	0h	0 = Default state for all messages besides Group Command message 1 = Enables transmission of Group Command message Reset type: SYSRSn
18	PEC_ENA	R/W	0h	0 = Disables PEC processing 1 = Enables PEC byte transmission/reception Reset type: SYSRSn
17	EXT_CMD	R/W	0h	0 = Use 1 byte for Command Code 1 = Use 2 bytes for Command Code Reset type: SYSRSn
16	CMD_ENA	R/W	0h	0 = Disables use of command code on Master initiated messages (1 = Enables use of command code on Master initiated messages Reset type: SYSRSn
15-8	BYTE_COUNT	R/W	0h	Indicates number of data bytes transmitted in current message. Byte count does not include any device addresses, command words or block lengths in block messages. In block messages, the PMBus Interface automatically inserts the block length into the message based on the byte count setting. The firmware only needs to load the address, command words and data to be transmitted. PMBus Interface supports byte writes up to 255 bytes. Reset type: SYSRSn
7-1	SLAVE_ADDR	R/W	0h	Specifies the address of the slave to which the current message is directed towards. Reset type: SYSRSn
0	RW	R/W	0h	0 = Message is a write transaction (data from Master to Slave) 1 = Message is a read transaction (data from Slave to Master) Reset type: SYSRSn

35.5.2.2 PMBTXBUF Register (Offset = 2h) [Reset = 0h]

PMBTXBUF is shown in Figure 35-29 and described in Table 35-5.

Return to the Summary Table.

PMBUS Transmit Buffer

Figure 35-29. PMBTXBUF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXDATA																															
R/W-0h																															

Table 35-5. PMBTXBUF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXDATA	R/W	0h	Bits 31-24: BYTE3 - Last data byte transmitted from Transmit Data Buffer Bits 23-16: BYTE2 - Third data byte transmitted from Transmit Data Buffer Bits 15-8: BYTE1 - Second data byte transmitted from Transmit Data Buffer Bits 7-0: BYTE0 - First data byte transmitted from Transmit Data Buffer Reset type: SYSRSn

35.5.2.3 PMBRXBUF Register (Offset = 4h) [Reset = 0h]

PMBRXBUF is shown in Figure 35-30 and described in Table 35-6.

Return to the Summary Table.

PMBUS Receive buffer

Figure 35-30. PMBRXBUF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXDATA																															
R-0h																															

Table 35-6. PMBRXBUF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXDATA	R	0h	Bits 31-24: BYTE3 - Last data byte received in Receive Data Buffer Bits 23-16: BYTE2 - Third data byte received in Receive Data Buffer Bits 15-8: BYTE1 - Second data byte received in Receive Data Buffer Bits 7-0: BYTE0 - First data byte received in Receive Data Buffer Reset type: SYSRSn

35.5.2.4 PMBACK Register (Offset = 6h) [Reset = 0h]

PMBACK is shown in Figure 35-31 and described in Table 35-7.

Return to the Summary Table.

PMBUS Acknowledge Register

Figure 35-31. PMBACK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ACK
R-0h															R/W-0h

Table 35-7. PMBACK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	ACK	R/W	0h	0 = NACK received data 1 = Acknowledge received data, bit clears upon issue of ACK on PMBus Reset type: SYSRSn

35.5.2.5 PMBSTS Register (Offset = 8h) [Reset = 00340000h]

PMBSTS is shown in Figure 35-32 and described in Table 35-8.

Return to the Summary Table.

PMBUS Status Register

Figure 35-32. PMBSTS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED		SCL_RAW	SDA_RAW	CONTROL_RAW	ALERT_RAW	CONTROL_EDGE	ALERT_EDGE
R-0h		R-1h	R-1h	R-0h	R-1h	RC-0h	RC-0h
15	14	13	12	11	10	9	8
MASTER	LOST_ARB	BUS_FREE	UNIT_BUSY	RPT_START	SLAVE_ADDR_READY	CLK_HIGH_DETECTED	CLK_LOW_TIMEOUT
RC-0h	RC-0h	RC-0h	RC-0h	RC-0h	RC-0h	RC-0h	RC-0h
7	6	5	4	3	2	1	0
PEC_VALID	NACK	EOM	DATA_REQUEST	DATA_READY	RD_BYTE_COUNT		
RC-0h	RC-0h	RC-0h	RC-0h	RC-0h	RC-0h		

Table 35-8. PMBSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reserved
21	SCL_RAW	R	1h	0 = PMBus clock pin observed at logic level low 1 = PMBus clock pin observed at logic level high Reset type: SYSRSn
20	SDA_RAW	R	1h	0 = PMBus data pin observed at logic level low 1 = PMBus data pin observed at logic level high Reset type: SYSRSn
19	CONTROL_RAW	R	0h	0 = Control pin observed at logic level low 1 = Control pin observed at logic level high Reset type: SYSRSn
18	ALERT_RAW	R	1h	0 = Alert pin observed at logic level low 1 = Alert pin observed at logic level high Reset type: SYSRSn
17	CONTROL_EDGE	RC	0h	0 = Control pin has not transitioned 1 = Control pin has been asserted by another device on PMBus Reset type: SYSRSn
16	ALERT_EDGE	RC	0h	0 = Alert pin has not transitioned 1 = Alert pin has been asserted by another device on PMBus Reset type: SYSRSn
15	MASTER	RC	0h	0 = PMBus Interface in Slave Mode or Idle Mode 1 = PMBus Interface in Master Mode Reset type: SYSRSn
14	LOST_ARB	RC	0h	0 = Master has attained control of PMBus 1 = Master has lost arbitration and control of PMBus Reset type: SYSRSn
13	BUS_FREE	RC	0h	0 = PMBus processing current message 1 = PMBus available for new message Reset type: SYSRSn

Table 35-8. PMBSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	UNIT_BUSY	RC	0h	0 = PMBus Interface is idle, ready to transmit/receive message 1 = PMBus Interface is busy, processing current message Reset type: SYSRSn
11	RPT_START	RC	0h	0 = No Repeated Start received by interface 1 = Repeated Start condition received by interface Reset type: SYSRSn
10	SLAVE_ADDR_READY	RC	0h	0 = Indicates no slave address is available for reading 1 = Slave address ready to be read from Receive Data Register (Bits 6:0) Reset type: SYSRSn
9	CLK_HIGH_DETECTED	RC	0h	0 = No Clock High condition detected 1 = Clock High exceeded 50us during message Reset type: SYSRSn
8	CLK_LOW_TIMEOUT	RC	0h	0 = No clock low timeout detected 1 = Clock low timeout detected, clock held low for greater than 35ms Reset type: SYSRSn
7	PEC_VALID	RC	0h	0 = Received PEC not valid (if EOM is asserted) 1 = Received PEC is valid Note: PEC_VALID status is don't care during the message. This will have a valid value only after EOM. Reset type: SYSRSn
6	NACK	RC	0h	0 = Data transmitted has been accepted by receiver 1 = Receiver has not accepted transmitted data Reset type: SYSRSn
5	EOM	RC	0h	0 = Message still in progress or PMBus in idle state. 1 = End of current message detected Reset type: SYSRSn
4	DATA_REQUEST	RC	0h	0 = No data needed by PMBus Interface 1 = PMBus Interface request additional data. PMBus clock stretching enabled to stall bus Reset type: SYSRSn
3	DATA_READY	RC	0h	0 = No data available for reading by processor 1 = PMBus Interface read buffer full, firmware required to read data prior to further bus activity. PMBus clock stretching enabled to stall bus until data is read by firmware. Reset type: SYSRSn
2-0	RD_BYTE_COUNT	RC	0h	0 = No received data 1 = 1 byte received. Data located in Receive Data Register, Bits 7-0 2 = 2 bytes received. Data located in Receive Data Register, Bits 15-0 3 = 3 bytes received. Data located in Receive Data Register, Bits 23-0 4 = 4 bytes received. Data located in Receive Data Register, Bits 31-0 Reset type: SYSRSn

35.5.2.6 PMBINTM Register (Offset = Ah) [Reset = 3FFh]

PMBINTM is shown in Figure 35-33 and described in Table 35-9.

Return to the Summary Table.

PMBUS Interrupt Mask Register

Figure 35-33. PMBINTM Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						CLK_HIGH_DETECT	LOST_ARB
R-0h						R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
CONTROL	ALERT	EOM	SLAVE_ADDR_READY	DATA_REQUEST	DATA_READY	BUS_LOW_TIMEOUT	BUS_FREE
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 35-9. PMBINTM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	CLK_HIGH_DETECT	R/W	1h	0 = Generates interrupt if clock high exceeds 50us during message 1 = Disables interrupt generation for Clock High detection Reset type: SYSRSn
8	LOST_ARB	R/W	1h	0 = Generates interrupt upon assertion of Lost Arbitration flag 1 = Disables interrupt generation upon assertion of Lost Arbitration flag Reset type: SYSRSn
7	CONTROL	R/W	1h	0 = Generates interrupt upon assertion of Control flag 1 = Disables interrupt generation upon assertion of Control flag Reset type: SYSRSn
6	ALERT	R/W	1h	0 = Generates interrupt upon assertion of Alert flag 1 = Disables interrupt generation upon assertion of Alert flag Reset type: SYSRSn
5	EOM	R/W	1h	0 = Generates interrupt upon assertion of End of Message flag 1 = Disables interrupt generation upon assertion of End of Message flag Reset type: SYSRSn
4	SLAVE_ADDR_READY	R/W	1h	0 = Generates interrupt upon assertion of Slave Address Ready flag 1 = Disables interrupt generation upon assertion of Slave Address Ready flag Reset type: SYSRSn
3	DATA_REQUEST	R/W	1h	0 = Generates interrupt upon assertion of Data Request flag 1 = Disables interrupt generation upon assertion of Data Request flag Reset type: SYSRSn
2	DATA_READY	R/W	1h	0 = Generates interrupt upon assertion of Data Ready flag 1 = Disables interrupt generation upon assertion of Data Ready flag Reset type: SYSRSn

Table 35-9. PMBINTM Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	BUS_LOW_TIMEOUT	R/W	1h	0 = Generates interrupt upon assertion of Clock Low Timeout flag 1 = Disables interrupt generation upon assertion of Clock Low Timeout flag Reset type: SYSRSn
0	BUS_FREE	R/W	1h	0 = Generates interrupt upon assertion of Bus Free flag 1 = Disables interrupt generation upon assertion of Bus Free flag Reset type: SYSRSn

35.5.2.7 PMBSC Register (Offset = Ch) [Reset = 00607F7Ch]

PMBSC is shown in Figure 35-34 and described in Table 35-10.

Return to the Summary Table.

PMBUS Slave Mode Configuration Register

Figure 35-34. PMBSC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED	RX_BYTE_ACK_CNT		MAN_CMD	TX_PEC	TX_COUNT		
R-0h	R/W-3h		R/W-0h	R/W-0h	R/W-0h		
15	14	13	12	11	10	9	8
PEC_ENA	SLAVE_MASK						
R/W-0h	R/W-7Fh						
7	6	5	4	3	2	1	0
MAN_SLAVE_ACK	SLAVE_ADDR						
R/W-0h	R/W-7Ch						

Table 35-10. PMBSC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22-21	RX_BYTE_ACK_CNT	R/W	3h	Configures number of data bytes to automatically acknowledge when receiving data in slave mode. 00 = 1 byte received by slave. Firmware is required to manually acknowledge every received byte. 01 = 2 bytes received by slave. Hardware automatically acknowledges the first received byte. Firmware is required to manually acknowledge after the second received byte. 10 = 3 bytes received by slave. Hardware automatically acknowledges the first 2 received bytes. Firmware is required to manually acknowledge after the third received byte. 11 = 4 bytes received by slave. Hardware automatically acknowledges the first 3 received bytes. Firmware is required to manually acknowledge after the fourth received byte Reset type: SYSRSn
20	MAN_CMD	R/W	0h	0 = Slave automatically acknowledges received command code 1 = Data Request flag generated after receipt of command code, firmware required to issue ACK to continue message Reset type: SYSRSn
19	TX_PEC	R/W	0h	Asserted when the slave needs to send a PEC byte at end of message. PMBus Interface will transmit the calculated PEC byte after transmitting the number of data bytes indicated by TX Byte Cnt(Bits 18:16). Reset type: SYSRSn

Table 35-10. PMBSC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-16	TX_COUNT	R/W	0h	0 = No bytes valid 1 = One byte valid, Byte #0 (Bits 7:0 of Transmit Data Register) 2 = Two bytes valid, Bytes #0 and #1 (Bits 15:0 of Transmit Data Register) 3 = Three bytes valid, Bytes #0-2 (Bits 23:0 of Transmit Data Register) 4 = Four bytes valid, Bytes #0-3 (Bits 31:0 of Transmit Data Register) Reset type: SYSRSn
15	PEC_ENA	R/W	0h	0 = PEC processing disabled 1 = PEC processing enabled Reset type: SYSRSn
14-8	SLAVE_MASK	R/W	7Fh	Used in address detection, the slave mask enables acknowledgement of multiple device addresses by the slave. Writing a '0' to a bit within the slave mask enables the corresponding bit in the slave address to be either '1' or '0' and still allow for a match. Writing a '0' to all bits in the mask enables the PMBus Interface to acknowledge any device address. Upon power-up, the slave mask defaults to 7Fh, indicating the slave will only acknowledge the address programmed into the Slave Address (Bits 6-0). Reset type: SYSRSn
7	MAN_SLAVE_ACK	R/W	0h	0 = Slave automatically acknowledges device address specified in SLAVE_ADDR, Bits 6:0 1 = Enables the Manual Slave Address Acknowledgement Mode. Firmware is required to read received address and acknowledge on every message Note: When bit 31 (I2C_mode) of PMBCTRL register is set it is recommended to use manual acknowledging of slave address only (MAN_SLAVE_ACK =1). Reset type: SYSRSn
6-0	SLAVE_ADDR	R/W	7Ch	Configures the current device address of the slave. Used in automatic slave address acknowledge mode (default mode). The PMBus Interface will compare the received device address with the value stored in the Slave Address bits and the mask configured in the Slave Mask bits. If matching, the slave will acknowledge the device address. Reset type: SYSRSn

35.5.2.8 PMBHSA Register (Offset = Eh) [Reset = 0h]

PMBHSA is shown in Figure 35-35 and described in Table 35-11.

Return to the Summary Table.

PMBUS Hold Slave Address Register

Figure 35-35. PMBHSA Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
SLAVE_ADDR							SLAVE_RW
R-0h							R-0h

Table 35-11. PMBHSA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-1	SLAVE_ADDR	R	0h	Stored device address acknowledged by the slave Reset type: SYSRSn
0	SLAVE_RW	R	0h	Stored R/W bit from address acknowledged by the slave 0 = Write Access 1 = Read Access Reset type: SYSRSn

35.5.2.9 PMBCTRL Register (Offset = 10h) [Reset = 00200000h]

PMBCTRL is shown in Figure 35-36 and described in Table 35-12.

Return to the Summary Table.

PMBUS Control Register

Figure 35-36. PMBCTRL Register

31	30	29	28	27	26	25	24
I2CMODE	RESERVED			CLKDIV			
R/W-0h	R-0h			R/W-0h			
23	22	21	20	19	18	17	16
CLKDIV	MASTER_EN	SLAVE_EN	CLK_LO_DIS	IBIAS_B_EN	IBIAS_A_EN	SCL_DIR	SCL_VALUE
R/W-0h	R/W-0h	R/W-1h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
SCL_MODE	SDA_DIR	SDA_VALUE	SDA_MODE	CNTL_DIR	CNTL_VALUE	CNTL_MODE	ALERT_DIR
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
ALERT_VALUE	ALERT_MODE	CNTL_INT_EDGE	RESERVED	FAST_MODE	BUS_LO_INT_EDGE	ALERT_EN	RESET
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 35-12. PMBCTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	I2CMODE	R/W	0h	0 = PMBUS mode 1 = I2C mode Reset type: SYSRSn
30-28	RESERVED	R	0h	Reserved
27-23	CLKDIV	R/W	0h	The clock to the PMBUS transmit/receive FSMs (FSM_CLK) is divided version of the SYSCLK clock. Frequency(FSM_CLK) = Frequency(SYSCLK)/(CLKDIV+1) Note: FSM_CLK should be less than (or) equal to 10MHz. Reset type: SYSRSn
22	MASTER_EN	R/W	0h	0 = Disables PMBus Master capability 1 = Enables PMBus Master capability Reset type: SYSRSn
21	SLAVE_EN	R/W	1h	0 = Disables PMBus Slave capability 1 = Enables PMBus Slave capability Reset type: SYSRSn
20	CLK_LO_DIS	R/W	0h	0 = Clock Low Timeout Enabled 1 = Clock Low Timeout Disabled Reset type: SYSRSn
19	IBIAS_B_EN	R/W	0h	0 = Disables Current Source for PMBUS address detection thru ADC 1 = Enables Current Source for PMBUS address detection thru ADC Reset type: SYSRSn
18	IBIAS_A_EN	R/W	0h	0 = Disables Current Source for PMBUS address detection thru ADC 1 = Enables Current Source for PMBUS address detection thru ADC Reset type: SYSRSn
17	SCL_DIR	R/W	0h	0 = PMBus clock pin configured as output 1 = PMBus clock pin configured as input Reset type: SYSRSn
16	SCL_VALUE	R/W	0h	0 = PMBus clock pin driven low in GPIO Mode 1 = PMBus clock pin driven high in GPIO Mode Reset type: SYSRSn

Table 35-12. PMBCTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15	SCL_MODE	R/W	0h	0 = PMBus clock pin configured in functional mode 1 = PMBus clock pin configured as GPIO Reset type: SYSRSn
14	SDA_DIR	R/W	0h	0 = PMBus data pin configured as output 1 = PMBus data pin configured as input Reset type: SYSRSn
13	SDA_VALUE	R/W	0h	0 = PMBus data pin driven low in GPIO Mode 1 = PMBus data pin driven high in GPIO Mode Reset type: SYSRSn
12	SDA_MODE	R/W	0h	0 = PMBus data pin driven low in GPIO Mode 1 = PMBus data pin driven high in GPIO Mode Reset type: SYSRSn
11	CNTL_DIR	R/W	0h	0 = Control pin configured as output 1 = Control pin configured as input Reset type: SYSRSn
10	CNTL_VALUE	R/W	0h	0 = Control pin driven low in GPIO Mode 1 = Control pin driven high in GPIO Mode Reset type: SYSRSn
9	CNTL_MODE	R/W	0h	0 = Control pin configured in functional mode (Default) 1 = Control pin configured as GPIO Reset type: SYSRSn
8	ALERT_DIR	R/W	0h	0 = Alert pin configured as output 1 = Alert pin configured as input Reset type: SYSRSn
7	ALERT_VALUE	R/W	0h	0 = Alert pin driven low in GPIO Mode 1 = Alert pin driven high in GPIO Mode Reset type: SYSRSn
6	ALERT_MODE	R/W	0h	0 = Alert pin configured in functional mode 1 = Aler3 pin configured as GPIO Reset type: SYSRSn
5	CNTL_INT_EDGE	R/W	0h	0 = Interrupt generated on falling edge of Control 1 = Interrupt generated on rising edge of Control Reset type: SYSRSn
4	RESERVED	R/W	0h	Reserved
3	FAST_MODE	R/W	0h	0 = Standard 100 KHz mode enabled 1 = Fast Mode enabled (400KHz operation on PMBus) Reset type: SYSRSn
2	BUS_LO_INT_EDGE	R/W	0h	0 = Interrupt generated on rising edge of clock low timeout 1 = Interrupt generated on falling edge of clock low timeout Reset type: SYSRSn
1	ALERT_EN	R/W	0h	0 = PMBus Alert is not driven by slave, pulled up high on PMBus 1 = PMBus Alert driven low by slave Reset type: SYSRSn
0	RESET	R/W	0h	0 = No reset of internal state machines (Default) 1 = Control state machines are reset to initial states Note: Status register PMBSTS should be explicitly cleared by reading the register after softrest as this will not be cleared by Software Reset. Reset type: SYSRSn

35.5.2.10 PMBTIMCTL Register (Offset = 12h) [Reset = 0h]

PMBTIMCTL is shown in Figure 35-37 and described in Table 35-13.

Return to the Summary Table.

PMBUS Timing Control Register

Figure 35-37. PMBTIMCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							TIM_OVERRIDE
R-0h							R/W-0h

Table 35-13. PMBTIMCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	TIM_OVERRIDE	R/W	0h	0 PMBUS FSMs uses the default settings of the timing parameters. 1 PMBUS FSMs would use the settings in following registers: * PMBTIMCLK * PMBTIMSTSETUP * PMBTIMBIDLE * PMBTIMLOWTIMOUT * PMBTIMHIGHTIMOUT Reset type: SYSRSn

35.5.2.11 PMBTIMCLK Register (Offset = 14h) [Reset = 0060002Fh]

PMBTIMCLK is shown in Figure 35-38 and described in Table 35-14.

Return to the Summary Table.

PMBUS Clock Timing Register

Figure 35-38. PMBTIMCLK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								CLK_FREQ							
R-0h								R/W-60h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								CLK_HIGH_LIMIT							
R-0h								R/W-2Fh							

Table 35-14. PMBTIMCLK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	CLK_FREQ	R/W	60h	Defines the number of PMBUS FSM input clock in the PMBUS master clock period. Number of FSM clocks in the one clock period = (CLK_FREQ+4) Reset type: SYSRSn
15-8	RESERVED	R	0h	Reserved
7-0	CLK_HIGH_LIMIT	R/W	2Fh	Defines the number of PMBUS FSM input clock in the PMBUS master clock high pulse. Number of FSM clocks in the one clock high pulse = (CLK_HIGH_LIMIT+3) Reset type: SYSRSn

35.5.2.12 PMBTIMSTSETUP Register (Offset = 16h) [Reset = 2Fh]

PMBTIMSTSETUP is shown in Figure 35-39 and described in Table 35-15.

Return to the Summary Table.

PMBUS Start Setup Time Register

Figure 35-39. PMBTIMSTSETUP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								TSU_STA							
R-0h																								R/W-2Fh							

Table 35-15. PMBTIMSTSETUP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	TSU_STA	R/W	2Fh	Determines the Setup time between last rise edge of the PMBUS master clock and the next start edge, TSU_STA value defines the setup time in terms of PMBUS FSM clock cycles. Reset type: SYSRSn

35.5.2.13 PMBTIMBIDLE Register (Offset = 18h) [Reset = 1F3h]

PMBTIMBIDLE is shown in Figure 35-40 and described in Table 35-16.

Return to the Summary Table.

PMBUS Bus Idle Time Register

Figure 35-40. PMBTIMBIDLE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						BUSIDLE									
R-0h																						R/W-1F3h									

Table 35-16. PMBTIMBIDLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9-0	BUSIDLE	R/W	1F3h	Determines the duration for which PMBUS clock and Data are 1 , to conclude that the bus is IDLE. BUSIDLE value is in terms of number of PMBUS FSM clock cycles. Reset type: SYSRSn

35.5.2.14 PMBTIMLOWTIMOUT Register (Offset = 1Ah) [Reset = 0005572Fh]

PMBTIMLOWTIMOUT is shown in Figure 35-41 and described in Table 35-17.

Return to the Summary Table.

PMBUS Clock Low Timeout Value Register

Figure 35-41. PMBTIMLOWTIMOUT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												CLKLOWTIMOUT																			
R-0h												R/W-0005572Fh																			

Table 35-17. PMBTIMLOWTIMOUT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-0	CLKLOWTIMOUT	R/W	0005572Fh	Determines the duration for which PMBUS clock if low , will result in a clock low timeout condition. CLKLOWTIMOUT value is in terms of number of PMBUS FSM clock cycles. Reset type: SYSRSn

35.5.2.15 PMBTIMHIGHTIMOUT Register (Offset = 1Ch) [Reset = 1F3h]

PMBTIMHIGHTIMOUT is shown in Figure 35-42 and described in Table 35-18.

Return to the Summary Table.

PMBUS Clock High Timeout Value Register

Figure 35-42. PMBTIMHIGHTIMOUT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						CLKHIGHTIMOUT									
R-0h						R/W-1F3h									

Table 35-18. PMBTIMHIGHTIMOUT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9-0	CLKHIGHTIMOUT	R/W	1F3h	Determines the duration for which PMBUS clock if high , will result in a clock high timeout condition. CLKHIGHTIMOUT value is in terms of number of PMBUS FSM clock cycles. Reset type: SYSRSn

35.5.3 PMBUS Registers to Driverlib Functions

Table 35-19. PMBUS Registers to Driverlib Functions

File	Driverlib Function
PMBCCR	
pmbus.h	PMBus_configController
pmbus.h	PMBus_setTargetAddress
PMBTXBUF	
pmbus.c	PMBus_putTargetData
pmbus.c	PMBus_putControllerData
PMBRXBUF	
pmbus.c	PMBus_getData
PMBACK	
pmbus.c	PMBus_ackAddress
pmbus.c	PMBus_ackCommand
pmbus.h	PMBus_ackTransaction
pmbus.h	PMBus_nackTransaction
PMBSTS	
pmbus.c	PMBus_getInterruptStatus
pmbus.h	PMBus_getStatus
PMBINTM	
pmbus.c	PMBus_initTargetMode
pmbus.c	PMBus_configTarget
pmbus.c	PMBus_initControllerMode
pmbus.c	PMBus_configModuleClock
pmbus.c	PMBus_configBusClock
pmbus.h	PMBus_enableInterrupt
pmbus.h	PMBus_disableInterrupt

Table 35-19. PMBUS Registers to Driverlib Functions (continued)

File	Driverlib Function
pmbus.h	PMBus_enableI2CMode
pmbus.h	PMBus_disableI2CMode
PMBTCR	
pmbus.c	PMBus_initTargetMode
pmbus.c	PMBus_configTarget
pmbus.c	PMBus_putTargetData
pmbus.c	PMBus_ackAddress
pmbus.c	PMBus_ackCommand
pmbus.h	PMBus_setOwnAddress
PMBHTA	
pmbus.c	PMBus_verifyPEC
pmbus.h	PMBus_getOwnAddress
pmbus.h	PMBus_getCurrentAccessType
PMBCTRL	
pmbus.c	PMBus_initTargetMode
pmbus.c	PMBus_initControllerMode
pmbus.c	PMBus_configModuleClock
pmbus.c	PMBus_configBusClock
pmbus.h	PMBus_disableModule
pmbus.h	PMBus_enableModule
pmbus.h	PMBus_enableI2CMode
pmbus.h	PMBus_disableI2CMode
pmbus.h	PMBus_assertAlertLine
pmbus.h	PMBus_deassertAlertLine
pmbus.h	PMBus_setCtrlIntEdge
pmbus.h	PMBus_setClkLowTimeoutIntEdge
PMBTIMCTL	
pmbus.c	PMBus_configBusClock
PMBTIMCLK	
pmbus.c	PMBus_configBusClock
PMBTIMSTSETUP	
pmbus.c	PMBus_configBusClock
PMBTIMBIDLE	
pmbus.c	PMBus_configBusClock
PMBTIMLOWTIMOUT	
pmbus.c	PMBus_configBusClock
PMBTIMHIGHTIMOUT	
pmbus.c	PMBus_configBusClock

Chapter 36

Serial Communications Interface (SCI)

This chapter describes the features and operation of the serial communication interface (SCI) module. SCI is a two-wire asynchronous serial port, commonly known as a UART. The SCI modules support digital communications between the CPU and other asynchronous peripherals that use the standard non-return-to-zero (NRZ) format. The SCI receiver and transmitter each have a 16-level deep FIFO for reducing servicing overhead, and each has its own separate enable and interrupt bits. Both can be operated independently for half-duplex communication, or simultaneously for full-duplex communication.

To specify data integrity, the SCI checks received data for break detection, parity, overrun, and framing errors. The bit rate is programmable to different speeds through a 16-bit baud-select register.

36.1 Introduction

The SCI interfaces are shown in Figure 36-1.

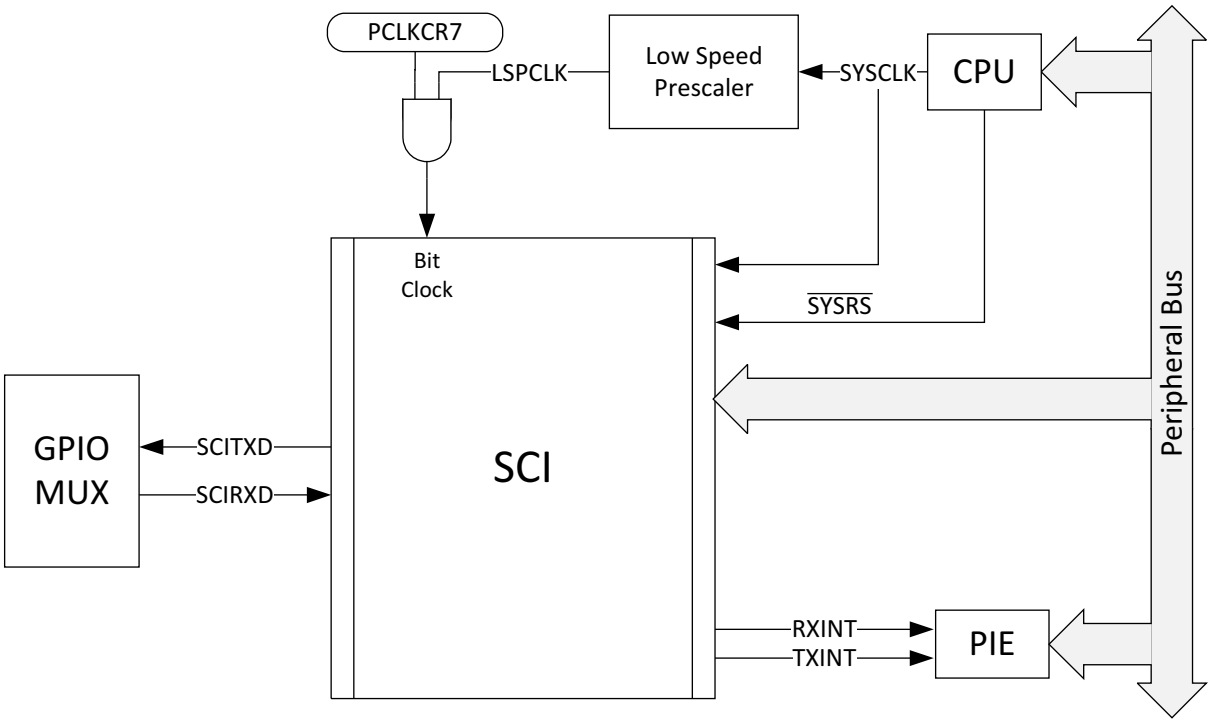


Figure 36-1. SCI CPU Interface

36.1.1 Features

Features of the SCI module include:

- Two external pins (both pins can be used as GPIO, if not used for SCI):
  - SCITXD: SCI transmit-output pin
  - SCIRXD: SCI receive-input pin
- Baud rate programmable to 64K different rates
- Data-word format
  - One start bit
  - Data-word length programmable from one to eight bits
  - Optional even/odd/no parity bit
  - One or two stop bits
  - An extra bit to distinguish addresses from data (address bit mode only)
- Four error-detection flags: parity, overrun, framing, and break detection
- Two wake-up multiprocessor modes: idle-line and address bit
- Half- or full-duplex operation
- Double-buffered receive and transmit functions
- Transmitter and receiver operations can be accomplished through interrupt-driven or polled algorithms with status flags
- Separate enable bits for transmitter and receiver interrupts (except BRKDT)
- NRZ (non-return-to-zero) format

Enhanced features include:

- Auto-baud-detect hardware logic
- 16-level transmit/receive FIFO

36.1.2 SCI Related Collateral

Foundational Materials

- C2000 Academy - SCI
- One Minute RS-485 Introduction (Video)
- RS-232, RS-422, RS-485: What Are the Differences? (Video)

Getting Started Materials

- [FAQ] My C2000 SCI is not Transmitting and/or Receiving data correctly, how do I fix this?

36.1.3 Block Diagram

Figure 36-2 shows the SCI module block diagram. The SCI port operation is configured and controlled by the registers listed in Section 36.15.

36.2 Architecture

The major elements used in full-duplex operation are shown in Figure 36-2 and include:

- A transmitter (TX) and its major registers (upper half of Figure 36-2)
  - SCITXBUF — transmitter data buffer register. Contains data (loaded by the CPU) to be transmitted
  - TXSHF register — transmitter shift register. Accepts data from register SCITXBUF and shifts data onto the SCITXD pin, one bit at a time
- A receiver (RX) and its major registers (lower half of Figure 36-2)
  - RXSHF register — receiver shift register. Shifts data in from SCIRXD pin, one bit at a time
  - SCIRXBUF — receiver data buffer register. Contains data to be read by the CPU. Data from a remote processor is loaded into register RXSHF and then into registers SCIRXBUF and SCIRXEMU
- A programmable baud generator
- Control and status registers

The SCI receiver and transmitter can operate either independently or simultaneously.

36.3 SCI Module Signal Summary

A summarized description of each SCI signal name is shown in Table 36-1.

Table 36-1. SCI Module Signal Summary

Signal Name	Description
External signals	
SCIRXD	SCI Asynchronous Serial Port receive data
SCITXD	SCI Asynchronous Serial Port transmit data
Control	
Baud clock	LSPCLK Prescaled clock
Interrupt signals	
TXINT	Transmit interrupt
RXINT	Receive Interrupt

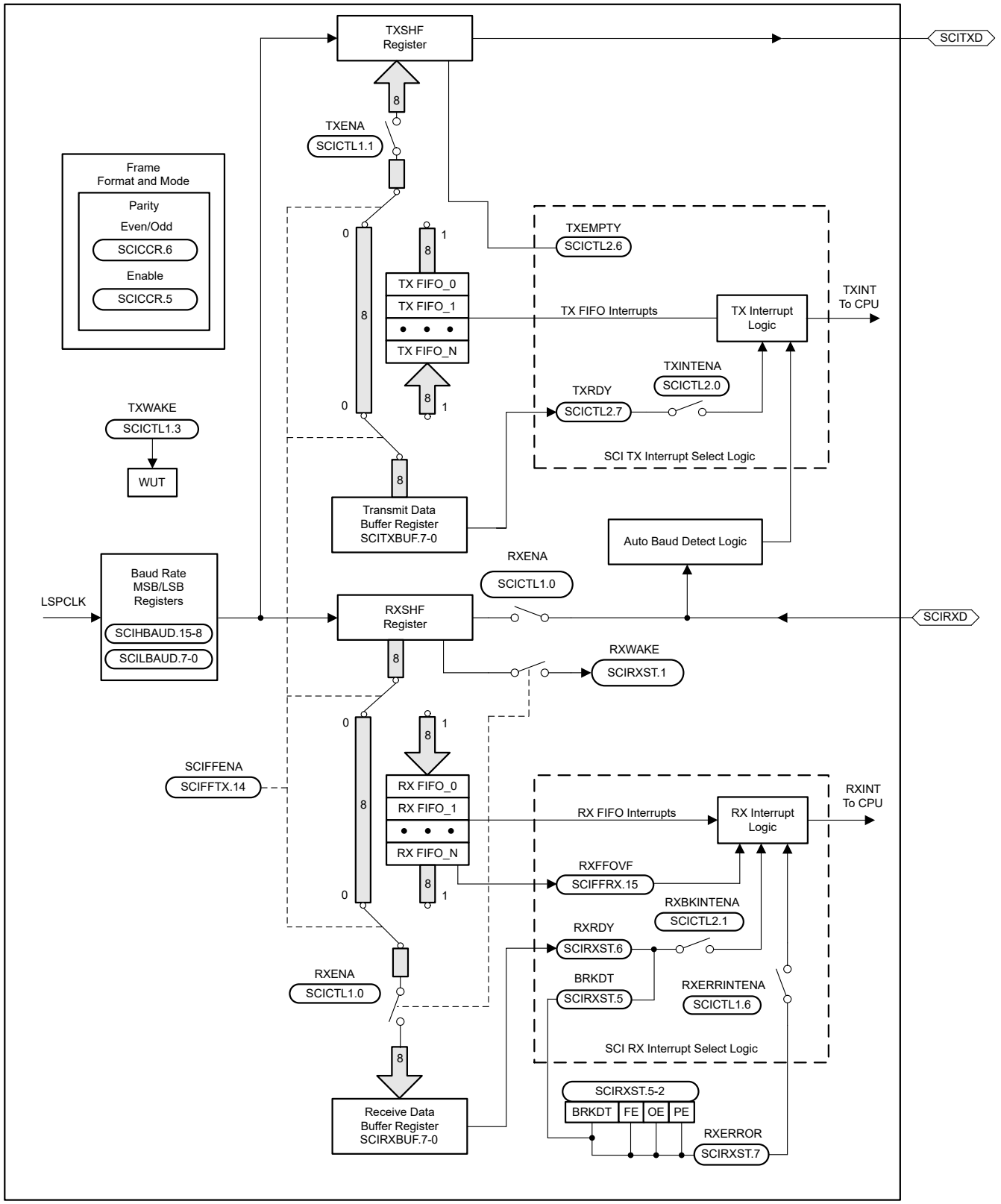


Figure 36-2. Serial Communications Interface (SCI) Module Block Diagram

36.4 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

Some IO functionality is defined by GPIO register settings independent of this peripheral. For input signals, the GPIO input qualification must be set to asynchronous mode by setting the appropriate GPxQSELn register bits to 11b. The internal pullups can be configured in the GPyPUD register.

See the *General-Purpose Input/Output (GPIO)* chapter for more details on GPIO mux and settings.

36.5 Multiprocessor and Asynchronous Communication Modes

The SCI has two multiprocessor protocols, the idle-line multiprocessor mode (see Section 36.8) and the address-bit multiprocessor mode (see Section 36.9). These protocols allow efficient data transfer between multiple processors.

The SCI offers the universal asynchronous receiver/transmitter (UART) communications mode for interfacing with many popular peripherals. The asynchronous mode (see Section 36.10) requires two lines to interface with many standard devices such as terminals and printers that use RS-232-C formats. Data transmission characteristics include:

- One start bit
- One to eight data bits
- An even/odd parity bit or no parity bit
- One or two stop bits

36.6 SCI Programmable Data Format

SCI data, both receive and transmit, is in NRZ (non-return-to-zero) format. The NRZ data format, shown in Figure 36-3, consists of:

- One start bit
- One to eight data bits
- An even/odd parity bit (optional)
- One or two stop bits
- An extra bit to distinguish addresses from data (address-bit mode only)

The basic unit of data is called a character and is one to eight bits in length. Each character of data is formatted with a start bit, one or two stop bits, and optional parity and address bits. A character of data with formatting information is called a frame and is shown in Figure 36-3.

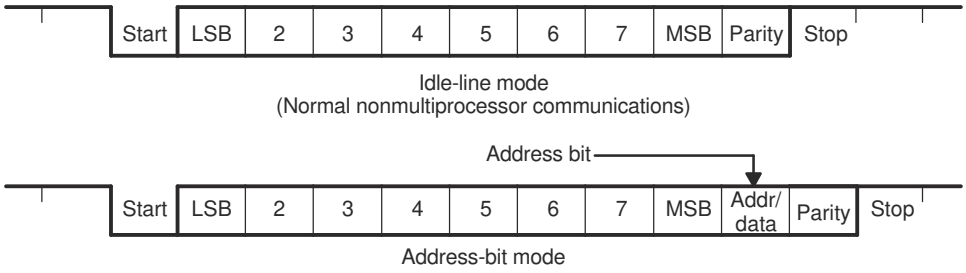


Figure 36-3. Typical SCI Data Frame Formats

To program the data format, use the SCICCR register. The bits used to program the data format are shown in Table 36-2.

Table 36-2. Programming the Data Format Using SCICCR

Bits	Bit Name	Designation	Functions
2-0	SCICHAR	SCICCR.2:0	Select the character (data) length (one to eight bits).
5	PARITYENA (ENABLE)	SCICCR.5	Enables the parity function if set to 1, or disables the parity function if cleared to 0.
6	PARITY (EVEN/ODD)	SCICCR.6	If parity is enabled, selects odd parity if cleared to 0; even parity if set to 1.
7	STOPBITS	SCICCR.7	Determines the number of stop bits transmitted—one stop bit if cleared to 0 or two stop bits if set to 1.

36.7 SCI Multiprocessor Communication

The multiprocessor communication format allows one processor to efficiently send blocks of data to other processors on the same serial link. On one serial line, there can be only one transfer at a time. In other words, there can be only one talker on a serial line at a time.

Address Byte

The first byte of a block of information that the talker sends contains an address byte that is read by all listeners. Only listeners with the correct address can be interrupted by the data bytes that follow the address byte. The listeners with an incorrect address remain uninterrupted until the next address byte.

Sleep Bit

All processors on the serial link set the SCI SLEEP bit (bit 2 of SCICTL1) to 1 so that the processor is interrupted only when the address byte is detected. When the processor reads a block address that corresponds to the CPU device address as set by your application software, your program must clear the SLEEP bit to enable the SCI to generate an interrupt on receipt of each data byte.

Although the receiver still operates when the SLEEP bit is 1, the receiver does not set RXRDY, RXINT, or any of the receiver error status bits to 1 unless the address byte is detected and the address bit in the received frame is a 1 (applicable to address-bit mode). The SCI does not alter the SLEEP bit; your software must alter the SLEEP bit.

36.7.1 Recognizing the Address Byte

A processor recognizes an address byte differently, depending on the multiprocessor mode used. For example:

- The idle-line mode (Section 36.8) leaves a quiet space before the address byte. This mode does not have an extra address/data bit and is more efficient than the address-bit mode for handling blocks that contain more than 10 bytes of data. The idle-line mode must be used for typical non-multiprocessor SCI communication.
- The address-bit mode (Section 36.9) adds an extra bit (that is, an address bit) into every byte to distinguish addresses from data. This mode is more efficient in handling many small blocks of data because, unlike the idle mode, this mode does not have to wait between blocks of data. However, at a high transmit speed, the program is not fast enough to avoid a 10-bit idle in the transmission stream.

36.7.2 Controlling the SCI TX and RX Features

The multiprocessor mode is software selectable using the ADDR/IDLE MODE bit (SCICCR, bit 3). Both modes use the TXWAKE flag bit (SCICTL1, bit 3), RXWAKE flag bit (SCIRXST, bit1), and the SLEEP flag bit (SCICTL1, bit 2) to control the SCI transmitter and receiver features of these modes.

36.7.3 Receipt Sequence

In both multiprocessor modes, the receive sequence is as follows:

1. At the receipt of an address block, the SCI port wakes up and requests an interrupt (bit number 1 RX/BK INT ENA-of SCICTL2 must be enabled to request an interrupt in non-FIFO mode of operation. In FIFO mode, RXFFINT serves this purpose and to enable this, RXFFINTEN in SCIFFRX register must be enabled with RXFFIL in the same register set to 1). It reads the first frame of the block, which contains the destination address.
2. A software routine is entered through the interrupt and checks the incoming address. This address byte is checked against its device address byte stored in memory.
3. If the check shows that the block is addressed to the device CPU, the CPU clears the SLEEP bit and reads the rest of the block. If not, the software routine exits with the SLEEP bit still set, and does not receive interrupts until the next block start.

36.8 Idle-Line Multiprocessor Mode

In the idle-line multiprocessor protocol (ADDR/IDLE MODE bit=0), blocks are separated by having a longer idle time between the blocks than between frames in the blocks. An idle time of ten or more high-level bits after a frame indicates the start of a new block. The time of a single bit is calculated directly from the baud value (bits per second). The idle-line multiprocessor communication format is shown in Figure 36-4 (ADDR/IDLE MODE bit is bit 3 of SCICCR).

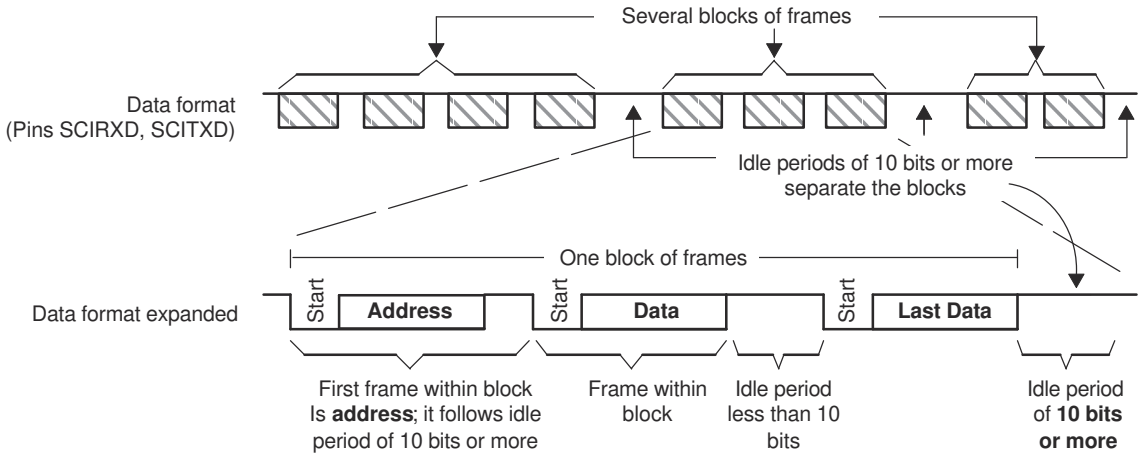


Figure 36-4. Idle-Line Multiprocessor Communication Format

36.8.1 Idle-Line Mode Steps

The steps followed by the idle-line mode:

1. SCI wakes up after receipt of the block-start signal.
2. The processor recognizes the next SCI interrupt.
3. The interrupt service routine compares the received address (sent by a remote transmitter) to its own.
4. If the CPU is being addressed, the service routine clears the SLEEP bit and receives the rest of the data block.
5. If the CPU is not being addressed, the SLEEP bit remains set. This lets the CPU continue to execute the main program without being interrupted by the SCI port until the next detection of a block start.

Note

In IDLE mode, if the SCI is taking greater than 10 bit periods to read all the RXDATA from the FIFO, it will miss the immediate block start to be detected.

The RXWAKE logic will assert only one time when it sees 10 bit periods of IDLE. It will not assert again if RXBUF is read (which clears the WAKE condition) even if the line continues to be idle after RXBUF read.

So, if the ISR is taking more than 10 bit periods of time to read all the RXDATA from the FIFO via RXBUF, the SCI may miss to detect next block start. This is applicable for both FIFO & Non-FIFO mode when the CPU takes greater than 10 bit clocks of SCI to read the data from RXBUF/FIFO.

To avoid this, either of the following is recommended:

- Set SCICTL1.SWRESET after reading all RX data at the end of the ISR.
- Read and check RXWAKE status bit before reading the RXBUF register. If RXWAKE is set, don't set SLEEP bit for RX at the end of the ISR.

36.8.2 Block Start Signal

There are two ways to send a block-start signal:

- **Method 1:** Deliberately leave an idle time of ten bits or more by delaying the time between the transmission of the last frame of data in the previous block and the transmission of the address frame of the new block.
- **Method 2:** The SCI port first sets the TXWAKE bit (SCICTL1, bit 3) to 1 before writing to the SCITXBUF register. This sends an idle time of exactly 11 bits. In this method, the serial communications line is not idle any longer than necessary. (A don't care byte has to be written to SCITXBUF after setting TXWAKE, and before sending the address, so as to transmit the idle time.)

36.8.3 Wake-Up Temporary (WUT) Flag

Associated with the TXWAKE bit is the wake-up temporary (WUT) flag. WUT is an internal flag, double-buffered with TXWAKE. When TXSHF is loaded from SCITXBUF, WUT is loaded from TXWAKE, and the TXWAKE bit is cleared to 0. This arrangement is shown in Figure 36-5.

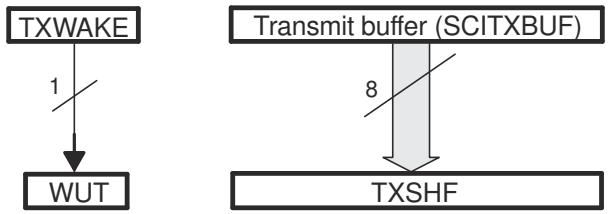


Figure 36-5. Double-Buffered WUT and TXSHF

36.8.3.1 Sending a Block Start Signal

To send out a block-start signal of exactly one frame time during a sequence of block transmissions:

1. Write a 1 to the TXWAKE bit.
2. Write a data word (content not important: a don’t care) to the SCITXBUF register (transmit data buffer) to send a block-start signal. (The first data word written is suppressed while the block-start signal is sent out and ignored after that.) When the TXSHF (transmit shift register) is free again, SCITXBUF contents are shifted to TXSHF, the TXWAKE value is shifted to WUT, and then TXWAKE is cleared.

   Because TXWAKE was set to a 1, the start, data, and parity bits are replaced by an idle period of 11 bits transmitted following the last stop bit of the previous frame.
3. Write a new address value to SCITXBUF.

   A don’t-care data word must first be written to register SCITXBUF so that the TXWAKE bit value can be shifted to WUT. After the don’t-care data word is shifted to the TXSHF register, the SCITXBUF (and TXWAKE, if necessary) can be written to again because TXSHF and WUT are both double-buffered.

36.8.4 Receiver Operation

The receiver operates regardless of the SLEEP bit. However, the receiver neither sets RXRDY nor the error status bits, nor does it request a receive interrupt until an address frame is detected.

36.9 Address-Bit Multiprocessor Mode

In the address-bit protocol (ADDR/IDLE MODE bit=1), frames have an extra bit called an address bit that immediately follows the last data bit. The address bit is set to 1 in the first frame of the block and to 0 in all other frames. The idle period timing is irrelevant (see Figure 36-6).

36.9.1 Sending an Address

The TXWAKE bit value is placed in the address bit. During transmission, when the SCITXBUF register and TXWAKE are loaded into the TXSHF register and WUT respectively, TXWAKE is reset to 0 and WUT becomes the value of the address bit of the current frame. Thus, to send an address:

1. Set the TXWAKE bit to 1 and write the appropriate address value to the SCITXBUF register.

   When this address value is transferred to the TXSHF register and shifted out, the address bit is sent as a 1. This flags the other processors on the serial link to read the address.
2. Write to SCITXBUF and TXWAKE after TXSHF and WUT are loaded. (Can be written to immediately since both TXSHF and WUT are both double-buffered.
3. Leave the TXWAKE bit set to 0 to transmit non-address frames in the block.

Note

As a general rule, the address-bit format is typically used for data frames of 11 bytes or less. This format adds one bit value (1 for an address frame, 0 for a data frame) to all data bytes transmitted. The idle-line format is typically used for data frames of 12 bytes or more.

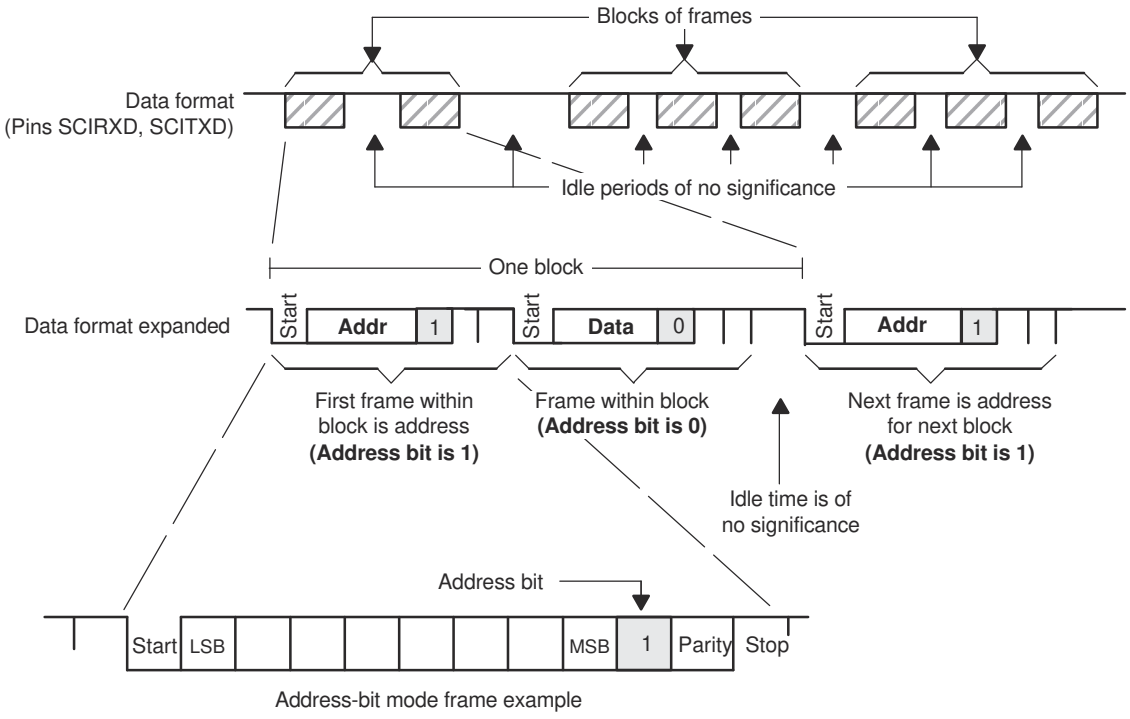


Figure 36-6. Address-Bit Multiprocessor Communication Format

36.10 SCI Communication Format

The SCI asynchronous communication format uses either single line (one way) or two line (two way) communications. In this mode, the frame consists of a start bit, one to eight data bits, an optional even/odd parity bit, and one or two stop bits (shown in Figure 36-7). There are eight SCICLK periods per data bit.

The receiver begins operation on receipt of a valid start bit. A valid start bit is identified by four consecutive internal SCICLK periods of zero bits as shown in Figure 36-7. If any bit is not zero, then the processor starts over and begins looking for another start bit.

For the bits following the start bit, the processor determines the bit value by making three samples in the middle of the bits. These samples occur on the fourth, fifth, and sixth SCICLK periods, and bit-value determination is on a majority (two out of three) basis. Figure 36-7 illustrates the asynchronous communication format for this with a start bit showing where a majority vote is taken.

Since the receiver synchronizes itself to frames, the external transmitting and receiving devices do not have to use a synchronized serial clock. The clock can be generated locally.

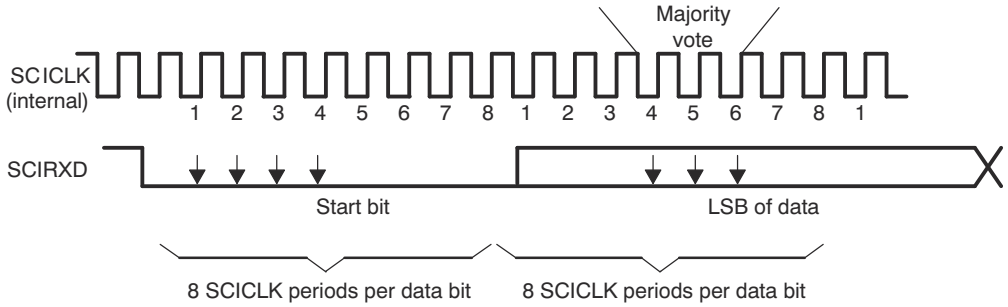


Figure 36-7. SCI Asynchronous Communications Format

36.10.1 Receiver Signals in Communication Modes

Figure 36-8 illustrates an example of receiver signal timing that assumes the following conditions:

- Address-bit wake-up mode (address bit does not appear in idle-line mode)
- Six bits per character

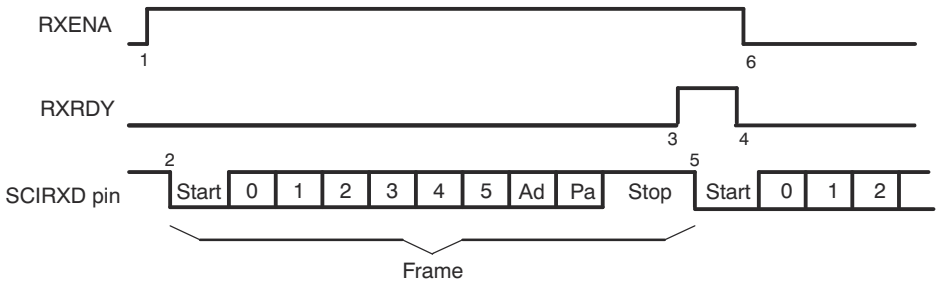


Figure 36-8. SCI RX Signals in Communication Modes

Notes:

1. Flag bit RXENA (SCICTL1, bit 0) goes high to enable the receiver.
2. Data arrives on the SCIRXD pin, start bit detected.
3. Data is shifted from RXSHF to the receiver buffer register (SCIRXBUF); an interrupt is requested. Flag bit RXRDY (SCIRXST, bit 6) goes high to signal that a new character has been received.
4. The program reads SCIRXBUF; flag RXRDY is automatically cleared.
5. The next byte of data arrives on the SCIRXD pin; the start bit is detected, then cleared.
6. Bit RXENA is brought low to disable the receiver. Data continues to be assembled in RXSHF but is not transferred to the receiver buffer register.

36.10.2 Transmitter Signals in Communication Modes

Figure 36-9 illustrates an example of transmitter signal timing that assumes the following conditions:

- Address-bit wake-up mode (address bit does not appear in idle-line mode)
- Three bits per character

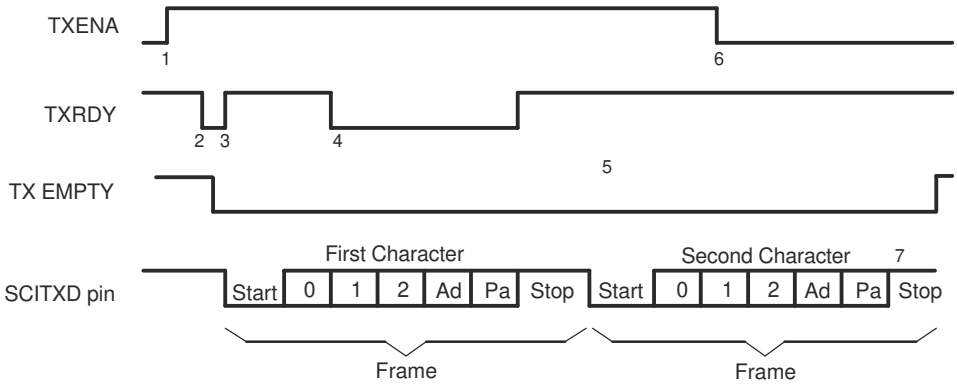


Figure 36-9. SCI TX Signals in Communications Mode

Notes:

1. Bit TXENA (SCICTL1, bit 1) goes high, enabling the transmitter to send data.
2. SCITXBUF is written to; thus, (1) the transmitter is no longer empty, and (2) TXRDY goes low.
3. The SCI transfers data to the shift register (TXSHF). The transmitter is ready for a second character (TXRDY goes high), and it requests an interrupt (to enable an interrupt, bit TX INT ENA — SCICTL2, bit 0 — must be set).
4. The program writes a second character to SCITXBUF after TXRDY goes high (item 3). (TXRDY goes low again after the second character is written to SCITXBUF.)
5. Transmission of the first character is complete. Transfer of the second character to shift register TXSHF begins.
6. Bit TXENA goes low to disable the transmitter; the SCI finishes transmitting the current character.
7. Transmission of the second character is complete; transmitter is empty and ready for new character.

36.11 SCI Port Interrupts

The SCI receiver and transmitter can be interrupt controlled. The SCICTL2 register has one flag bit (TXRDY) that indicates active interrupt conditions, and the SCIRXST register has two interrupt flag bits (RXRDY and BRKDT), plus the RX ERROR interrupt flag that is a logical-OR of the FE, OE, BRKDT, and PE conditions. The transmitter and receiver have separate interrupt-enable bits. When not enabled, the interrupts are not asserted; however, the condition flags remain active, reflecting transmission and receipt status.

The SCI has independent peripheral interrupt vectors for the receiver and transmitter. Peripheral interrupt requests can be either high priority or low priority. This is indicated by the priority bits that are output from the peripheral to the PIE controller. When both RX and TX interrupt requests are made at the same priority level, the receiver always has higher priority than the transmitter, reducing the possibility of receiver overrun.

The operation of peripheral interrupts is described in the Peripheral Interrupts section of the *System Control and Interrupts* chapter.

- If the RX/BK INT ENA bit (SCICTL2, bit 1) is set, the receiver peripheral interrupt request is asserted when one of the following events occurs:
  - The SCI receives a complete frame and transfers the data in the RXSHF register to the SCIRXBUF register. This action sets the RXRDY flag (SCIRXST, bit 6) and initiates an interrupt.
  - A break detect condition occurs (the SCIRXD is low for 9.625 bit periods following a missing stop bit). This action sets the BRKDT flag bit (SCIRXST, bit 5) and initiates an interrupt.
- If the TX INT ENA bit (SCICTL2.0) is set, the transmitter peripheral interrupt request is asserted whenever the data in the SCITXBUF register is transferred to the TXSHF register, indicating that the CPU can write to SCITXBUF; this action sets the TXRDY flag bit (SCICTL2, bit 7) and initiates an interrupt.

Note

Interrupt generation due to the RXRDY and BRKDT bits is controlled by the RX/BK INT ENA bit (SCICTL2, bit 1). Interrupt generation due to the RX ERROR bit is controlled by the RX ERR INT ENA bit (SCICTL1, bit 6).

36.12 SCI Baud Rate Calculations

The internally generated serial clock is determined by the low-speed peripheral clock LSPCLK) and the baud-select registers. The SCI uses the 16-bit value of the baud-select registers to select one of the 64K different serial clock rates possible for a given LSPCLK.

See the bit descriptions in the baud-select registers, for the formula to use when calculating the SCI asynchronous baud. Table 36-3 shows the baud-select values for common SCI bit rates. LSPCLK/16 is the maximum baud rate. For example, if LSPCLK is 100 MHz, then the maximum baud rate is 6.25 Mbps.

Table 36-3. Asynchronous Baud Register Values for Common SCI Bit Rates

	LSPCLK Clock Frequency, 100 MHz		
Ideal Baud	**BRR**	**Actual Baud**	**% Error**
2400	5207 (1457h)	2400	0
4800	2603 (A2Bh)	4800	0
9600	1301 (515h)	9601	0.01
19200	650 (28Ah)	19201	0.01
38400	324 (144h)	38462	0.16

36.13 SCI Enhanced Features

The C28x SCI features autobaud detection and transmit/receive FIFO. The following section explains the FIFO operation.

36.13.1 SCI FIFO Description

The following steps explain the FIFO features and help with programming the SCI with FIFOs.

1. **Reset.** At reset the SCI powers up in standard SCI mode and the FIFO function is disabled. The FIFO registers SCIFFTX, SCIFFRX, and SCIFFCT remain inactive.
2. **Standard SCI.** The standard SCI modes work normally with TXINT/RXINT interrupts as the interrupt source for the module.
3. **FIFO enable.** FIFO mode is enabled by setting the SCIFFEN bit in the SCIFFTX register. SCIRST can reset the FIFO mode at any stage of the operation.
4. **Active registers.** All the SCI registers and SCI FIFO registers (SCIFFTX, SCIFFRX, and SCIFFCT) are active.
5. **Interrupts.** FIFO mode has two interrupts; transmit FIFO (TXINT) and receive FIFO (RXINT). The RXINT is the common interrupt for SCI FIFO receive, receive error, and receive FIFO overflow conditions. The TXINT of the standard SCI is disabled and this interrupt serves as SCI transmit FIFO interrupt.
6. **Buffers.** Transmit and receive buffers are supplemented with two 16-level FIFOs. The transmit FIFO registers are 8-bits wide and receive FIFO registers are 10-bits wide. The one-word transmit buffer (SCITXBUF) of the standard SCI functions as a transition buffer before the transmit FIFO and shift register. SCITXBUF is loaded into either the FIFO (when FIFO is enabled) or the TXSHF (when FIFO is disabled). When FIFO is enabled, SCITXBUF loads into the FIFO only after the last bit of the shift register is shifted out, so SCITXBUF cannot be treated as an additional level of buffer. With the FIFO enabled, TXSHF is directly loaded from the FIFO (not TXBUF) after an optional delay value (SCIFFCT). When FIFO mode is enabled for SCI, characters written to SCITXBUF are queued in to SCI-TXFIFO and the characters received in SCI-RXFIFO can be read using SCIRXBUF.
7. **Delayed transfer.** The rate that words in the FIFO are transferred to the transmit shift register is programmable. The SCIFFCT register bits (7−0) FFTXDLY7−FFTXDLY0 define the delay between the word transfer. The delay is defined in the number SCI baud clock cycles. The 8 bit register can define a minimum delay of 0 baud clock cycles and a maximum of 256-baud clock cycles. With zero delay, the SCI module can transmit data in continuous mode with the FIFO words shifting out back to back. With the 256 clock delay the SCI module can transmit data in a maximum delayed mode with the FIFO words shifting out with a delay of 256 baud clocks between each words. The programmable delay facilitates communication with slow SCI/UARTs with little CPU intervention.
8. **FIFO status bits.** Both the transmit and receive FIFOs have status bits TXFFST or RXFFST (bits 12−8) that define the number of words available in the FIFOs at any time. The transmit FIFO reset bit TXFIFO and receive reset bit RXFIFO reset the FIFO pointers to zero when these bits are cleared to 0. The FIFOs resumes operation from start once these bits are set to 1.
9. **Programmable interrupt levels.** Both transmit and receive FIFO can generate CPU interrupts. The interrupt trigger is generated whenever the transmit FIFO status bits TXFFST (bits 12−8) match (less than or equal to) the interrupt trigger level bits TXFFIL (bits 4−0). This provides a programmable interrupt trigger for transmit and receive sections of the SCI. Default value for these trigger level bits is 0x11111 for receive FIFO and 0x00000 for transmit FIFO, respectively.

Figure 36-10 and Table 36-4 explain the operation/configuration of SCI interrupts in nonFIFO/FFO mode.

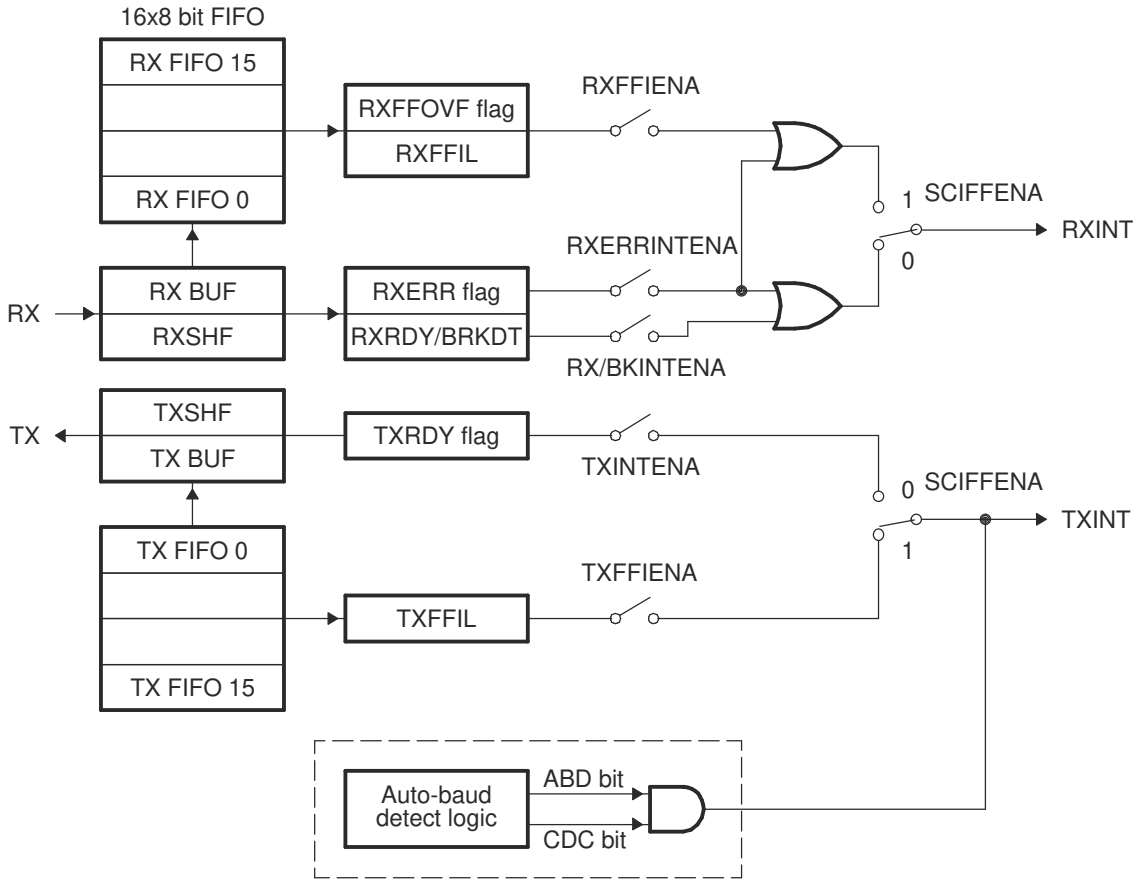


Figure 36-10. SCI FIFO Interrupt Flags and Enable Logic

Table 36-4. SCI Interrupt Flags

FIFO Options[1]	SCI Interrupt Source	Interrupt Flags	Interrupt Enables	FIFO Enable SCIFFENA	Interrupt Line
SCI without FIFO	Receive error	RXERR[2]	RXERRINTENA	0	RXINT
	Receive break	BRKDT	RX/BKINTENA	0	RXINT
	Data receive	RXRDY	RX/BKINTENA	0	RXINT
	Transmit empty	TXRDY	TXINTENA	0	TXINT
SCI with FIFO	Receive error and receive break	RXERR	RXERRINTENA	1	RXINT
	FIFO receive	RXFFIL	RXFFIENA	1	RXINT
	Transmit empty	TXFFIL	TXFFIENA	1	TXINT
Auto-baud	Auto-baud detected	ABD	Don't care	x	TXINT

(1) FIFO mode TXSHF is directly loaded after delay value, TXBUF is not used.
(2) RXERR can be set by BRKDT, FE, OE, PE flags. In FIFO mode, BRKDT interrupt is only through RXERR flag.

36.13.2 SCI Auto-Baud

Most SCI modules do not have an auto-baud detect logic built-in hardware. These SCI modules are integrated with embedded controllers whose clock rates are dependent on PLL reset values. Often embedded controller clocks change after final design. In the enhanced feature set this module supports an autobaud-detect logic in hardware. The following section explains the enabling sequence for autobaud-detect feature.

36.13.3 Autobaud-Detect Sequence

Bits ABD and CDC in SCIFFCT control the autobaud logic. The SCIRST bit can be enabled to make autobaud logic work.

If ABD is set while CDC is 1, which indicates auto-baud alignment, SCI transmit FIFO interrupt occurs (TXINT). After the interrupt service, the CDC bit must be cleared by software. If CDC remains set even after interrupt service, there can be no repeat interrupts.

1. Enable autobaud-detect mode for the SCI by setting the CDC bit (bit 13) in SCIFFCT and clearing the ABD bit (Bit 15) by writing a 1 to ABDCLR bit (bit 14).
2. Initialize the baud register to be 1 or less than a baud rate limit of 500 Kbps.
3. Allow SCI to receive either character "A" or "a" from a host at the desired baud rate. If the first character is either "A" or "a", the autobaud-detect hardware detects the incoming baud rate and sets the ABD bit.
4. The auto-detect hardware updates the baud rate register with the equivalent baud value hex. The logic also generates an interrupt to the CPU.
5. Respond to the interrupt clear ADB bit by writing a 1 to ABD CLR (bit 14) of SCIFFCT register and disable further autobaud locking by clearing CDC bit by writing a 0.
6. Read the receive buffer for character "A" or "a" to empty the buffer and buffer status.
7. If ABD is set while CDC is 1, which indicates autobaud alignment, the SCI transmit FIFO interrupt occurs (TXINT). After the interrupt service, the CDC bit must be cleared by software.

Note

At higher baud rates, the slew rate of the incoming data bits can be affected by transceiver and connector performance. While normal serial communications can work well, this slew rate can limit reliable autobaud detection at higher baud rates (typically beyond 100k baud) and cause the auto-baudlock feature to fail.

To avoid this, the following is recommended:

- Achieve a baud-lock between the host and C28x SCI boot loader using a lower baud rate.
- The host can then handshake with the loaded C28x application to set the SCI baud rate register to the desired higher baud rate.

36.14 Software

36.14.1 SCI Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/sci

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

36.14.1.1 Tune Baud Rate via UART Example

FILE: baud_tune_via_uart.c

This example demonstrates the process of tuning the UART/SCI baud rate of a C2000 device based on the UART input from another device. As UART does not have a clock signal, reliable communication requires baud rates to be reasonably matched. This example addresses cases where a clock mismatch between devices is greater than is acceptable for communications, requiring baud compensation between boards. As reliable communication only requires matching the EFFECTIVE baud rate, it does not matter which of the two boards executes the tuning (the board with the less-accurate clock source does not need to be the one to tune; as long as one of the two devices tunes to the other, then proper communication can be established).

To tune the baud rate of this device, SCI data (of the desired baud rate) must be sent to this device.
The input SCI baud rate must be within the +/- MARGINPERCENT of the TARGETBAUD chosen below.
These two variables are defined below, and should be chosen based on the application requirements. Higher MARGINPERCENT will allow more data to be considered "correct" in noisy conditions, and may decrease accuracy. The TARGETBAUD is what was expected to be the baud rate, but due to clock differences, needs to be tuned for better communication robustness with the other device.

NOTE: Lower baud rates have more granularity in register options, and therefore tuning is more affective at these speeds.

External Connections for Control Card
- SCIA_RX/eCAP1 is on GPIO9, connect to incoming SCI communications
- SCIA_TX is on GPIO8, for observation externally

Watch Variables
- *avgBaud* - Baud rate that was detected and set after tuning

36.14.1.2 SCI FIFO Digital Loop Back

FILE: sci_ex1_loopback.c

This program uses the internal loop back test mode of the peripheral. Other then boot mode pin configuration, no other hardware configuration is required. The pinmux and SCI modules are configured through the sysconfig file.

This test uses the loopback test mode of the SCI module to send characters starting with 0x00 through 0xFF. The test will send a character and then check the receive buffer for a correct match.

Watch Variables
- *loopCount* - Number of characters sent
- *errorCount* - Number of errors detected
- *sendChar* - Character sent
- *receivedChar* - Character received

36.14.1.3 Watchdog Reset - C28X_DUAL

FILE: sci_ex1_sysconfig_cpu2.c

This example shows how to configure the watchdog to reset CPU2 which will trigger an NMI in CPU1. LED1 is toggled at the start of main indicating CPU reset.

External Connections
- None.

Watch Variables

- loopCount - The number of loops performed while not in ISR

36.14.1.4 NMI handling - C28X_DUAL

FILE: sci_ex1_sysconfig_cpu1.c

This example demonstrates how to handle an NMI.

The watchdog of CPU2 is configured to reset the core once the watchdog overflows and in the CPU1 the NMI is triggered. The NMI status is read and is verified to be due to CPU2 Watchdog reset. The NMI ISR reboots the CPU2 core and the process is repeated.

Watch Variables

- nmi_isr_count Indicates the number of times the NMI ISR was hit because of CPU2 watchdog reset.

36.14.1.5 SCI Digital Loop Back with Interrupts

FILE: sci_ex2_loopback_interrupts.c

This test uses the internal loop back test mode of the peripheral. Other then boot mode pin configuration, no other hardware configuration is required. Both interrupts and the SCI FIFOs are used.

A stream of data is sent and then compared to the received stream. The SCI-A sent data looks like this:
00 01
01 02
02 03
....
FE FF
FF 00
etc..
The pattern is repeated forever.

Watch Variables

- *sDataA* - Data being sent
- *rDataA* - Data received
- *rDataPointA* - Keep track of where we are in the data stream. This is used to check the incoming data

36.14.1.6 SCI Echoback

FILE: sci_ex3_echoback.c

This test receives and echo-backs data through the SCI-A port.

A terminal such as 'putty' can be used to view the data from the SCI and to send information to the SCI. Characters received by the SCI port are sent back to the host.

Running the Application Open a COM port with the following settings using a terminal:

- Find correct COM port
- Bits per second = 9600
- Data Bits = 8
- Parity = None
- Stop Bits = 1
- Hardware Control = None

The program will print out a greeting and then ask you to enter a character which it will echo back to the terminal.

Watch Variables

- loopCounter - the number of characters sent

External Connections
Connect the USB cable from Control card J1:A to PC

36.14.1.7 stdout redirect example

FILE: sci_ex4_stdout_redirect.c This test transmits data through the SCI-A port to a terminal

A terminal such as 'putty' can be used to view the data from the SCI. Characters received by the SCI port are sent back to the host.

Running the Application Open a COM port with the following settings using a terminal:

- Find correct COM port
- Bits per second = 9600
- Data Bits = 8
- Parity = None
- Stop Bits = 1
- Hardware Control = None

The program will print out three sentences: one to the SCIA, one to CCS, and a final one to SCIA.

External Connections
Connect the SCI-A port to a PC via a transceiver and cable.

- DEVICE_GPIO_PIN_SCIRXDA is SCI_A-RXD (Connect to Pin3, PC-TX, of serial DB9 cable)
- DEVICE_GPIO_PIN_SCITXDA is SCI_A-TXD (Connect to Pin2, PC-RX, of serial DB9 cable)

36.15 SCI Registers

The section describes the Serial Communication Interface module registers.

36.15.1 SCI Base Address Table (C28)

Table 36-5. SCI Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
SciaRegs	SCI_REGS	SCIA_BASE	0x0000_7200	YES	YES	-	-	YES
ScibRegs	SCI_REGS	SCIB_BASE	0x0000_7210	YES	YES	-	-	YES
ScicRegs	SCI_REGS	SCIC_BASE	0x0000_7220	YES	YES	-	-	YES
ScidRegs	SCI_REGS	SCID_BASE	0x0000_7230	YES	YES	-	-	YES

36.15.2 SCI_REGS Registers

Table 36-6 lists the memory-mapped registers for the SCI_REGS registers. All register offset addresses not listed in Table 36-6 should be considered as reserved locations and the register contents should not be modified.

Table 36-6. SCI_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	SCICCR	Communications control register		Go
1h	SCICTL1	Control register 1		Go
2h	SCIHBAUD	Baud rate (high) register		Go
3h	SCILBAUD	Baud rate (low) register		Go
4h	SCICTL2	Control register 2		Go
5h	SCIRXST	Receive status register		Go
6h	SCIRXEMU	Receive emulation buffer register		Go
7h	SCIRXBUF	Receive data buffer		Go
9h	SCITXBUF	Transmit data buffer		Go
Ah	SCIFFTX	FIFO transmit register		Go
Bh	SCIFFRX	FIFO receive register		Go
Ch	SCIFFCT	FIFO control register		Go
Fh	SCIPRI	SCI priority control		Go

Complex bit access types are encoded to fit into small table cells. Table 36-7 shows the codes that are used for access types in this section.

Table 36-7. SCI_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

36.15.2.1 SCICCR Register (Offset = 0h) [Reset = 0h]

SCICCR is shown in Figure 36-11 and described in Table 36-8.

Return to the Summary Table.

SCICCR defines the character format, protocol, and communications mode used by the SCI.

Figure 36-11. SCICCR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
STOPBITS	PARITY	PARITYENA	LOOPBKENA	ADDRIDLE_MODE	SCICHAR		
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h		

Table 36-8. SCICCR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	STOPBITS	R/W	0h	SCI number of stop bits. This bit specifies the number of stop bits transmitted. The receiver checks for only one stop bit. Reset type: SYSRSn 0h (R/W) = One stop bit 1h (R/W) = Two stop bits
6	PARITY	R/W	0h	SCI parity odd/even selection. If the PARITY ENABLE bit (SCICCR, bit 5) is set, PARITY (bit 6) designates odd or even parity (odd or even number of bits with the value of 1 in both transmitted and received characters). Reset type: SYSRSn 0h (R/W) = Odd parity 1h (R/W) = Even parity
5	PARITYENA	R/W	0h	SCI parity enable. This bit enables or disables the parity function. If the SCI is in the addressbit multiprocessor mode (set using bit 3 of this register), the address bit is included in the parity calculation (if parity is enabled). For characters of less than eight bits, the remaining unused bits should be masked out of the parity calculation. Reset type: SYSRSn 0h (R/W) = Parity disabled no parity bit is generated during transmission or is expected during reception 1h (R/W) = Parity is enabled
4	LOOPBKENA	R/W	0h	Loop Back test mode enable. This bit enables the Loop Back test mode where the Tx pin is internally connected to the Rx pin. Reset type: SYSRSn 0h (R/W) = Loop Back test mode disabled 1h (R/W) = Loop Back test mode enabled
3	ADDRIDLE_MODE	R/W	0h	SCI multiprocessor mode control bit. This bit selects one of the multiprocessor protocols.Multiprocessor communication is different from the other communication modes because it uses SLEEP and TXWAKE functions (bits SCICTL1, bit 2 and SCICTL1, bit 3, respectively). The idle-line mode is usually used for normal communications because the address-bit mode adds an extra bit to the frame. The idle-line mode does not add this extra bit and is compatible with RS-232 type communications. Reset type: SYSRSn 0h (R/W) = Idle-line mode protocol selected 1h (R/W) = Address-bit mode protocol selected

Table 36-8. SCICCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	SCICHAR	R/W	0h	Character-length control bits 2-0. These bits select the SCI character length from one to eight bits. Characters of less than eight bits are right-justified in SCIRXBUF and SCIRXEMU and are padded with leading zeros in SCIRXBUF. SCITXBUF doesn't need to be padded with leading zeros. Reset type: SYSRSn 0h (R/W) = SCICHAR_LENGTH_1 1h (R/W) = SCICHAR_LENGTH_2 2h (R/W) = SCICHAR_LENGTH_3 3h (R/W) = SCICHAR_LENGTH_4 4h (R/W) = SCICHAR_LENGTH_5 5h (R/W) = SCICHAR_LENGTH_6 6h (R/W) = SCICHAR_LENGTH_7 7h (R/W) = SCICHAR_LENGTH_8

36.15.2.2 SCICTL1 Register (Offset = 1h) [Reset = 0h]

SCICTL1 is shown in Figure 36-12 and described in Table 36-9.

Return to the Summary Table.

SCICTL1 controls the receiver/transmitter enable, TXWAKE and SLEEP functions, and the SCI software reset.

Figure 36-12. SCICTL1 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RXERRINTENA	SWRESET	RESERVED	TXWAKE	SLEEP	TXENA	RXENA
R-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 36-9. SCICTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6	RXERRINTENA	R/W	0h	SCI receive error interrupt enable. Setting this bit enables an interrupt if the RX ERROR bit (SCIRXST, bit 7) becomes set because of errors occurring. Reset type: SYSRSn 0h (R/W) = Receive error interrupt disabled 1h (R/W) = Receive error interrupt enabled
5	SWRESET	R/W	0h	SCI software reset (active low). Writing a 0 to this bit initializes the SCI state machines and operating flags (registers SCICTL2 and SCIRXST) to the reset condition. The SW RESET bit does not affect any of the configuration bits. All affected logic is held in the specified reset state until a 1 is written to SW RESET (the bit values following a reset are shown beneath each register diagram in this section). Thus, after a system reset, re-enable the SCI by writing a 1 to this bit. Clear this bit after a receiver break detect (BRKDT flag, bit SCIRXST, bit 5). SW RESET affects the operating flags of the SCI, but it neither affects the configuration bits nor restores the reset values. Once SW RESET is asserted, the flags are frozen until the bit is deasserted. The affected flags are as follows: Value After SW SCI Flag Register Bit RESET 1 TXRDY SCICTL2, bit 7 1 TX EMPTY SCICTL2, bit 6 0 RXWAKE SCIRXST, bit 1 0 PE SCIRXST, bit 2 0 OE SCIRXST, bit 3 0 FE SCIRXST, bit 4 0 BRKDT SCIRXST, bit 5 0 RXRDY SCIRXST, bit 6 0 RX ERROR SCIRXST, bit 7 Reset type: SYSRSn 0h (R/W) = Writing a 0 to this bit initializes the SCI state machines and operating flags (registers SCICTL2 and SCIRXST) to the reset condition. 1h (R/W) = After a system reset, re-enable the SCI by writing a 1 to this bit.
4	RESERVED	R	0h	Reserved

Table 36-9. SCICTL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	TXWAKE	R/W	0h	SCI transmitter wake-up method select. The TXWAKE bit controls selection of the data-transmit feature, depending on which transmit mode (idle-line or address-bit) is specified at the ADDR/IDLE MODE bit (SCICCR, bit 3) Reset type: SYSRSn 0h (R/W) = Transmit feature is not selected. In idle-line mode: write a 1 to TXWAKE, then write data to register SCITXBUF to generate an idle period of 11 data bits In address-bit mode: write a 1 to TXWAKE, then write data to SCITXBUF to set the address bit for that frame to 1 1h (R/W) = Transmit feature selected is dependent on the mode, idle-line or address-bit: TXWAKE is not cleared by the SW RESET bit (SCICTL1, bit 5) it is cleared by a system reset or the transfer of TXWAKE to the WUT flag.
2	SLEEP	R/W	0h	SCI sleep. The TXWAKE bit controls selection of the data-transmit feature, depending on which transmit mode (idle-line or address-bit) is specified at the ADDR/IDLE MODE bit (SCICCR, bit 3). In a multiprocessor configuration, this bit controls the receiver sleep function. Clearing this bit brings the SCI out of the sleep mode. The receiver still operates when the SLEEP bit is set however, operation does not update the receiver buffer ready bit (SCIRXST, bit 6, RXRDY) or the error status bits (SCIRXST, bit 5-2: BRKDT, FE, OE, and PE) unless the address byte is detected. SLEEP is not cleared when the address byte is detected. Reset type: SYSRSn 0h (R/W) = Sleep mode disabled 1h (R/W) = Sleep mode enabled
1	TXENA	R/W	0h	SCI transmitter enable. Data is transmitted through the SCITXD pin only when TXENA is set. If reset, transmission is halted but only after all data previously written to SCITXBUF has been sent. Data written into SCITXBUF when TXENA is disabled will not be transmitted even if the TXENA is enabled later. Reset type: SYSRSn 0h (R/W) = Transmitter disabled 1h (R/W) = Transmitter enabled
0	RXENA	R/W	0h	SCI receiver enable. Data is received on the SCIRXD pin and is sent to the receiver shift register and then the receiver buffers. This bit enables or disables the receiver (transfer to the buffers). Clearing RXENA stops received characters from being transferred to the two receiver buffers and also stops the generation of receiver interrupts. However, the receiver shift register can continue to assemble characters. Thus, if RXENA is set during the reception of a character, the complete character will be transferred into the receiver buffer registers, SCIRXEMU and SCIRXBUF. Reset type: SYSRSn 0h (R/W) = Prevent received characters from transfer into the SCIRXEMU and SCIRXBUF receiver buffers 1h (R/W) = Send received characters to SCIRXEMU and SCIRXBUF

36.15.2.3 SCIHBAUD Register (Offset = 2h) [Reset = 0h]

SCIHBAUD is shown in Figure 36-13 and described in Table 36-10.

Return to the Summary Table.

The values in SCIHBAUD and SCILBAUD specify the baud rate for the SCI.

Figure 36-13. SCIHBAUD Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
BAUD							
R/W-0h							

Table 36-10. SCIHBAUD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	BAUD	R/W	0h	SCI 16-bit baud selection Registers SCIHBAUD (MSbyte). The internally-generated serial clock is determined by the low speed peripheral clock (LSPCLK) signal and the two baud-select registers. The SCI uses the 16-bit value of these registers to select one of 64K serial clock rates for the communication modes. BRR = (SCIHBAUD << 8) + (SCILBAUD) The SCI baud rate is calculated using the following equation: SCI Asynchronous Baud = LSPCLK / ((BRR + 1) *8) Alternatively, BRR = LSPCLK / (SCI Asynchronous Baud * 8) - 1 Note that the above formulas are applicable only when 0 < BRR < 65536. If BRR = 0, then SCI Asynchronous Baud = LSPCLK / 16 Where: BRR = the 16-bit value (in decimal) in the baud-select registers Reset type: SYSRSn

36.15.2.4 SCILBAUD Register (Offset = 3h) [Reset = 0h]

SCILBAUD is shown in Figure 36-14 and described in Table 36-11.

Return to the Summary Table.

The values in SCIHBAUD and SCILBAUD specify the baud rate for the SCI.

Figure 36-14. SCILBAUD Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
BAUD							
R/W-0h							

Table 36-11. SCILBAUD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	BAUD	R/W	0h	See SCIHBAUD Detailed Description Reset type: SYSRSn

36.15.2.5 SCICTL2 Register (Offset = 4h) [Reset = C0h]

SCICTL2 is shown in Figure 36-15 and described in Table 36-12.

Return to the Summary Table.

SCICTL2 enables the receive-ready, break-detect, and transmit-ready interrupts as well as transmitter-ready and -empty flags.

Figure 36-15. SCICTL2 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
TXRDY	TXEMPTY	RESERVED				RXBKINTENA	TXINTENA
R-1h	R-1h	R-0h				R/W-0h	R/W-0h

Table 36-12. SCICTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	TXRDY	R	1h	Transmitter buffer register ready flag. When set, this bit indicates that the transmit data buffer register, SCITXBUF, is ready to receive another character. Writing data to the SCITXBUF automatically clears this bit. When set, this flag asserts a transmitter interrupt request if the interrupt-enable bit, TX INT ENA (SCICTL2.0), is also set. TXRDY is set to 1 by enabling the SW RESET bit (SCICTL1.5) or by a system reset. Reset type: SYSRSn 0h (R/W) = SCITXBUF is full 1h (R/W) = SCITXBUF is ready to receive the next character
6	TXEMPTY	R	1h	Transmitter empty flag. This flag's value indicates the contents of the transmitter's buffer register (SCITXBUF) and shift register (TXSHF). An active SW RESET (SCICTL1.5), or a system reset, sets this bit. This bit does not cause an interrupt request. Reset type: SYSRSn 0h (R/W) = Transmitter buffer or shift register or both are loaded with data 1h (R/W) = Transmitter buffer and shift registers are both empty
5-2	RESERVED	R	0h	Reserved
1	RXBKINTENA	R/W	0h	Receiver-buffer/break interrupt enable. This bit controls the interrupt request caused by either the RXRDY flag or the BRKDT flag (bits SCIRXST.6 and .5) being set. However, RX/BK INT ENA does not prevent the setting of these flags. Reset type: SYSRSn 0h (R/W) = Disable RXRDY/BRKDT interrupt 1h (R/W) = Enable RXRDY/BRKDT interrupt

Table 36-12. SCICTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	TXINTENA	R/W	0h	SCITXBUF-register interrupt enable. This bit controls the interrupt request caused by the setting of TXRDY flag bit (SCICTL2.7). However, it does not prevent the TXRDY flag from being set (which indicates SCITXBUF is ready to receive another character). 0 Disable TXRDY interrupt 1 Enable TXRDY interrupt. In non-FIFO mode, a dummy (or a valid) data has to be written to SCITXBUF for the first transmit interrupt to occur. This is the case when you enable the transmit interrupt for the first time and also when you re-enable (disable and then enable) the transmit interrupt. If TXINTENA is enabled after writing the data to SCITXBUF, it will not generate an interrupt. Reset type: SYSRSn 0h (R/W) = Disable TXRDY interrupt 1h (R/W) = Enable TXRDY interrupt

36.15.2.6 SCIRXST Register (Offset = 5h) [Reset = 0h]

SCIRXST is shown in Figure 36-16 and described in Table 36-13.

Return to the Summary Table.

SCIRXST contains seven bits that are receiver status flags (two of which can generate interrupt requests). Each time a complete character is transferred to the receiver buffers (SCIRXEMU and SCIRXBUF), the status flags are updated.

Figure 36-16. SCIRXST Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RXERROR	RXRDY	BRKDT	FE	OE	PE	RXWAKE	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 36-13. SCIRXST Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	RXERROR	R	0h	SCI receiver error flag. The RX ERROR flag indicates that one of the error flags in the receiver status register is set. RX ERROR is a logical OR of the break detect, framing error, overrun, and parity error enable flags (bits 5-2: BRKDT, FE, OE, and PE). A 1 on this bit will cause an interrupt if the RX ERR INT ENA bit (SCICTL1.6) is set. This bit can be used for fast error-condition checking during the interrupt service routine. This error flag cannot be cleared directly it is cleared by an active SW RESET or by a system reset. Reset type: SYSRSn 0h (R/W) = No error flags set 1h (R/W) = Error flag(s) set
6	RXRDY	R	0h	SCI receiver-ready flag. When a new character is ready to be read from the SCIRXBUF register, the receiver sets this bit, and a receiver interrupt is generated if the RX/BK INT ENA bit (SCICTL2.1) is a 1. RXRDY is cleared by a reading of the SCIRXBUF register, by an active SW RESET, or by a system reset. Reset type: SYSRSn 0h (R/W) = No new character in SCIRXBUF 1h (R/W) = Character ready to be read from SCIRXBUF
5	BRKDT	R	0h	SCI break-detect flag. The SCI sets this bit when a break condition occurs. A break conditionoccurs when the SCI receiver data line (SCIRXD) remains continuously low for at least 9.625 bits, beginning after a missing first stop bit. The occurrence of a break causes a receiver interrupt to be generated if the RX/BK INT ENA bit is a 1, but it does not cause the receiver buffer to be loaded. A BRKDT interrupt can occur even if the receiver SLEEP bit is set to 1. BRKDT is cleared by an active SW RESET or by a system reset. It is not cleared by receipt of a character after the break is detected. In order to receive more characters, the SCI must be reset by toggling the SW RESET bit or by a system reset. Reset type: SYSRSn 0h (R/W) = No break condition 1h (R/W) = Break condition occurred

Table 36-13. SCIRXST Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	FE	R	0h	SCI framing-error flag. The SCI sets this bit when an expected stop bit is not found. Only the first stop bit is checked. The missing stop bit indicates that synchronization with the start bit has been lost and that the character is incorrectly framed. The FE bit is reset by a clearing of the SW RESET bit or by a system reset. NOTE: FE will be flagged prior to BRKDT, except when RX is in sleep mode. In sleep mode, when there is no RX WAKEUP and RXD line is low for greater than 10 bits, BRKDT will be flagged while FE will not be flagged. Reset type: SYSRSn 0h (R/W) = No framing error detected 1h (R/W) = Framing error detected
3	OE	R	0h	SCI overrun-error flag. The SCI sets this bit when a character is transferred into registers SCIRXEMU and SCIRXBUF before the previous character is fully read by the CPU or DMAC. The previous character is overwritten and lost. The OE flag bit is reset by an active SW RESET or by a system reset. Reset type: SYSRSn 0h (R/W) = No overrun error detected 1h (R/W) = Overrun error detected
2	PE	R	0h	SCI parity-error flag. This flag bit is set when a character is received with a mismatch between the number of 1s and its parity bit. The address bit is included in the calculation. If parity generation and detection is not enabled, the PE flag is disabled and read as 0. The PE bit is reset by an active SW RESET or a system reset. Reset type: SYSRSn 0h (R/W) = No parity error or parity is disabled 1h (R/W) = Parity error is detected
1	RXWAKE	R	0h	Receiver wake-up-detect flag Reset type: SYSRSn 0h (R/W) = No detection of a receiver wake-up condition 1h (R/W) = A value of 1 in this bit indicates detection of a receiver wake-up condition. In the address-bit multiprocessor mode (SCICCR.3 = 1), RXWAKE reflects the value of the address bit for the character contained in SCIRXBUF. In the idle-line multiprocessor mode, RXWAKE is set if the SCIRXD data line is detected as idle. RXWAKE is a read-only flag, cleared by one of the following: - The transfer of the first byte after the address byte to SCIRXBUF (only in non-FIFO mode) - The reading of SCIRXBUF - An active SW RESET - A system reset
0	RESERVED	R	0h	Reserved

36.15.2.7 SCIRXEMU Register (Offset = 6h) [Reset = 0h]

SCIRXEMU is shown in Figure 36-17 and described in Table 36-14.

Return to the Summary Table.

Normal SCI data-receive operations read the data received from the SCIRXBUF register. The SCIRXEMU register is used principally by the emulator (EMU) because it can continuously read the data received for screen updates without clearing the RXRDY flag. SCIRXEMU is cleared by a system reset. This is the register that should be used in an emulator watch window to view the contents of the SCIRXBUF register. SCIRXEMU is not physically implemented
it is just a different address location to access the SCIRXBUF register without clearing the RXRDY flag.

Figure 36-17. SCIRXEMU Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
ERXDT							
R-0h							

Table 36-14. SCIRXEMU Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	ERXDT	R	0h	Receive emulation buffer data Reset type: SYSRSn

36.15.2.8 SCIRXBUF Register (Offset = 7h) [Reset = 0h]

SCIRXBUF is shown in Figure 36-18 and described in Table 36-15.

Return to the Summary Table.

When the current data received is shifted from RXSHF to the receiver buffer, flag bit RXRDY is set and the data is ready to be read. If the RXBKINTENA bit (SCICTL2.1) is set, this shift also causes an interrupt. When SCIRXBUF is read, the RXRDY flag is reset. SCIRXBUF is cleared by a system reset.

Figure 36-18. SCIRXBUF Register

15	14	13	12	11	10	9	8
SCIFFFE	SCIFFPE	RESERVED					
R-0h	R-0h	R-0h					
7	6	5	4	3	2	1	0
SAR							
R-0h							

Table 36-15. SCIRXBUF Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SCIFFFE	R	0h	SCIFFFE. SCI FIFO Framing error flag bit (applicable only if the FIFO is enabled) Reset type: SYSRSn 0h (R/W) = No frame error occurred while receiving the character, in bits 7-0. This bit is associated with the character on the top of the FIFO. 1h (R/W) = A frame error occurred while receiving the character in bits 7-0. This bit is associated with the character on the top of the FIFO.
14	SCIFFPE	R	0h	SCIFFPE. SCI FIFO parity error flag bit (applicable only if the FIFO is enabled) Reset type: SYSRSn 0h (R/W) = No parity error occurred while receiving the character, in bits 7-0. This bit is associated with the character on the top of the FIFO. 1h (R/W) = A parity error occurred while receiving the character in bits 7-0. This bit is associated with the character on the top of the FIFO.
13-8	RESERVED	R	0h	Reserved
7-0	SAR	R	0h	Receive Character bits Reset type: SYSRSn

36.15.2.9 SCITXBUF Register (Offset = 9h) [Reset = 0h]

SCITXBUF is shown in Figure 36-19 and described in Table 36-16.

Return to the Summary Table.

Data bits to be transmitted are written to SCITXBUF. These bits must be rightjustified because the leftmost bits are ignored for characters less than eight bits long. The transfer of data from this register to the TXSHF transmitter shift register sets the TXRDY flag (SCICTL2.7), indicating that SCITXBUF is ready to receive another set of data. If bit TXINTENA (SCICTL2.0) is set, this data transfer also causes an interrupt.

Figure 36-19. SCITXBUF Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
TXDT							
R/W-0h							

Table 36-16. SCITXBUF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	TXDT	R/W	0h	Transmit data buffer Reset type: SYSRSn

36.15.2.10 SCIFFTX Register (Offset = Ah) [Reset = A000h]

SCIFFTX is shown in Figure 36-20 and described in Table 36-17.

Return to the Summary Table.

SCIFFTX controls the transmit FIFO interrupt, FIFO enhancements, and reset for the SCI transmit and receive channels.

Figure 36-20. SCIFFTX Register

15	14	13	12	11	10	9	8
SCIRST	SCIFFENA	TXFIFORESET	TXFFST				
R/W-1h	R/W-0h	R/W-1h	R-0h				
7	6	5	4	3	2	1	0
TXFFINT	TXFFINTCLR	TXFFIENA	TXFFIL				
R-0h	R-0/W1S-0h	R/W-0h	R/W-0h				

Table 36-17. SCIFFTX Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SCIRST	R/W	1h	SCI Reset 0 Write 0 to reset the SCI transmit and receive channels. SCI FIFO register configuration bits will be left as is. 1 SCI FIFO can resume transmit or receive. SCIRST should be 1 even for Autobaud logic to work. Reset type: SYSRSn
14	SCIFFENA	R/W	0h	SCI FIFO enable Reset type: SYSRSn 0h (R/W) = SCI FIFO enhancements are disabled 1h (R/W) = SCI FIFO enhancements are enabled
13	TXFIFORESET	R/W	1h	Transmit FIFO reset Reset type: SYSRSn 0h (R/W) = Reset the FIFO pointer to zero and hold in reset 1h (R/W) = Re-enable transmit FIFO operation
12-8	TXFFST	R	0h	FIFO status Reset type: SYSRSn 0h (R/W) = Transmit FIFO is empty 1h (R/W) = Transmit FIFO has 1 words 2h (R/W) = Transmit FIFO has 2 words 3h (R/W) = Transmit FIFO has 3 words 4h (R/W) = Transmit FIFO has 4 words 5h (R/W) = Transmit FIFO has 5 words 6h (R/W) = Transmit FIFO has 6 words 7h (R/W) = Transmit FIFO has 7 words 8h (R/W) = Transmit FIFO has 8 words 9h (R/W) = Transmit FIFO has 9 words Ah (R/W) = Transmit FIFO has 10 words Bh (R/W) = Transmit FIFO has 11 words Ch (R/W) = Transmit FIFO has 12 words Dh (R/W) = Transmit FIFO has 13 words Eh (R/W) = Transmit FIFO has 14 words Fh (R/W) = Transmit FIFO has 15 words 10h (R/W) = Transmit FIFO has 16 words
7	TXFFINT	R	0h	Transmit FIFO interrupt Reset type: SYSRSn 0h (R/W) = TXFIFO interrupt has not occurred, read-only bit 1h (R/W) = TXFIFO interrupt has occurred, read-only bit
6	TXFFINTCLR	R-0/W1S	0h	Transmit FIFO clear Reset type: SYSRSn 0h (R/W) = Write 0 has no effect on TXFIFINT flag bit, Bit reads back a zero 1h (R/W) = Write 1 to clear TXFFINT flag in bit 7

Table 36-17. SCIFFTX Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	TXFFIENA	R/W	0h	Transmit FIFO interrrupt enable Reset type: SYSRSn 0h (R/W) = TX FIFO interrupt is disabled 1h (R/W) = TX FIFO interrupt is enabled. This interrupt is triggered whenever the transmit FIFO status (TXFFST) bits match (equal to or less than) the interrupt trigger level bits TXFFIL (bits 4-0).
4-0	TXFFIL	R/W	0h	TXFFIL4-0 Transmit FIFO interrupt level bits. The transmit FIFO generates an interrupt whenever the FIFO status bits (TXFFST4-0) are less than or equal to the FIFO level bits (TXFFIL4-0). The maximum value that can be assigned to these bits to generate an interrupt cannot be more than the depth of the TX FIFO. The default value of these bits after reset is 00000b. Users should set TXFFIL to best fit their application needs by weighing between the CPU overhead to service the ISR and the best possible usage of SCI bus bandwidth. Reset type: SYSRSn

36.15.2.11 SCIFFRX Register (Offset = Bh) [Reset = 201Fh]

SCIFFRX is shown in Figure 36-21 and described in Table 36-18.

Return to the Summary Table.

SCIFFRX controls the receive FIFO interrupt, receive FIFO reset, and status of the receive FIFO overflow.

Figure 36-21. SCIFFRX Register

15	14	13	12	11	10	9	8
RXFFOVF	RXFFOVRCLR	RXFIFORESET	RXFFST				
R-0h	R-0/W1S-0h	R/W-1h	R-0h				
7	6	5	4	3	2	1	0
RXFFINT	RXFFINTCLR	RXFFIENA	RXFFIL				
R-0h	W-0h	R/W-0h	R/W-1Fh				

Table 36-18. SCIFFRX Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RXFFOVF	R	0h	Receive FIFO overflow. This will function as flag, but cannot generate interrupt by itself. This condition will occur while receive interrupt is active. Receive interrupts should service this flag condition. Reset type: SYSRSn 0h (R/W) = Receive FIFO has not overflowed, read-only bit 1h (R/W) = Receive FIFO has overflowed, read-only bit. More than 16 words have been received in to the FIFO, and the first received word is lost
14	RXFFOVRCLR	R-0/W1S	0h	RXFFOVF clear Reset type: SYSRSn 0h (R/W) = Write 0 has no effect on RXFFOVF flag bit, Bit reads back a zero 1h (R/W) = Write 1 to clear RXFFOVF flag in bit 15
13	RXFIFORESET	R/W	1h	Receive FIFO reset Reset type: SYSRSn 0h (R/W) = Write 0 to reset the FIFO pointer to zero, and hold in reset. 1h (R/W) = Re-enable receive FIFO operation
12-8	RXFFST	R	0h	FIFO status Reset type: SYSRSn 0h (R/W) = Receive FIFO is empty 1h (R/W) = Receive FIFO has 1 words 2h (R/W) = Receive FIFO has 2 words 3h (R/W) = Receive FIFO has 3 words 4h (R/W) = Receive FIFO has 4 words 5h (R/W) = Receive FIFO has 5 words 6h (R/W) = Receive FIFO has 6 words 7h (R/W) = Receive FIFO has 7 words 8h (R/W) = Receive FIFO has 8 words 9h (R/W) = Receive FIFO has 9 words Ah (R/W) = Receive FIFO has 10 words Bh (R/W) = Receive FIFO has 11 words Ch (R/W) = Receive FIFO has 12 words Dh (R/W) = Receive FIFO has 13 words Eh (R/W) = Receive FIFO has 14 words Fh (R/W) = Receive FIFO has 15 words 10h (R/W) = Receive FIFO has 16 words
7	RXFFINT	R	0h	Receive FIFO interrupt Reset type: SYSRSn 0h (R/W) = RXFIFO interrupt has not occurred, read-only bit 1h (R/W) = RXFIFO interrupt has occurred, read-only bit

Table 36-18. SCIFFRX Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	RXFFINTCLR	W	0h	Receive FIFO interrupt clear Reset type: SYSRSn 0h (R/W) = Write 0 has no effect on RXFIFINT flag bit. Bit reads back a zero. 1h (R/W) = Write 1 to clear RXFFINT flag in bit 7
5	RXFFIENA	R/W	0h	Receive FIFO interrupt enable Reset type: SYSRSn 0h (R/W) = RX FIFO interrupt is disabled 1h (R/W) = RX FIFO interrupt is enabled. This interrupt is triggered whenever the receive FIFO status (RXFFST) bits match (equal to or greater than) the interrupt trigger level bits RXFFIL (bits 4-0).
4-0	RXFFIL	R/W	1Fh	Receive FIFO interrupt level bits The receive FIFO generates an interrupt whenever the FIFO status bits (RXFFST4-0) are greater than or equal to the FIFO level bits (RXFFIL4-0). The maximum value that can be assigned to these bits to generate an interrupt cannot be more than the depth of the RX FIFO. The default value of these bits after reset is 11111b. Users should set RXFFIL to best fit their application needs by weighing between the CPU overhead to service the ISR and the best possible usage of received SCI data. Reset type: SYSRSn

36.15.2.12 SCIFFCT Register (Offset = Ch) [Reset = 0h]

SCIFFCT is shown in Figure 36-22 and described in Table 36-19.

Return to the Summary Table.

SCIFFCT contains the status of auto-baud detect, clears the auto-baud flag, and calibrate for A-detect bit.

Figure 36-22. SCIFFCT Register

15	14	13	12	11	10	9	8
ABD	ABDCLR	CDC	RESERVED				
R-0h	W-0h	R/W-0h	R-0h				
7	6	5	4	3	2	1	0
FFTXDLY							
R/W-0h							

Table 36-19. SCIFFCT Register Field Descriptions

Bit	Field	Type	Reset	Description
15	ABD	R	0h	Auto-baud detect (ABD) bit Reset type: SYSRSn 0h (R/W) = Auto-baud detection is not complete. "A","a" character has not been received successfully. 1h (R/W) = Auto-baud hardware has detected "A" or "a" character on the SCI receive register. Auto-detect is complete.
14	ABDCLR	W	0h	ABD-clear bit Reset type: SYSRSn 0h (R/W) = Write 0 has no effect on ABD flag bit. Bit reads back a zero. 1h (R/W) = Write 1 to clear ABD flag in bit 15.
13	CDC	R/W	0h	CDC calibrate A-detect bit Reset type: SYSRSn 0h (R/W) = Disables auto-baud alignment 1h (R/W) = Enables auto-baud alignment
12-8	RESERVED	R	0h	Reserved
7-0	FFTXDLY	R/W	0h	FIFO transfer delay. These bits define the delay between every transfer from FIFO transmit bufferto transmit shift register. The delay is defined in the number of SCI serial baud clock cycles. The 8 bit register could define a minimum delay of 0 baud clock cycles and a maximum of 256 baud clock cycles In FIFO mode, the buffer (TXBUF) between the shift register and the FIFO should be filled only after the shift register has completed shifting of the last bit. This is required to pass on the delay between transfers to the data stream. In FIFO mode, TXBUF should not be treated as one additional level of buffer. The delayed transmit feature will help to create an auto-flow scheme without RTS/CTS controls as in standard UARTS. When SCI is configured for one stop-bit, delay introduced by FFTXDLY between one frame and the next frame is equal to number of baud clock cycles that FFTXDLY is set to. When SCI is configured for two stop-bits, delay introduced by FFTXDLY between one frame and the next frame is equal to number of baud clock cycles that FFTXDLY is set to minus 1. Reset type: SYSRSn

36.15.2.13 SCIPRI Register (Offset = Fh) [Reset = 0h]

SCIPRI is shown in Figure 36-23 and described in Table 36-20.

Return to the Summary Table.

SCIPRI determines what happens when an emulation suspend event occurs.

Figure 36-23. SCIPRI Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			FREESOFT		RESERVED		
R-0h			R/W-0h		R-0h		

Table 36-20. SCIPRI Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-5	RESERVED	R	0h	Reserved
4-3	FREESOFT	R/W	0h	These bits determine what occurs when an emulation suspend event occurs (for example, when the debugger hits a breakpoint). The peripheral can continue whatever it is doing (free-run mode), or if in stop mode, it can either stop immediately or stop when the current operation (the current receive/transmit sequence) is complete. Reset type: SYSRSn 0h (R/W) = Immediate stop on suspend 1h (R/W) = Complete current receive/transmit sequence before stopping 2h (R/W) = Free run 3h (R/W) = Free run
2-0	RESERVED	R	0h	Reserved

36.15.3 SCI Registers to Driverlib Functions

Table 36-21. SCI Registers to Driverlib Functions

File	Driverlib Function
SCICCR	
sci.c	SCI_setConfig
sci.h	SCI_setParityMode
sci.h	SCI_getParityMode
sci.h	SCI_setAddrMultiProcessorMode
sci.h	SCI_setIdleMultiProcessorMode
sci.h	SCI_getConfig
sci.h	SCI_enableLoopback
sci.h	SCI_disableLoopback
SCICTL1	
sci.c	SCI_enableInterrupt
sci.c	SCI_disableInterrupt
sci.c	SCI_setWakeFlag
sci.h	SCI_enableModule
sci.h	SCI_disableModule
sci.h	SCI_enableTxModule
sci.h	SCI_disableTxModule

Table 36-21. SCI Registers to Driverlib Functions (continued)

File	Driverlib Function
sci.h	SCI_enableRxModule
sci.h	SCI_disableRxModule
sci.h	SCI_enableSleepMode
sci.h	SCI_disableSleepMode
sci.h	SCI_performSoftwareReset
SCIHBAUD	
sci.c	SCI_setConfig
sci.c	SCI_setBaud
sci.h	SCI_lockAutobaud
sci.h	SCI_getConfig
SCILBAUD	
sci.c	SCI_setConfig
sci.c	SCI_setBaud
sci.h	SCI_lockAutobaud
sci.h	SCI_getConfig
SCICTL2	
sci.c	SCI_enableInterrupt
sci.c	SCI_disableInterrupt
sci.c	SCI_getInterruptStatus
sci.h	SCI_isSpaceAvailableNonFIFO
sci.h	SCI_isTransmitterBusy
SCIRXST	
sci.c	SCI_getInterruptStatus
sci.h	SCI_isDataAvailableNonFIFO
sci.h	SCI_getRxStatus
SCIRXEMU	
-	
SCIRXBUF	
sci.c	SCI_readCharArray
sci.h	SCI_readCharBlockingFIFO
sci.h	SCI_readCharBlockingNonFIFO
sci.h	SCI_readCharNonBlocking
SCITXBUF	
sci.c	SCI_writeCharArray
sci.h	SCI_writeCharBlockingFIFO
sci.h	SCI_writeCharBlockingNonFIFO
sci.h	SCI_writeCharNonBlocking
SCIFFTX	
sci.c	SCI_enableInterrupt
sci.c	SCI_disableInterrupt
sci.c	SCI_getInterruptStatus
sci.c	SCI_clearInterruptStatus
sci.h	SCI_setFIFOInterruptLevel
sci.h	SCI_getFIFOInterruptLevel
sci.h	SCI_disableModule

Table 36-21. SCI Registers to Driverlib Functions (continued)

File	Driverlib Function
sci.h	SCI_enableFIFO
sci.h	SCI_disableFIFO
sci.h	SCI_isFIFOEnabled
sci.h	SCI_resetTxFIFO
sci.h	SCI_resetChannels
sci.h	SCI_getTxFIFOStatus
sci.h	SCI_isTransmitterBusy
SCIFFRX	
sci.c	SCI_enableInterrupt
sci.c	SCI_disableInterrupt
sci.c	SCI_getInterruptStatus
sci.c	SCI_clearInterruptStatus
sci.h	SCI_setFIFOInterruptLevel
sci.h	SCI_getFIFOInterruptLevel
sci.h	SCI_enableFIFO
sci.h	SCI_resetRxFIFO
sci.h	SCI_getRxFIFOStatus
sci.h	SCI_getOverflowStatus
sci.h	SCI_clearOverflowStatus
SCIFFCT	
sci.h	SCI_lockAutobaud
SCIPRI	
-	

This page intentionally left blank.

Chapter 37

Serial Peripheral Interface (SPI)

This chapter describes the serial peripheral interface (SPI) which is a high-speed synchronous serial input and output (I/O) port that allows a serial bit stream of programmed length (one to 16 bits) to be shifted into and out of the device at a programmed bit-transfer rate. The SPI is normally used for communications between the MCU controller and external peripherals or another controller. Typical applications include external I/O or peripheral expansion using devices such as shift registers, display drivers, and analog-to-digital converters (ADCs). Multi-device communications are supported by the master or slave operation of the SPI. The port supports a 16-level, receive and transmit FIFO for reducing CPU servicing overhead.

37.1 Introduction

37.1.1 Features

The SPI module features include:

- SPISOMI: SPI slave-output/master-input pin
- SPISIMO: SPI slave-input/master-output pin
- $\overline{\text{SPISTE}}$: SPI slave transmit-enable pin
- SPICLK: SPI serial-clock pin

Note

All four pins can be used as GPIO if the SPI module is not used.

- Two operational modes: Master and Slave
- Baud rate: 125 different programmable rates. The maximum baud rate that can be employed is limited by the maximum speed of the I/O buffers used on the SPI pins. See the device data manual for more details.
- Data word length: 1 to 16 data bits
- Four clocking schemes (controlled by clock polarity and clock phase bits) include:
  - Falling edge without phase delay: SPICLK active-high. SPI transmits data on the falling edge of the SPICLK signal and receives data on the rising edge of the SPICLK signal.
  - Falling edge with phase delay: SPICLK active-high. SPI transmits data one half-cycle ahead of the falling edge of the SPICLK signal and receives data on the falling edge of the SPICLK signal.
  - Rising edge without phase delay: SPICLK inactive-low. SPI transmits data on the rising edge of the SPICLK signal and receives data on the falling edge of the SPICLK signal.
  - Rising edge with phase delay: SPICLK inactive-low. SPI transmits data one half-cycle ahead of the rising edge of the SPICLK signal and receives data on the rising edge of the SPICLK signal.
- Simultaneous receive and transmit operation (transmit function can be disabled in software)
- Transmitter and receiver operations are accomplished through either interrupt- driven or polled algorithm
- 16-level transmit/receive FIFO
- DMA support
- High-speed mode
- Delayed transmit control
- 3-wire SPI mode
- $\overline{\text{SPISTE}}$ inversion for digital audio interface receive mode on devices with two SPI modules

37.1.2 SPI Related Collateral

Foundational Materials

- C2000 Academy - SPI
- *KeyStone Architecture Serial Peripheral Interface (SPI)*

Getting Started Materials

- SPI: Microcontroller overview (Video)

37.1.3 Block Diagram

Figure 37-1 shows the SPI CPU interfaces.

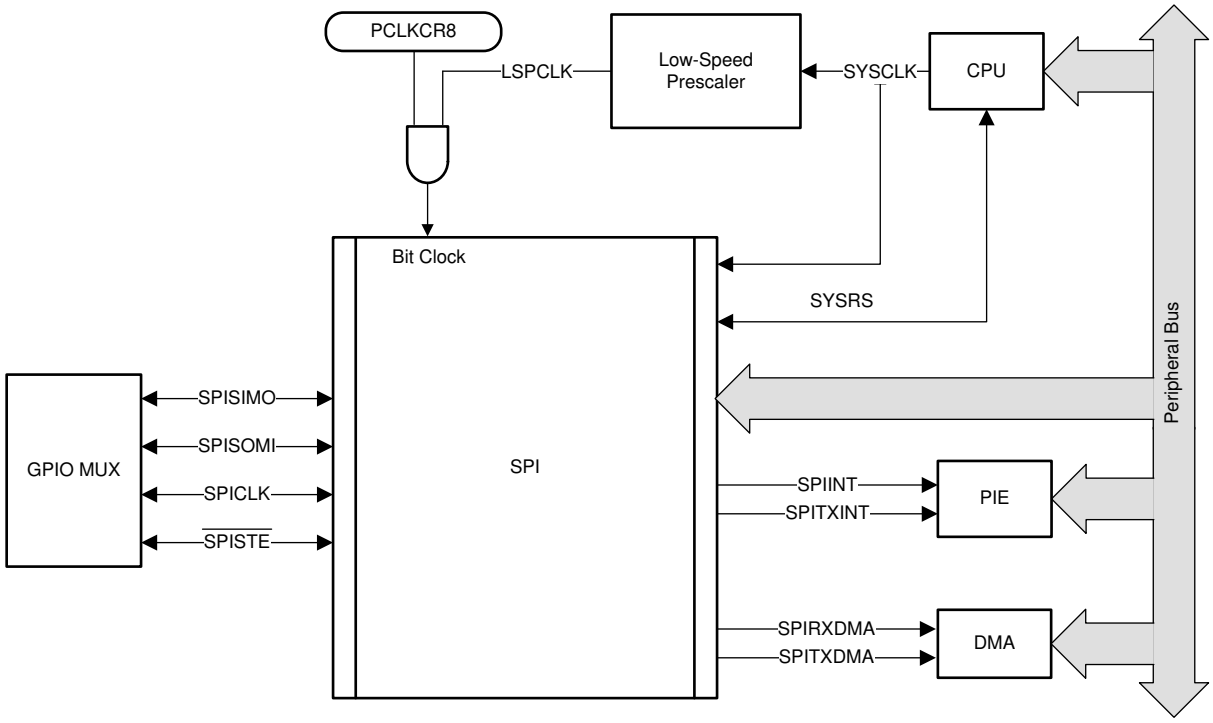


Figure 37-1. SPI CPU Interface

37.2 System-Level Integration

This section describes the various functionality that is applicable to the device integration. These features require configuration of other modules in the device that are not within the scope of this chapter.

37.2.1 SPI Module Signals

Table 37-1 classifies and provides a summary of the SPI module signals.

Table 37-1. SPI Module Signal Summary

Signal Name	Description
External Signals	
SPICLK	SPI clock
SPISIMO	SPI slave in, master out
SPISOMI	SPI slave out, master in
$\overline{\text{SPISTE}}$	SPI slave transmit enable
Control	
SPI Clock Rate	LSPCLK
Interrupt Signals	
SPIINT/SPIRXINT	Transmit interrupt/ Receive Interrupt in non FIFO mode (referred to as SPIINT)
	Receive interrupt in FIFO mode
SPITXINT	Transmit interrupt in FIFO mode
DMA Triggers	
SPITXDMA	Transmit request to DMA
SPIRXDMA	Receive request to DMA

Special Considerations

The $\overline{\text{SPISTE}}$ signal provides the ability to gate any spurious clock and data pulses when the SPI is in slave mode. A HIGH logic signal on $\overline{\text{SPISTE}}$ does not allow the slave to receive data. This prevents the SPI slave from losing synchronization with the master. TI does not recommend that the $\overline{\text{SPISTE}}$ always be tied to the active state.

If the SPI slave does ever lose synchronization with the master, toggling SPISWRESET resets the internal bit counter as well as the various status flags in the module. By resetting the bit counter, the SPI interprets the next clock transition as the first bit of a new transmission. The register bit fields that are reset by SPISWRESET are found in Section 37.6.

Configuring a GPIO to Emulate $\overline{\text{SPISTE}}$

In many systems, a SPI master can be connected to multiple SPI slaves using multiple instances of $\overline{\text{SPISTE}}$. Though this SPI module does not natively support multiple $\overline{\text{SPISTE}}$ signals, it is possible to emulate this behavior in software using GPIOs. In this configuration, the SPI must be configured as the master. Rather than using the GPIO Mux to select $\overline{\text{SPISTE}}$, the application can configure pins to be GPIO outputs, one GPIO per SPI slave. Before transmitting any data, the application can drive the desired GPIO to the active state. Immediately after the transmission has been completed, the GPIO chip select can be driven to the inactive state. This process can be repeated for many slaves that share the SPICLK, SPISIMO, and SPISOMI lines.

37.2.2 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

Some IO functionality is defined by GPIO register settings independent of this peripheral. For input signals, the GPIO input qualification must be set to asynchronous mode by setting the appropriate GPxQSELn register bits to 11b. The internal pullups can be configured in the GPyPUD register.

See the *General-Purpose Input/Output (GPIO)* chapter for more details on GPIO mux and settings.

37.2.2.1 GPIOs Required for High-Speed Mode

The high-speed mode of the SPI is available on the specified GPIO mux options in the device data sheet. To enable the high-speed enhancements, set SPICCR.HS_MODE to 1. Make sure that the capacitive loading on the pin does not exceed the value stated in the device data sheet.

When not operating in high-speed mode or if the capacitive loading on the pins exceed the value stated in the device data manual, SPICCR.HS_MODE can be set to 0.

37.2.3 SPI Interrupts

This section includes information on the available interrupts present in the SPI module. The SPI module contains two interrupt lines: SPIINT/SPIRXINT and SPITXINT. When the SPI is operating in non-FIFO mode, all available interrupts are routed together to generate the single SPIINT interrupt. When FIFO mode is used, both SPIRXINT and SPITXINT can be generated.

SPIINT/SPIRXINT

When the SPI is operating in non-FIFO mode, the interrupt generated is called SPIINT. If FIFO enhancements are enabled, the interrupt is called SPIRXINT. These interrupts share the same interrupt vector in the Interrupt Controller block.

In non-FIFO mode, two conditions can trigger an interrupt: a transmission is complete (INT_FLAG), or there is overrun in the receiver (OVERRUN_FLAG). Both of these conditions share the same interrupt vector: SPIINT.

The transmission complete flag (INT_FLAG) indicates that the SPI has completed sending or receiving the last bit and is ready to be serviced. At the same time this bit is set, the received character is placed in the receiver buffer (SPIRXBUF). The INT_FLAG generates an interrupt on the SPIINT vector if the SPIINTENA bit is set.

The receiver overrun flag (OVERRUN_FLAG) indicates that a transmit or receive operation has completed before the previous character has been read from the buffer. The OVERRUN_FLAG generates an interrupt on the SPIINT vector if the OVERRUNINTENA bit is set and OVERRUN_FLAG was previously cleared.

In FIFO mode, the SPI can interrupt the CPU upon a match condition between the current receive FIFO status (RXFFST) and the receive FIFO interrupt level (RXFFIL). If RXFFST is greater than or equal to RXFFIL, the receive FIFO interrupt flag (RXFFINT) is set. SPIRXINT is triggered in the Interrupt Controller block, if RXFFINT is set and the receive FIFO interrupt is enabled (RXFFIENA = 1).

SPITXINT

The SPITXINT interrupt is not available when the SPI is operating in non-FIFO mode.

In FIFO mode, the SPITXINT behavior is similar to the SPIRXINT. SPITXINT is generated upon a match condition between the current transmit FIFO status (TXFFST) and the transmit FIFO interrupt level (TXFFIL). If TXFFST is less than or equal to TXFFIL, the transmit FIFO interrupt flag (TXFFINT) is set. SPITXINT is triggered in the Interrupt Controller block, if TXFFINT is set and the transmit FIFO interrupt is enabled in the SPI module (TXFFIENA = 1).

Figure 37-2 and Table 37-2 show how these control bits influence the SPI interrupt generation.

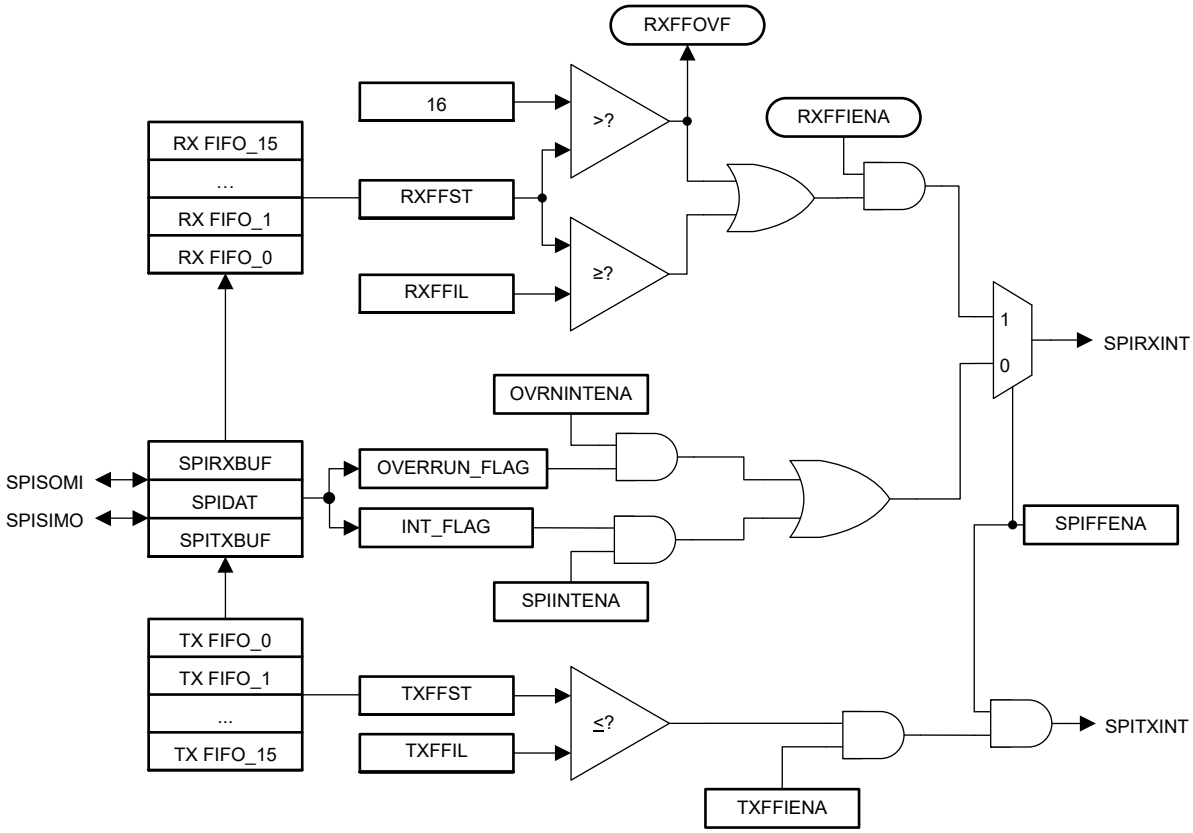


Figure 37-2. SPI Interrupt Flags and Enable Logic Generation

Table 37-2. SPI Interrupt Flag Modes

FIFO Options	SPI Interrupt Source	Interrupt Flags	Interrupt Enables	FIFO Enable (SPIFFENA)	Interrupt Line (1)
SPI without FIFO	Receive overrun	RXOVRN	OVRNINTENA	0	SPIRXINT
	Data receive	SPIINT	SPIINTENA	0	SPIRXINT
	Transmit empty	SPIINT	SPIINTENA	0	SPIRXINT
SPI FIFO mode	FIFO receive	RXFFIL	RXFFIENA	1	SPIRXINT
	Transmit empty	TXFFIL	TXFFIENA	1	SPITXINT

(1) In non-FIFO mode, SPIRXINT is the same name as the SPIINT interrupt in C28x devices.

37.2.4 DMA Support

Both the CPU and DMA have access to the SPI data registers using the internal peripheral bus. This access is limited to 16-bit register reads and writes. Each SPI module can generate two DMA events, SPITXDMA and SPIRXDMA. The DMA events are controlled by configuring the SPIFFTX.TXFFIL and SPIFFRX.RXFFIL appropriately. SPITXDMA activates when TXFFST is less than the interrupt level (TXFFIL). SPIRXDMA activates when RXFFST is greater than or equal to the interrupt level (RXFFIL).

The SPI must have FIFO enhancements enabled for the DMA triggers to be generated.

For more information on configuring the SPI for DMA transfers, refer to Section 37.3.8.

Figure 37-3 is a block diagram showing the DMA trigger generation from the SPI module.

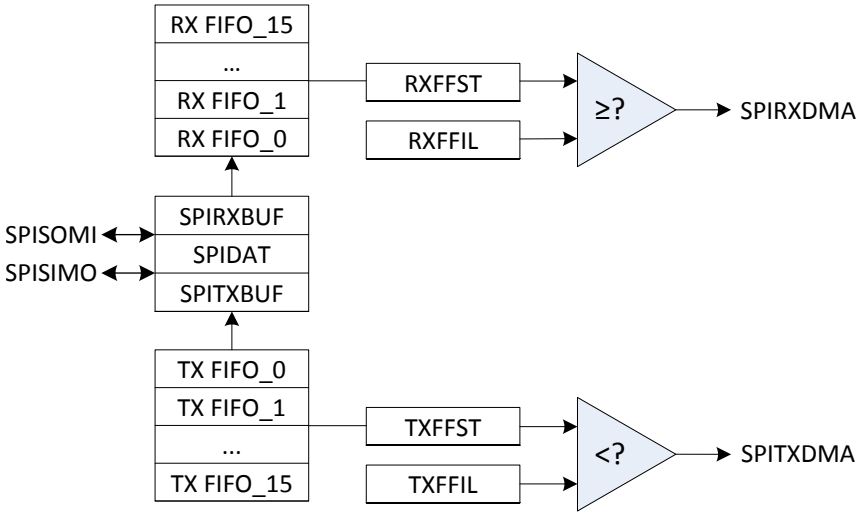


Figure 37-3. SPI DMA Trigger Diagram

37.3 SPI Operation

This section describes the various modes of operation of the SPI. Included are explanations of the operational modes, interrupts, data format, clock sources, and initialization. Typical timing diagrams for data transfers are given.

37.3.1 Introduction to Operation

Figure 37-4 shows typical connections of the SPI for communications between two controllers: a master and a slave.

The master transfers data by sending the SPICLK signal. For both the slave and the master, data is shifted out of the shift registers on one edge of the SPICLK and latched into the shift register on the opposite SPICLK clock edge. If the CLK_PHASE bit is high, data is transmitted and received a half-cycle before the SPICLK transition. As a result, both controllers send and receive data simultaneously. The application software determines whether the data is meaningful or dummy data. There are three possible methods for data transmission:

- Master sends data; slave sends dummy data.
- Master sends data; slave sends data.
- Master sends dummy data; slave sends data.

The master can initiate data transfer at any time because the master controls the SPICLK signal. The software, however, determines how the master detects when the slave is ready to broadcast data.

The SPI operates in master or slave mode. The MASTER_SLAVE bit selects the operating mode and the source of the SPICLK signal.

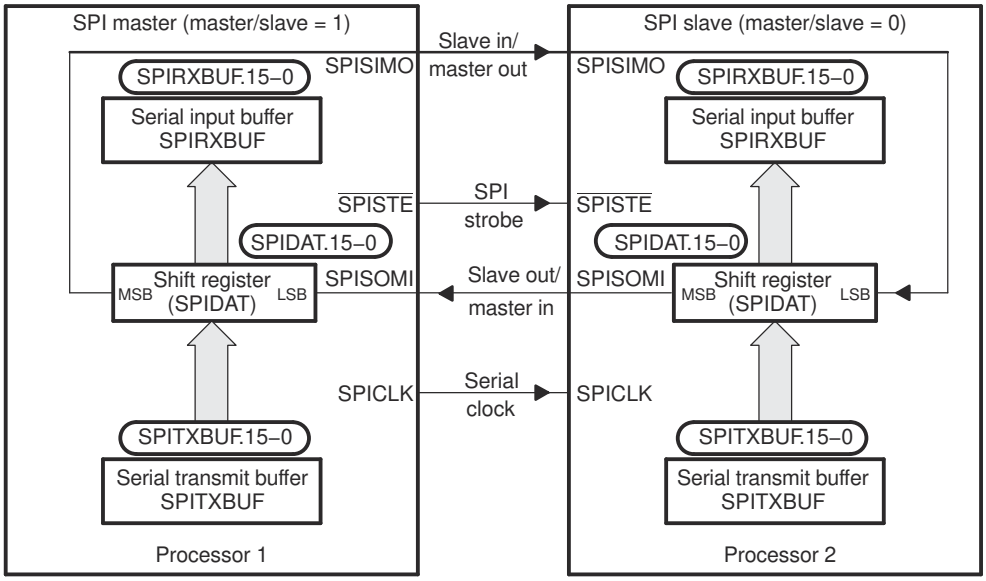


Figure 37-4. SPI Master/Slave Connection

37.3.2 Master Mode

In master mode (MASTER_SLAVE = 1), the SPI provides the serial clock on the SPICLK pin for the entire serial communications network. Data is output on the SPISIMO pin and latched from the SPISOMI pin.

The SPIBRR register determines both the transmit and receive bit transfer rate for the network. SPIBRR can select 125 different data transfer rates.

Data written to SPIDAT or SPITXBUF initiates data transmission on the SPISIMO pin, MSB (most-significant bit) first. Simultaneously, received data is shifted through the SPISOMI pin into the LSB (least- significant bit) of SPIDAT. When the selected number of bits has been transmitted, the received data is transferred to the SPIRXBUF (buffered receiver) for the CPU to read. Data is stored right-justified in SPIRXBUF.

When the specified number of data bits has been shifted through SPIDAT, the following events occur:

- SPIDAT contents are transferred to SPIRXBUF.
- INT_FLAG bit is set to 1.
- If there is valid data in the transmit buffer SPITXBUF, as indicated by the transmit buffer full flag (BUFFULL_FLAG), this data is transferred to SPIDAT and is transmitted; otherwise, SPICLK stops after all bits have been shifted out of SPIDAT.
- If the SPIINTENA bit is set to 1, an interrupt is asserted.

In a typical application, the $\overline{\text{SPISTE}}$ pin serves as a chip-enable pin for a SPI slave device. This pin is driven low by the master before transmitting data to the slave and is taken high after the transmission is complete.

Figure 37-5 is a block diagram of the SPI in master mode. The block diagram shows the basic control blocks available in SPI master mode.

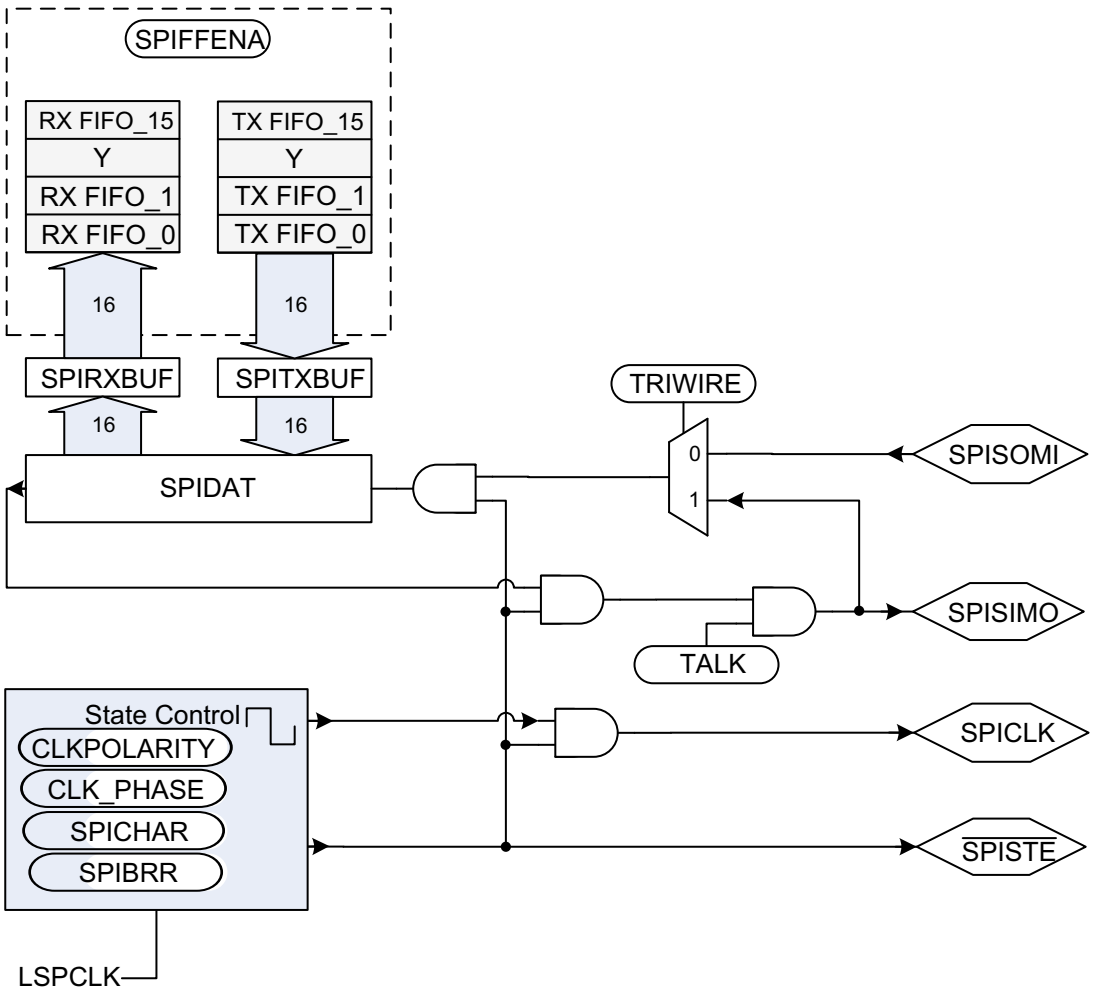


Figure 37-5. SPI Module Master Configuration

37.3.3 Slave Mode

In slave mode (MASTER_SLAVE = 0), data shifts out on the SPISOMI pin and in on the SPISIMO pin. The SPICLK pin is used as the input for the serial shift clock, which is supplied from the external network master. The transfer rate is defined by this clock. The SPICLK input frequency can be no greater than the LSPCLK frequency divided by 4.

Data written to SPIDAT or SPITXBUF is transmitted to the network when appropriate edges of the SPICLK signal are received from the network master. A character written to the SPITXBUF register is copied to the SPIDAT register when all bits of the current character in SPIDAT have been shifted out. If no character was previously copied to SPIDAT, then any character written to SPITXBUF is immediately copied to SPIDAT. If a character was previously copied to SPIDAT, any data written to SPITXBUF is not copied to SPIDAT until the current character in SPIDAT has been shifted out. To receive data, the SPI waits for the network master to send the SPICLK signal and then shifts the data on the SPISIMO pin into SPIDAT. If data is to be transmitted by the slave simultaneously, and SPIDAT has not been previously loaded, the character must be written to SPITXBUF before the beginning of the SPICLK signal.

When the TALK bit is cleared, data transmission is disabled, and the output line (SPISOMI) is put into the high-impedance state. If this occurs while a transmission is active, the current character is completely transmitted even though SPISOMI is forced into the high-impedance state. This makes sure that the SPI is still able to receive incoming data correctly. This TALK bit allows many slave devices to be tied together on the network, but only one slave at a time is allowed to drive the SPISOMI line.

The $\overline{\text{SPISTE}}$ pin operates as the slave-select pin. An active-low signal on the $\overline{\text{SPISTE}}$ pin allows the slave SPI to transfer data to the serial data line; an inactive-high signal causes the slave SPI serial shift register to stop and the serial output pin to be put into the high-impedance state. This allows many slave devices to be tied together on the network, although only one slave device is selected at a time.

Figure 37-6 is a block diagram of the SPI in slave mode. The block diagram shows the basic control blocks available in SPI slave mode.

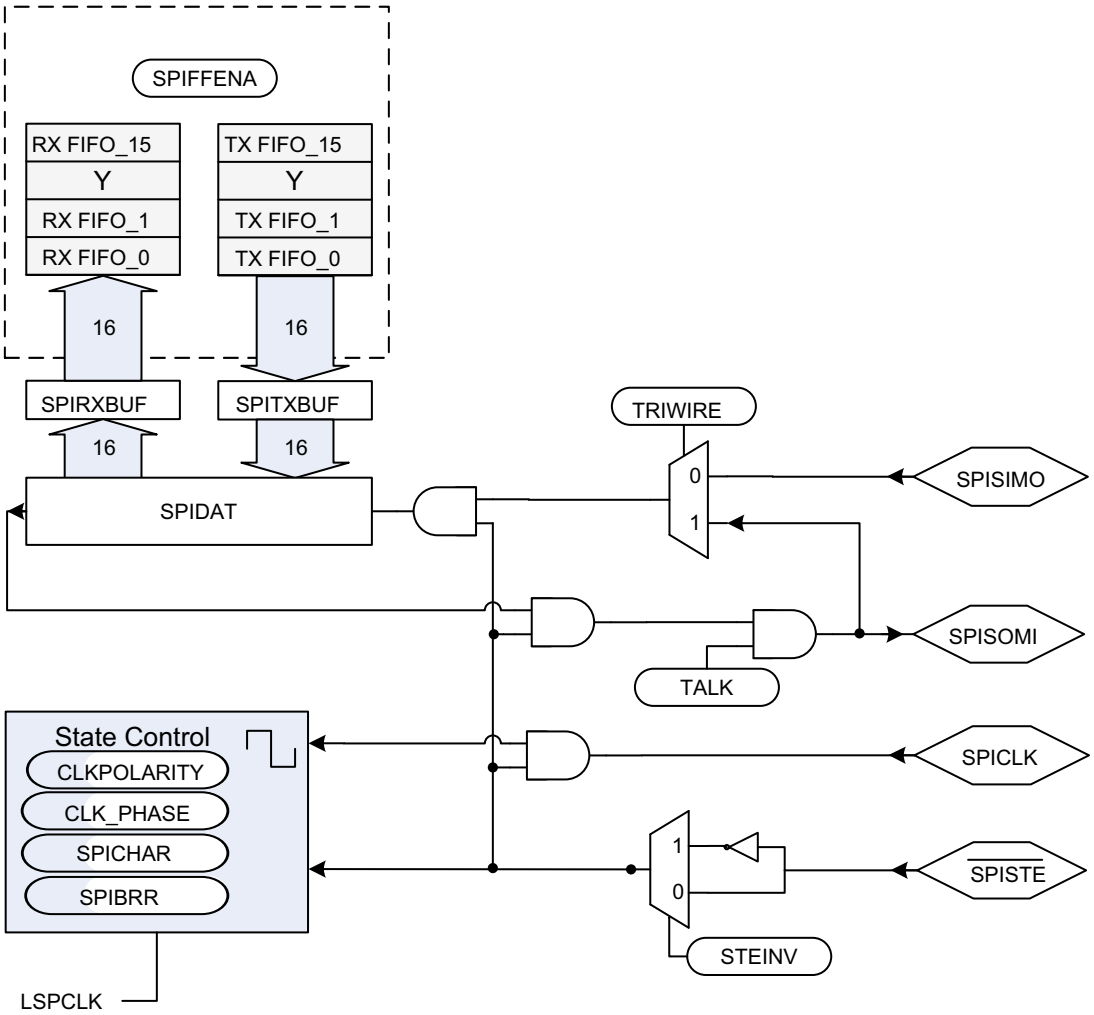


Figure 37-6. SPI Module Slave Configuration

37.3.4 Data Format

The four-bit SPICHAR register field specifies the number of bits in the data character (1 to 16). This information directs the state control logic to count the number of bits received or transmitted to determine when a complete character has been processed.

The following statements apply to characters with fewer than 16 bits:

- Data must be left-justified when written to SPIDAT and SPITXBUF.
- Data read back from SPIRXBUF is right-justified.
- SPIRXBUF contains the most recently received character, right-justified, plus any bits that remain from previous transmission(s) that have been shifted to the left (shown in Example 37-1).

Example 37-1. Transmission of Bit from SPIRXBUF

Conditions:

1. Transmission character length = 1 bit (specified in SPICHAR bits)
2. The current value of SPIDAT = 737Bh

	SPIDAT (before transmission)																
	0	1	1	1	0	0	1	1	0	1	1	1	1	0	1	1	
	SPIDAT (after transmission)																
(TXed) 0 ←	1	1	1	0	0	1	1	0	1	1	1	1	0	1	1	x(1)	← (RXed)
	SPIRXBUF (after transmission)																
	1	1	1	0	0	1	1	0	1	1	1	1	0	1	1	x(1)	

(1) x = 1, if SPISOMI data is high; x = 0, if SPISOMI data is low; master mode is assumed.

37.3.5 Baud Rate Selection

The SPI module supports 125 different baud rates and four different clock schemes. Depending on whether the SPI clock is in slave or master mode, the SPICLK pin can receive an external SPI clock signal or provide the SPI clock signal, respectively.

- In the slave mode, the SPI clock is received on the SPICLK pin from the external source and can be no greater than the LSPCLK frequency divided by 4.
- In the master mode, the SPI clock is generated by the SPI and is output on the SPICLK pin and can be no greater than the LSPCLK frequency divided by 4.

Note

The baud rate must be configured to not exceed the maximum rated GPIO toggle frequency. Refer to the device data sheet for the maximum GPIO toggle frequency.

Example 37-2 shows how to determine the SPI baud rates.

Example 37-3 shows how to calculate the baud rate of the SPI module in standard SPI mode (HS_MODE = 0).

Example 37-2. Baud Rate Determination

For SPIBRR = 3 to 127:

$$\text{SPI Baud Rate} = \frac{\text{LSPCLK}}{(\text{SPIBRR} + 1)}$$

For SPIBRR = 0, 1, or 2:

$$\text{SPI Baud Rate} = \frac{\text{LSPCLK}}{4}$$

where:

LSPCLK = Low-speed peripheral clock frequency of the device

SPIBRR = Contents of the SPIBRR in the master SPI device

To determine what value to load into SPIBRR, you must know the device system clock (LSPCLK) frequency (that is device-specific) and the baud rate at which you are operating.

Example 37-3. Baud Rate Calculation in Non-High Speed Mode (HS_MODE = 0)

$$SPI\ Baud\ Rate = \frac{LSPCLK}{SPIBRR + 1'} \quad LSPCLK = 50\ MHz$$

$$= \frac{50 \times 10^6}{3 + 1}$$

$$= 12.5\ Mbps$$

37.3.6 SPI Clocking Schemes

The clock polarity select bit (CLKPOLARITY) and the clock phase select bit (CLK_PHASE) control four different clocking schemes on the SPICLK pin. CLKPOLARITY selects the active edge, either rising or falling, of the clock. CLK_PHASE selects a half-cycle delay of the clock. The four different clocking schemes are:

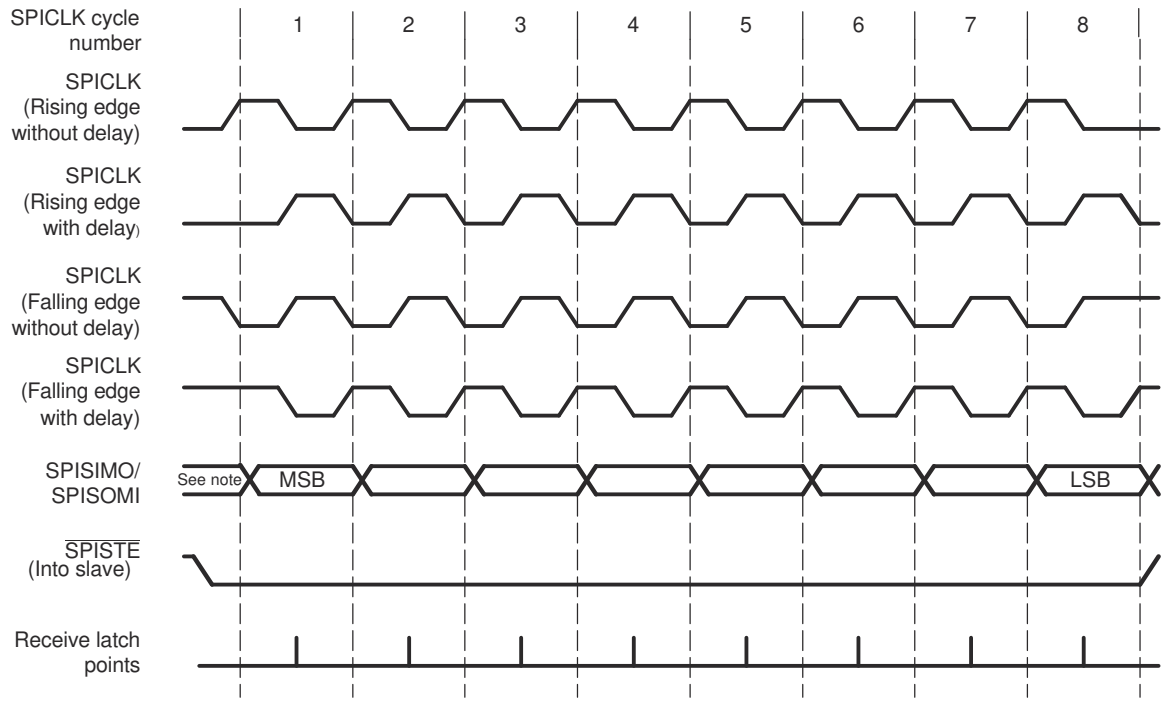
- Falling Edge Without Delay. The SPI transmits data on the falling edge of the SPICLK and receives data on the rising edge of the SPICLK.
- Falling Edge With Delay. The SPI transmits data one half-cycle ahead of the falling edge of the SPICLK signal and receives data on the falling edge of the SPICLK signal.
- Rising Edge Without Delay. The SPI transmits data on the rising edge of the SPICLK signal and receives data on the falling edge of the SPICLK signal.
- Rising Edge With Delay. The SPI transmits data one half-cycle ahead of the rising edge of the SPICLK signal and receives data on the rising edge of the SPICLK signal.

The selection procedure for the SPI clocking scheme is shown in Table 37-3. Examples of these four clocking schemes relative to transmitted and received data are shown in Figure 37-7.

Table 37-3. SPI Clocking Scheme Selection Guide

SPICLK Scheme	CLKPOLARITY(1)	CLK_PHASE(1)
Rising edge without delay	0	0
Rising edge with delay	0	1
Falling edge without delay	1	0
Falling edge with delay	1	1

(1) The description of CLK_PHASE and CLKPOLARITY differs between manufacturers. For proper operation, select the desired waveform to determine the clock phase and clock polarity settings.



Note: Previous data bit

Figure 37-7. SPICLK Signal Options

SPICLK symmetry is retained only when the result of (SPIBRR + 1) is an even value. When (SPIBRR + 1) is an odd value and SPIBRR is greater than 3, SPICLK becomes asymmetrical. The low pulse of SPICLK is one LSPCLK cycle longer than the high pulse when CLKPOLARITY bit is clear (0). When CLKPOLARITY bit is set to 1, the high pulse of the SPICLK is one LSPCLK cycle longer than the low pulse, as shown in Figure 37-8.

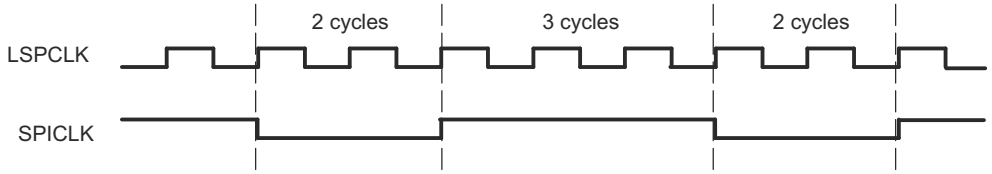


Figure 37-8. SPI: SPICLK-LSPCLK Characteristic when (BRR + 1) is Odd, BRR > 3, and CLKPOLARITY = 1

37.3.7 SPI FIFO Description

The following steps explain the FIFO features and help with programming the SPI FIFOs:

1. **Reset.** At reset the SPI powers up in standard SPI mode and the FIFO function is disabled. The FIFO registers SPIFFTX, SPIFFRX and SPIFFCT remain inactive.
2. **Standard SPI.** The standard 28x SPI mode works with SPIINT/SPIRXINT as the interrupt source.
3. **Mode change.** FIFO mode is enabled by setting the SPIFFENA bit to 1 in the SPIFFTX register. SPIRST can reset the FIFO mode at any stage of the operation.
4. **Active registers.** All the SPI registers and SPI FIFO registers SPIFFTX, SPIFFRX, and SPIFFCT are active.
5. **Interrupts.** FIFO mode has two interrupts: one for the transmit FIFO, SPITXINT; one for the receive FIFO, SPIRXINT. SPIRXINT is the common interrupt for SPI FIFO receive, receive error and receive FIFO overflow conditions. The single SPIINT for both transmit and receive sections of the standard SPI are disabled and this interrupt is serviced as SPI receive FIFO interrupt. For more information, refer to Section 37.2.3.
6. **Buffers.** Transmit and receive buffers are each supplemented with a 16-word FIFO. The one-word transmit buffer (SPITXBUF) of the standard SPI functions as a transition buffer between the transmit FIFO and shift register. The one-word transmit buffer is loaded from transmit FIFO only after the last bit of the shift register is shifted out.
7. **Delayed transfer.** The rate at which transmit words in the FIFO are transferred to transmit shift register is programmable. The SPIFFCT register bits (7−0) FFTXDLY7−FFTXDLY0 define the delay between the word transfer. The delay is defined in number SPI serial clock cycles. The 8-bit register can define a minimum delay of 0 SPICLK cycles and a maximum of 255 SPICLK cycles. With zero delay, the SPI module can transmit data in continuous mode with the FIFO words shifting out back to back. With the 255 clock delay, the SPI module can transmit data in a maximum delayed mode with the FIFO words shifting out with a delay of 255 SPICLK cycles between each words. The programmable delay facilitates glueless interface to various slow SPI peripherals, such as EEPROMs, ADC, DAC, and so on.
8. **FIFO status bits.** Both transmit and receive FIFOs have status bits TXFFST or RXFFST that define the number of words available in the FIFOs at any time. The transmit FIFO reset bit (TXFIFO) and receive reset bit (RXFIFO) reset the FIFO pointers to zero when these bits are set to 1. The FIFOs resume operation from start once these bits are cleared to zero.
9. **Programmable interrupt levels.** Both transmit and receive FIFOs can generate CPU interrupts and DMA triggers. The transmit interrupt (SPITXINT) is generated whenever the transmit FIFO status bits (TXFFST) match (less than or equal to) the interrupt trigger level bits (TXFFIL). The receive interrupt (SPIRXINT) is generated whenever the receive FIFO status bits (RXFFST) match (greater than or equal to) the interrupt trigger level RXFFIL. This provides a programmable interrupt trigger for transmit and receive sections of the SPI. The default value for these trigger level bits is 0x11111 for receive FIFO and 0x00000 for transmit FIFO, respectively.

37.3.8 SPI DMA Transfers

37.3.8.1 Transmitting Data Using SPI with DMA

When using the DMA with the TX FIFO, the DMA Burst Size (DMA_BURST_SIZE) must be no greater than 16 – TXFFIL, to prevent the DMA from writing to an already full FIFO. This leads to data loss and is not recommended.

For complete data transmission, follow these steps:

1. Calculate the total number or words to be transmitted. [NUM_WORDS]
2. Decide the transmit FIFO level. [TXFFIL]
3. Calculate the number of DMA transfers. [DMA_TRANSFER_SIZE]
4. Calculate the size of the DMA Burst. [DMA_BURST_SIZE]
5. Configure DMA using calculated values.
6. Configure SPI with FIFO using the calculated values.

To transfer 128 words to SPI using the DMA:

NUM_WORDS: 128

TXFFIL: 8

DMA_TRANSFER_SIZE: (NUM_WORDS /TXFFIL) – 1 = (128/8) – 1 = 15 (16 transfers)

DMA_BURST_SIZE: (16 – TXFFIL) – 1 = (16 – 8) – 1 = 7 (8 words per burst)

Note

Avoid setting TXFFIL to 0h or 10h to make sure of proper DMA configuration.

37.3.8.2 Receiving Data Using SPI with DMA

When using the DMA with the RX FIFO, the DMA Burst Size (BURST_SIZE) must be no greater than RXFFIL to prevent the DMA from reading from an empty FIFO. To make sure that the DMA correctly receives all data from the RX FIFO, the DMA Burst Size can equal RXFFIL and also be an integer divisor of the total number of SPI transmissions.

For complete data reception, follow these steps:

1. Calculate the number of words to be received. [NUM_WORDS]
2. Calculate the necessary FIFO level [RXFFIL]
3. Calculate the total number of DMA transfers. [DMA_TRANSFER_SIZE]
4. Calculate the size of the DMA Burst. [DMA_BURST_SIZE]
5. Configure DMA using the calculated values.
6. Configure SPI with FIFO using the calculated values.

To receive 200 words from SPI using the DMA:

NUM_WORDS = 200

RXFFIL: 4

DMA_TRANSFER_SIZE: (NUM_WORDS /RXFFIL) – 1 = (200/4) – 1 = 49 (50 transfers)

DMA_BURST_SIZE = RXFFIL-1 = 3 (4 words per burst)

Note

Avoid setting RXFFIL to 0h to make sure proper DMA configuration.

37.3.9 SPI High-Speed Mode

The SPI module is capable of reaching full-duplex communication speeds up to LSPCLK/4 (where LSPCLK equals SYSCLK). For the maximum rated speed, refer the device data manual.

To achieve the maximum full-duplex speeds, the following restrictions are placed on the design:

- Single master to single slave configuration is supported.
- Loading on the pins must not exceed the value stated in the device data manual.

When configuring the GPIOs to support high-speed mode, refer to Section 37.2.2.1 for more information.

37.3.10 SPI 3-Wire Mode Description

SPI 3-wire mode allows for SPI communication over three pins instead of the normal four pins.

In master mode, if the TRIWIRE bit is set, enabling 3-wire SPI mode, SPISIMOx becomes the bi-directional SPIMOMIx (SPI master out, master in) pin, and SPISOMIx is no longer used by the SPI. In slave mode, if the TRIWIRE bit is set, SPISOMIx becomes the bi-directional SPISISOx (SPI slave in, slave out) pin, and SPISIMOx is no longer used by the SPI.

Table 37-4 indicates the pin function differences between 3-wire and 4-wire SPI mode for a master and slave SPI.

Table 37-4. 4-wire versus 3-wire SPI Pin Functions

4-wire SPI	3-wire SPI (Master)	3-wire SPI (Slave)
SPICLKx	SPICLKx	SPICLKx
SPISTEx	SPISTEx	SPISTEx
SPISIMOx	SPIMOMIx	Free
SPISOMIx	Free	SPISISOx

Because in 3-wire mode, the receive and transmit paths within the SPI are connected, any data transmitted by the SPI module is also received by itself. The application software must take care to perform a dummy read to clear the SPI data register of the additional received data.

The TALK bit plays an important role in 3-wire SPI mode. The bit must be set to transmit data and cleared prior to reading data. In master mode, in order to initiate a read, the application software must write dummy data to the SPI data register (SPIDAT or SPIRXBUF) while the TALK bit is cleared (no data is transmitted out the SPIMOMI pin) before reading from the data register.

Figure 37-9 and Figure 37-10 illustrate 3-wire master and slave mode.

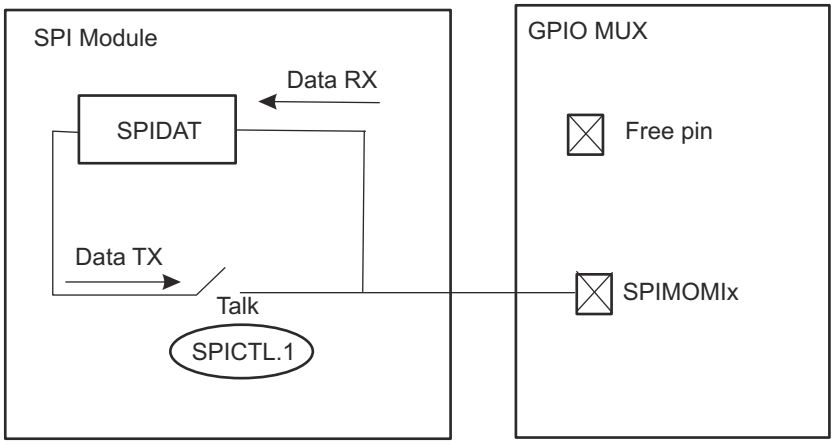


Figure 37-9. SPI 3-wire Master Mode

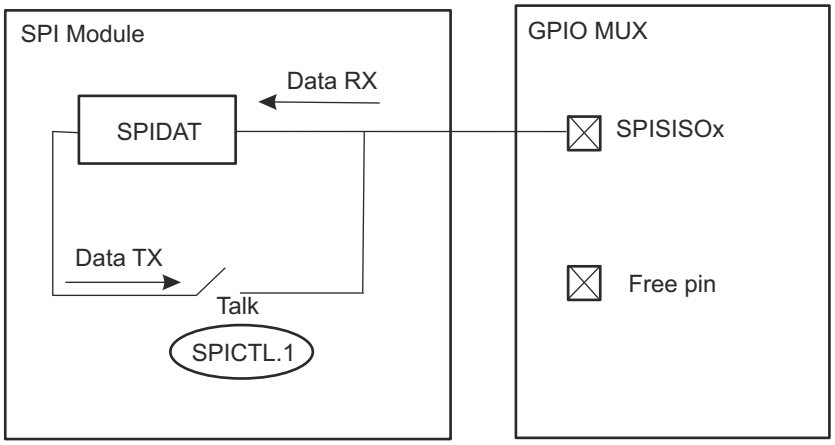


Figure 37-10. SPI 3-wire Slave Mode

Table 37-5 indicates how data is received or transmitted in the various SPI modes while the TALK bit is set or cleared.

Table 37-5. 3-Wire SPI Pin Configuration

Pin Mode	SPIPRI[TRIWIRE]	SPICTL[TALK]	SPISIMO	SPISOMI
		Master Mode		
4-wire	0	X	TX	RX
3-pin mode	1	0	RX	Disconnect from SPI
		1	TX/RX	
		Slave Mode		
4-wire	0	X	RX	TX
3-pin mode	1	0	Disconnect from SPI	RX
		1		TX/RX

37.4 Programming Procedure

This section describes the procedure for configuring the SPI for the various modes of operation.

37.4.1 Initialization Upon Reset

A system reset forces the SPI peripheral into the following default configuration:

- Unit is configured as a slave module (MASTER_SLAVE = 0)
- Transmit capability is disabled (TALK = 0)
- Data is latched at the input on the falling edge of the SPICLK signal
- Character length is assumed to be one bit
- SPI interrupts are disabled
- Data in SPIDAT is reset to 0000h

37.4.2 Configuring the SPI

This section describes the procedure in which to configure the SPI module for operation. To prevent unwanted and unforeseen events from occurring during or as a result of initialization changes, clear the SPISWRESET bit before making initialization changes, and then set this bit after initialization is complete. While the SPI is held in reset (SPISWRESET = 0), configuration can be changed in any order. The following list shows the SPI configuration procedure in a logical order. However, the SPI registers can be written with single 16-bit writes, so the order is not required with the exception of SPISWRESET.

Note

Do not change the SPI configuration when communication is in progress.

To change the SPI configuration:

1. Clear the SPI Software Reset bit (SPISWRESET) to 0 to force the SPI to the reset state.
2. Configure the SPI as desired:
   - Select either master or slave mode (MASTER_SLAVE).
   - Choose SPICLK polarity and phase (CLKPOLARITY and CLK_PHASE).
   - Set the desired baud rate (SPIBRR).
   - Enable high speed mode, if desired (HS_MODE).
   - Set the SPI character length (SPICHAR).
   - Clear the SPI Flags (OVERRUN_FLAG, INT_FLAG).
   - Enable $\overline{\text{SPISTE}}$ inversion (STEINV), if needed.
   - Enable 3-wire mode (TRIWIRE), if needed.
   - If using FIFO enhancements:
     - Enable the FIFO enhancements (SPIFFENA).
     - Clear the FIFO Flags (TXFFINTCLR, RXFFOVFCLR, and RXFFINTCLR).
     - Release transmit and receive FIFO resets (TXFIFO and RXFIFORESET).
     - Release SPI FIFO channels from reset (SPIRST).
3. If interrupts are used:
   - In non-FIFO mode, enable the receiver overrun and/or SPI interrupts (OVERRUNINTENA and SPIINTENA).
   - In FIFO mode, set the transmit and receive interrupt levels (TXFFIL and RXFFIL) then enable the interrupts (TXFFIENA and RXFFIENA).
4. Set SPISWRESET to 1 to release the SPI from the reset state.

37.4.3 Configuring the SPI for High-Speed Mode

To achieve the maximum rated speeds, the following settings must be made. This example assumes that the device is operating at 100 MHz.

Set LSPCLK equal to SYSCLK:

```
ClkCfgRegs.LOSPCP.bit.LSPCLKDIV = 0;
```

Select the appropriate Pin Mux options in GPIO_CTRL_REGS.

During the SPI configuration procedure:

Set HS_MODE to 1.

```
SpiaRegs.SPICCR.bit.HS_MODE = 0x1;
```

Set SPIBRR to 3. SPICLK = LSPCLK/(SPIBRR+1) = 25- MHz

```
SpiaRegs.SPIBRR = 0x3;
```

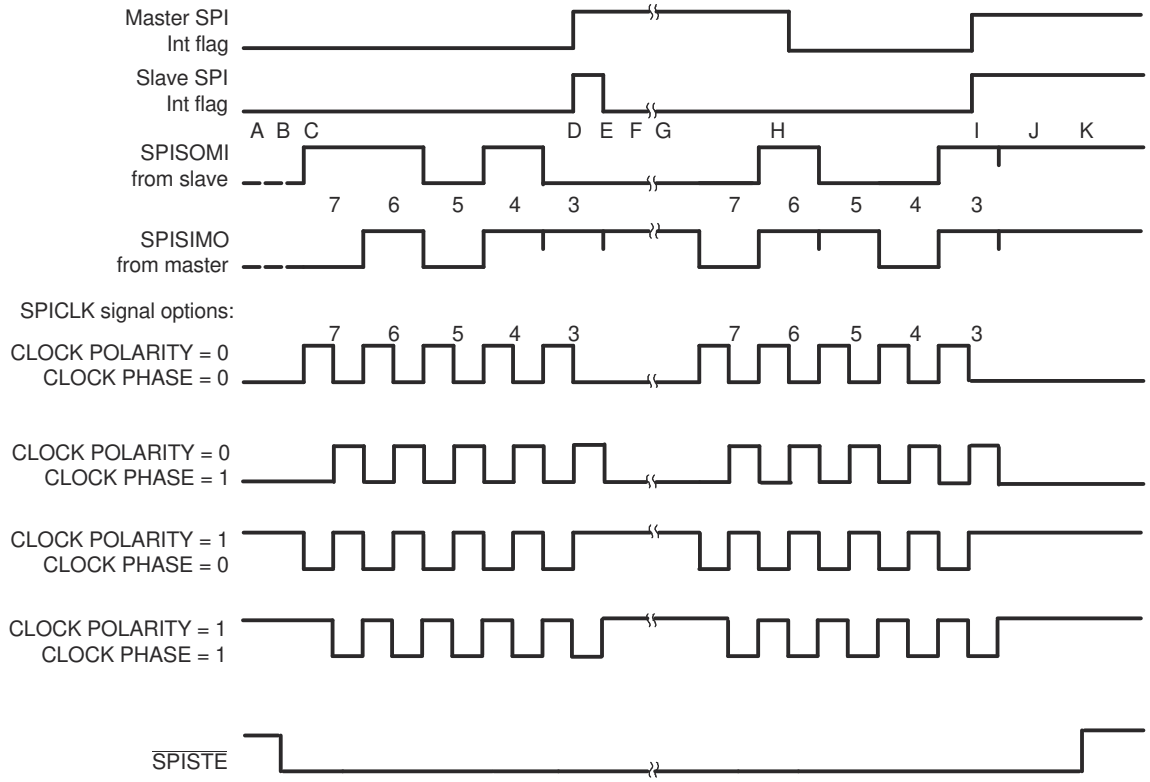
There are no other differences in the configuration from normal SPI operation. Sending and receiving data, DMA operation, and interrupts operates without change.

37.4.4 Data Transfer Example

The timing diagram shown in Figure 37-11 shows an SPI data transfer between two devices using a character length of five bits with the SPICLK being symmetrical.

The timing diagram with SPICLK asymmetrical (Figure 37-8) shares similar characterizations with Figure 37-11 except that the data transfer is one LSPCLK cycle longer per bit during the low pulse (CLKPOLARITY = 0) or during the high pulse (CLKPOLARITY = 1) of the SPICLK.

Figure 37-11 is applicable for 8-bit SPI only and is not for C28x devices that are capable of working with 16-bit data. The figure is shown for illustrative purposes only.



A. Slave writes 0D0h to SPIDAT and waits for the master to shift out the data.
B. Master sets the slave $\overline{\text{SPISTE}}$ signal low (active).
C. Master writes 058h to SPIDAT, which starts the transmission procedure.
D. First byte is finished and sets the interrupt flags.
E. Slave reads 0Bh from its SPIRXBUF (right-justified).
F. Slave writes 04Ch to SPIDAT and waits for the master to shift out the data.
G. Master writes 06Ch to SPIDAT, which starts the transmission procedure.
H. Master reads 01Ah from the SPIRXBUF (right-justified).
I. Second byte is finished and sets the interrupt flags.
J. Master reads 89h and the slave reads 8Dh from their respective SPIRXBUF. After the user software masks off the unused bits, the master receives 09h and the slave receives 0Dh.
K. Master clears the slave $\overline{\text{SPISTE}}$ signal high (inactive).

Figure 37-11. Five Bits per Character

37.4.5 SPI 3-Wire Mode Code Examples

In addition to the normal SPI initialization, to configure the SPI module for 3-wire mode, the TRIWIRE bit (SPIPRI.0) must be set to 1. After initialization, there are several considerations to take into account when transmitting and receiving data in 3-wire master and slave mode. The following examples demonstrate these considerations.

In 3-wire master mode, SPICLKx, $\overline{\text{SPISTE}}$x, and SPISIMOx pins must be configured as SPI pins (SPISOMIx pin can be configured as non-SPI pin). When the master transmits, the master receives the data the master transmits (because SPISIMOx and SPISOMIx are connected internally in 3-wire mode). Therefore, the junk data received must be cleared from the receive buffer every time data is transmitted.

Example 37-4. 3-Wire Master Mode Transmit

```
Uint16 data;
Uint16 dummy;
     SpiaRegs.SPICTL.bit.TALK = 1;              // Enable Transmit path
     SpiaRegs.SPITXBUF = data; // Master transmits data
     while(SpiaRegs.SPISTS.bit.INT_FLAG !=1) {} // Waits until data rx'd
     dummy = SpiaRegs.SPIRXBUF;                 // Clears junk data from itself
                                                // bc it rx'd same data tx'd
```

To receive data in 3-wire master mode, the master must clear the TALK (SPICTL.1) bit to 0 to close the transmit path and then transmit dummy data in order to initiate the transfer from the slave. Because the TALK bit is 0, unlike in transmit mode, the master dummy data does not appear on the SPISIMOx pin, and the master does not receive its own dummy data. Instead, the data from the slave is received by the master.

Example 37-5. 3-Wire Master Mode Receive

```
Uint16 rdata;
Uint16 dummy;
     SpiaRegs.SPICTL.bit.TALK = 0;              // Disable Transmit path
     SpiaRegs.SPITXBUF = dummy;                 // Send dummy to start tx
     // NOTE: because TALK = 0, data does not tx onto SPISIMOA pin
     while(SpiaRegs.SPISTS.bit.INT_FLAG !=1) {} // Wait until data received
     rdata = SpiaRegs.SPIRXBUF;                 // Master reads data
```

In 3-wire slave mode, SPICLKx, SPISTEx, and SPISOMIx pins must be configured as SPI pins (SPISIMOx pin can be configured as non-SPI pin). Like in master mode, when transmitting, the slave receives the data it transmits and must clear this junk data from its receive buffer.

Example 37-6. 3-Wire Slave Mode Transmit

```
Uint16 data;
Uint16 dummy;
     SpiaRegs.SPICTL.bit.TALK = 1;              // Enable Transmit path
     SpiaRegs.SPITXBUF = data;                  // Slave transmits data
     while(SpiaRegs.SPISTS.bit.INT_FLAG !=1) {} // Wait until data rx'd
     dummy = SpiaRegs.SPIRXBUF;                 // Clears junk data from itself
```

As in 3-wire master mode, the TALK bit must be cleared to 0. Otherwise, the slave receives data normally.

Example 37-7. 3-Wire Slave Mode Receive

```
Uint16 rdata;
    SpiaRegs.SPICTL.bit.TALK = 0;              // Disable Transmit path
    while(SpiaRegs.SPISTS.bit.INT_FLAG !=1) {} // Waits until data rx'd
    rdata = SpiaRegs.SPIRXBUF;                 // Slave reads data
```

37.4.6 SPI STEINV Bit in Digital Audio Transfers

On those devices with two SPI modules, enabling the STEINV bit on one of the SPI modules allows the pair of SPIs to receive both left and right-channel digital audio data in slave mode. The SPI module that receives a normal active-low $\overline{\text{SPISTE}}$ signal stores right-channel data, and the SPI module that receives an inverted active-high $\overline{\text{SPISTE}}$ signal stores left-channel data from the master. To receive digital audio data from a digital audio interface receiver, the SPI modules can be connected as shown in Figure 37-12.

Note

This configuration is only applicable to slave mode (MASTER_SLAVE = 0). When the SPI is configured as master (MASTER_SLAVE = 1), the STEINV bit has no effect on the $\overline{\text{SPISTE}}$ pin.

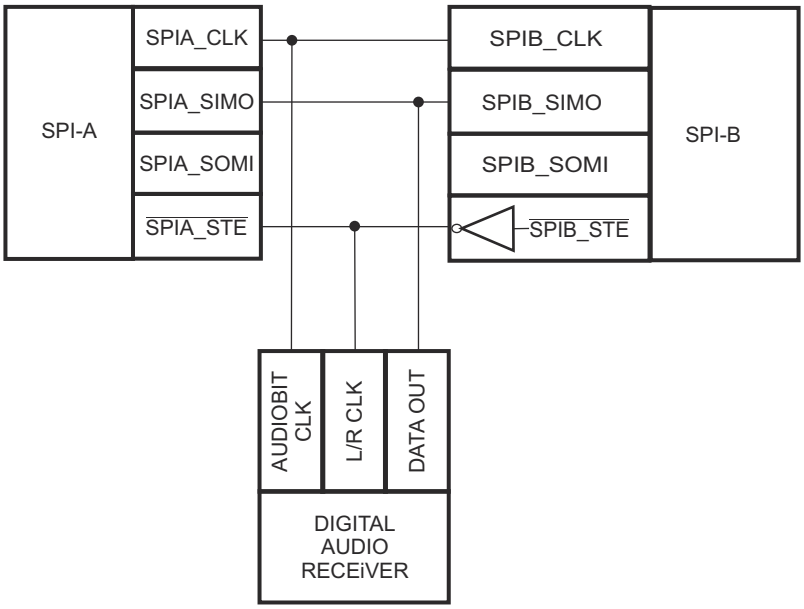


Figure 37-12. SPI Digital Audio Receiver Configuration Using Two SPIs

Standard C28x SPI timing requirements limit the number of digital audio interface formats supported using the 2-SPI configuration with the STEINV bit. See the device data manual electrical specifications for SPI timing requirements. With the SPI clock phase configured such that the CLKPOLARITY bit is 0 and the CLK_PHASE bit is 1 (data latched on rising edge of clock), standard right-justified digital audio interface data format is supported as shown in Figure 37-13.

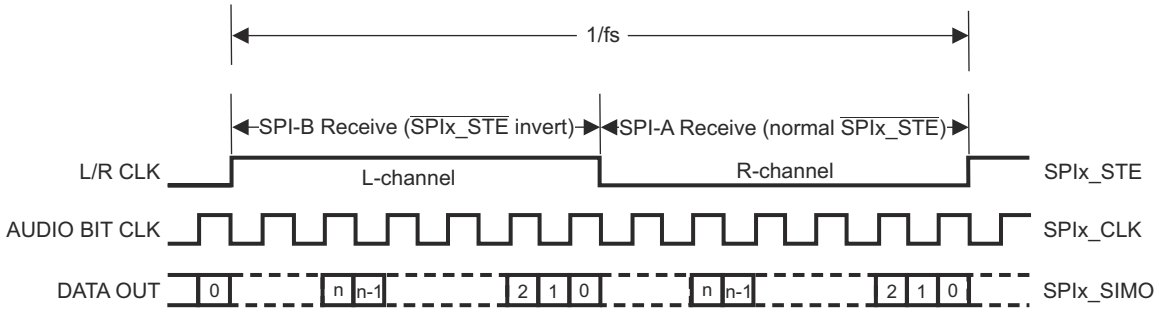


Figure 37-13. Standard Right-Justified Digital Audio Data Format

37.5 Software

37.5.1 SPI Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/spi

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

37.5.1.1 SPI Digital Loopback

FILE: spi_ex1_loopback.c

This program uses the internal loopback test mode of the SPI module. This is a very basic loopback that does not use the FIFOs or interrupts. A stream of data is sent and then compared to the received stream. The pinmux and SPI modules are configure through the sysconfig file.

The sent data looks like this:
0000 0001 0002 0003 0004 0005 0006 0007 .... FFFE FFFF 0000

This pattern is repeated forever.

External Connections
- None

Watch Variables
- *sData* - Data to send
- *rData* - Received data

37.5.1.2 SPI Digital Loopback with FIFO Interrupts

FILE: spi_ex2_loopback_fifo_interrupts.c

This program uses the internal loopback test mode of the SPI module. Both the SPI FIFOs and their interrupts are used.

A stream of data is sent and then compared to the received stream. The sent data looks like this:
0000 0001
0001 0002
0002 0003
....
FFFE FFFF
FFFF 0000
etc..
This pattern is repeated forever.

External Connections
- None

Watch Variables
- *sData* - Data to send
- *rData* - Received data
- *rDataPoint* - Used to keep track of the last position in the receive stream for error checking

37.5.1.3 SPI Digital External Loopback without FIFO Interrupts

FILE: spi_ex3_external_loopback.c

This program uses the external loopback between two SPI modules. Both the SPI FIFOs and interrupts are not used in this example. SPIA is configured as a peripheral and SPI B is configured as controller. This example demonstrates full duplex communication where both controller and peripheral transmits and receives data simultaneously.

External Connections

- GPIO24 and GPIO16 - SPIPICO
- GPIO25 and GPIO17 - SPIPOCI
- GPIO26 and GPIO18 - SPICLK
- GPIO27 and GPIO19 - SPIPTE

Watch Variables

- *TxData_SPIA* - Data send from SPIA (peripheral)
- *TxData_SPIB* - Data send from SPIB (controller)
- *RxData_SPIA* - Data received by SPIA (peripheral)
- *RxData_SPIB* - Data received by SPIB (controller)

37.5.1.4 SPI Digital External Loopback with FIFO Interrupts

FILE: spi_ex4_external_loopback_fifo_interrupts.c

This program uses the external loopback between two SPI modules. Both the SPI FIFOs and their interrupts are used. SPIA is configured as a peripheral and receives data from SPI B which is configured as a controller.

A stream of data is sent and then compared to the received stream. The sent data looks like this:
0000 0001
0001 0002
0002 0003
....
FFFE FFFF
FFFF 0000
etc..
This pattern is repeated forever.

External Connections

- GPIO24 and GPIO16 - SPIPICO
- GPIO25 and GPIO17 - SPIPOCI
- GPIO26 and GPIO18 - SPICLK
- GPIO27 and GPIO19 - SPIPTE

Watch Variables

- *sData* - Data to send
- *rData* - Received data
- *rDataPoint* - Used to keep track of the last position in the receive stream for error checking

37.5.1.5 SPI Digital Loopback with DMA

FILE: spi_ex5_loopback_dma.c

This program uses the internal loopback test mode of the SPI module. Both DMA interrupts and the SPI FIFOs are used. When the SPI transmit FIFO has enough space (as indicated by its FIFO level interrupt signal), the DMA will transfer data from global variable sData into the FIFO. This will be transmitted to the receive FIFO via the internal loopback.

When enough data has been placed in the receive FIFO (as indicated by its FIFO level interrupt signal), the DMA will transfer the data from the FIFO into global variable rData.

When all data has been placed into rData, a check of the validity of the data will be performed in one of the DMA channels' ISRs.

External Connections

- None

Watch Variables

- *sData* - Data to send
- *rData* - Received data

37.5.1.6 SPI EEPROM

FILE: spi_ex6_eeprom.c

This program will write 8 bytes to EEPROM and read them back. The device communicates with the EEPROM via SPI and specific opcodes. This example is written to work with the SPI Serial EEPROM AT25128/256.

External Connections
ExternalConnections
- Connect external SPI EEPROM
- Connect GPIO16 (PICO) to external EEPROM SI pin
- Connect GPIO17 (POCI) to external EEPROM SO pin
- Connect GPIO18 (CLK) to external EEPROM SCK pin
- Connect GPIO11 (CS) to external EEPROM CS pin
- Connect the external EEPROM VCC and GND pins

Watch Variables
- writeBuffer - Data that is written to external EEPROM
- readBuffer - Data that is read back from EEPROM
- error - Error count

37.5.1.7 SPI DMA EEPROM

FILE: spi_ex7_eeprom_dma.c

This program will write 8 bytes to EEPROM and read them back. The device communicates with the EEPROM via SPI using DMA and specific opcodes. This example is written to work with the SPI Serial EEPROM AT25128/256.

External Connections
ExternalConnections
- Connect external SPI EEPROM
- Connect GPIO16 (PICO) to external EEPROM SI pin
- Connect GPIO17 (POCI) to external EEPROM SO pin
- Connect GPIO18 (CLK) to external EEPROM SCK pin
- Connect GPIO11 (CS) to external EEPROM CS pin
- Connect the external EEPROM VCC and GND pins

Watch Variables
- writeBuffer - Data that is written to external EEPROM
- SPI_DMA_Handle.RXdata - Data that is read back from EEPROM when number of received bytes is less than 4
- SPI_DMA_Handle.pSPIRXDMA->pbuffer - Start address of received data from EEPROM
- error - Error count

37.6 SPI Registers

This section describes the Serial Peripheral Interface registers. It is important to note that the SPI registers only allow 16-bit accesses.

37.6.1 SPI Base Address Table (C28)

Table 37-6. SPI Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
SpiaRegs	SPI_REGS	SPIA_BASE	0x0000_6100	YES	YES	YES	YES	YES
SpibRegs	SPI_REGS	SPIB_BASE	0x0000_6110	YES	YES	YES	YES	YES
SpicRegs	SPI_REGS	SPIC_BASE	0x0000_6120	YES	YES	YES	YES	YES
SpidRegs	SPI_REGS	SPID_BASE	0x0000_6130	YES	YES	YES	YES	YES

37.6.2 SPI_REGS Registers

Table 37-7 lists the memory-mapped registers for the SPI_REGS registers. All register offset addresses not listed in Table 37-7 should be considered as reserved locations and the register contents should not be modified.

Table 37-7. SPI_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	SPICCR	SPI Configuration Control Register		Go
1h	SPICTL	SPI Operation Control Register		Go
2h	SPISTS	SPI Status Register		Go
4h	SPIBRR	SPI Baud Rate Register		Go
6h	SPIRXEMU	SPI Emulation Buffer Register		Go
7h	SPIRXBUF	SPI Serial Input Buffer Register		Go
8h	SPITXBUF	SPI Serial Output Buffer Register		Go
9h	SPIDAT	SPI Serial Data Register		Go
Ah	SPIFFTX	SPI FIFO Transmit Register		Go
Bh	SPIFFRX	SPI FIFO Receive Register		Go
Ch	SPIFFCT	SPI FIFO Control Register		Go
Fh	SPIPRI	SPI Priority Control Register		Go

Complex bit access types are encoded to fit into small table cells. Table 37-8 shows the codes that are used for access types in this section.

Table 37-8. SPI_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
RC	R C	Read to Clear
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

37.6.2.1 SPICCR Register (Offset = 0h) [Reset = 0h]

SPICCR is shown in Figure 37-14 and described in Table 37-9.

Return to the Summary Table.

SPICCR controls the setup of the SPI for operation.

Figure 37-14. SPICCR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
SPISWRESET	CLKPOLARITY	HS_MODE	SPILBK	SPICHAR			
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h			

Table 37-9. SPICCR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	SPISWRESET	R/W	0h	SPI Software Reset When changing configuration, you should clear this bit before the changes and set this bit before resuming operation. Reset type: SYSRSn 0h (R/W) = Initializes the SPI operating flags to the reset condition. Specifically, the RECEIVER OVERRUN Flag bit (SPISTS.7), the SPI INT FLAG bit (SPISTS.6), and the TXBUF FULL Flag bit (SPISTS.5) are cleared. SPISTE will become inactive. SPICLK will be immediately driven to 0 regardless of the clock polarity. The SPI configuration remains unchanged. 1h (R/W) = SPI is ready to transmit or receive the next character. When the SPI SW RESET bit is a 0, a character written to the transmitter will not be shifted out when this bit is set. A new character must be written to the serial data register. SPICLK will be returned to its inactive state one SPICLK cycle after this bit is set.
6	CLKPOLARITY	R/W	0h	Shift Clock Polarity This bit controls the polarity of the SPICLK signal. CLOCK POLARITY and POLARITY CLOCK PHASE (SPICTL.3) control four clocking schemes on the SPICLK pin. Reset type: SYSRSn 0h (R/W) = Data is output on rising edge and input on falling edge. When no SPI data is sent, SPICLK is at low level. The data input and output edges depend on the value of the CLOCK PHASE bit (SPICTL.3) as follows: - CLOCK PHASE = 0: Data is output on the rising edge of the SPICLK signal. Input data is latched on the falling edge of the SPICLK signal. - CLOCK PHASE = 1: Data is output one half-cycle before the first rising edge of the SPICLK signal and on subsequent falling edges of the SPICLK signal. Input data is latched on the rising edge of the SPICLK signal. 1h (R/W) = Data is output on falling edge and input on rising edge. When no SPI data is sent, SPICLK is at high level. The data input and output edges depend on the value of the CLOCK PHASE bit (SPICTL.3) as follows: - CLOCK PHASE = 0: Data is output on the falling edge of the SPICLK signal. Input data is latched on the rising edge of the SPICLK signal. - CLOCK PHASE = 1: Data is output one half-cycle before the first falling edge of the SPICLK signal and on subsequent rising edges of the SPICLK signal. Input data is latched on the falling edge of the SPICLK signal.

Table 37-9. SPICCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	HS_MODE	R/W	0h	High Speed Mode Enable Bits This bit determines if the High Speed mode is enabled. The correct GPIOs should be selected in the GPxGMUX/GPxMUX registers. Reset type: SYSRSn 0h (R/W) = SPI High Speed mode disabled. This is the default value after reset. 1h (R/W) = SPI High Speed mode enabled,
4	SPILBK	R/W	0h	SPI Loopback Mode Select Loopback mode allows module validation during device testing. This mode is valid only in master mode of the SPI. Reset type: SYSRSn 0h (R/W) = SPI loopback mode disabled. This is the default value after reset. 1h (R/W) = SPI loopback mode enabled, SIMO/SOMI lines are connected internally. Used for module self-tests.
3-0	SPICHAR	R/W	0h	Character Length Control Bits These four bits determine the number of bits to be shifted in or SPI CHAR0 out as a single character during one shift sequence. SPICHAR = Word length - 1 Reset type: SYSRSn 0h (R/W) = 1-bit word 1h (R/W) = 2-bit word 7h (R/W) = 8-bit word Fh (R/W) = 16-bit word

37.6.2.2 SPICTL Register (Offset = 1h) [Reset = 0h]

SPICTL is shown in Figure 37-15 and described in Table 37-10.

Return to the Summary Table.

SPICTL controls data transmission, the SPI's ability to generate interrupts, the SPICLK phase, and the operational mode (slave or master).

Figure 37-15. SPICTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			OVERRUNINTENA	CLK_PHASE	MASTER_SLAVE	TALK	SPIINTENA
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 37-10. SPICTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4	OVERRUNINTENA	R/W	0h	Overrun Interrupt Enable Overrun Interrupt Enable. Setting this bit causes an interrupt to be generated when the RECEIVER OVERRUN Flag bit (SPISTS.7) is set by hardware. Interrupts generated by the RECEIVER OVERRUN Flag bit and the SPI INT FLAG bit (SPISTS.6) share the same interrupt vector. Reset type: SYSRSn 0h (R/W) = Disable RECEIVER OVERRUN interrupts. 1h (R/W) = Enable RECEIVER_OVERRUN interrupts.
3	CLK_PHASE	R/W	0h	SPI Clock Phase Select This bit controls the phase of the SPICLK signal. CLOCK PHASE and CLOCK POLARITY (SPICCR.6) make four different clocking schemes possible (see clocking figures in SPI chapter). When operating with CLOCK PHASE high, the SPI (master or slave) makes the first bit of data available after SPIDAT is written and before the first edge of the SPICLK signal, regardless of which SPI mode is being used. Reset type: SYSRSn 0h (R/W) = Normal SPI clocking scheme, depending on the CLOCK POLARITY bit (SPICCR.6). 1h (R/W) = SPICLK signal delayed by one half-cycle. Polarity determined by the CLOCK POLARITY bit.
2	MASTER_SLAVE	R/W	0h	SPI Network Mode Control This bit determines whether the SPI is a network master or slave. SLAVE During reset initialization, the SPI is automatically configured as a network slave. Reset type: SYSRSn 0h (R/W) = SPI is configured as a slave. 1h (R/W) = SPI is configured as a master.

Table 37-10. SPICTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	TALK	R/W	0h	Transmit Enable The TALK bit can disable data transmission (master or slave) by placing the serial data output in the high-impedance state. If this bit is disabled during a transmission, the transmit shift register continues to operate until the previous character is shifted out. When the TALK bit is disabled, the SPI is still able to receive characters and update the status flags. TALK is cleared (disabled) by a system reset. Reset type: SYSRSn 0h (R/W) = Disables transmission: - Slave mode operation: If not previously configured as a general-purpose I/O pin, the SPISOMI pin will be put in the high-impedance state. - Master mode operation: If not previously configured as a general-purpose I/O pin, the SPISIMO pin will be put in the high-impedance state. 1h (R/W) = Enables transmission For the 4-pin option, ensure to enable the receiver's SPISTEn input pin.
0	SPIINTENA	R/W	0h	SPI Interrupt Enable This bit controls the SPI's ability to generate a transmit/receive interrupt. The SPI INT FLAG bit (SPISTS.6) is unaffected by this bit. Reset type: SYSRSn 0h (R/W) = Disables the interrupt. 1h (R/W) = Enables the interrupt.

37.6.2.3 SPISTS Register (Offset = 2h) [Reset = 0h]

SPISTS is shown in Figure 37-16 and described in Table 37-11.

Return to the Summary Table.

SPISTS contains interrupt and status bits.

Figure 37-16. SPISTS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
OVERRUN_FLAG	INT_FLAG	BUFFULL_FLAG	RESERVED				
W1C-0h	RC-0h	R-0h	R-0h				

Table 37-11. SPISTS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	OVERRUN_FLAG	W1C	0h	SPI Receiver Overrun Flag This bit is a read/clear-only flag. The SPI hardware sets this bit when a receive or transmit operation completes before the previous character has been read from the buffer. The bit is cleared in one of three ways: - Writing a 1 to this bit - Writing a 0 to SPI SW RESET (SPICCR.7) - Resetting the system If the OVERRUN INT ENA bit (SPICTL.4) is set, the SPI requests only one interrupt upon the first occurrence of setting the RECEIVER OVERRUN Flag bit. Subsequent overruns will not request additional interrupts if this flag bit is already set. This means that in order to allow new overrun interrupt requests the user must clear this flag bit by writing a 1 to SPISTS.7 each time an overrun condition occurs. In other words, if the RECEIVER OVERRUN Flag bit is left set (not cleared) by the interrupt service routine, another overrun interrupt will not be immediately re-entered when the interrupt service routine is exited. Reset type: SYSRSn 0h (R/W) = A receive overrun condition has not occurred. 1h (R/W) = The last received character has been overwritten and therefore lost (when the SPIRXBUF was overwritten by the SPI module before the previous character was read by the user application). Writing a '1' will clear this bit. The RECEIVER OVERRUN Flag bit should be cleared during the interrupt service routine because the RECEIVER OVERRUN Flag bit and SPI INT FLAG bit (SPISTS.6) share the same interrupt vector. This will alleviate any possible doubt as to the source of the interrupt when the next byte is received.

Table 37-11. SPISTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	INT_FLAG	RC	0h	SPI Interrupt Flag SPI INT FLAG is a read-only flag. Hardware sets this bit to indicate that the SPI has completed sending or receiving the last bit and is ready to be serviced. This flag causes an interrupt to be requested if the SPI INT ENA bit (SPICTL.0) is set. The received character is placed in the receiver buffer at the same time this bit is set. This bit is cleared in one of three ways: - Reading SPIRXBUF - Writing a 0 to SPI SW RESET (SPICCR.7) - Resetting the system Note: This bit should not be used if FIFO mode is enabled. The internal process of copying the received word from SPIRXBUF to the Receive FIFO will clear this bit. Use the FIFO status, or FIFO interrupt bits for similar functionality. Reset type: SYSRSn 0h (R/W) = No full words have been received or transmitted. 1h (R/W) = Indicates that the SPI has completed sending or receiving the last bit and is ready to be serviced.
5	BUFFULL_FLAG	R	0h	SPI Transmit Buffer Full Flag This read-only bit gets set to 1 when a character is written to the SPI Transmit buffer SPITXBUF. It is cleared when the character is automatically loaded into SPIDAT when the shifting out of a previous character is complete. Reset type: SYSRSn 0h (R/W) = Transmit buffer is not full. 1h (R/W) = Transmit buffer is full.
4-0	RESERVED	R	0h	Reserved

37.6.2.4 SPIBRR Register (Offset = 4h) [Reset = 0h]

SPIBRR is shown in Figure 37-17 and described in Table 37-12.

Return to the Summary Table.

SPIBRR contains the bits used for baud-rate selection.

Figure 37-17. SPIBRR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	SPI_BIT_RATE						
R-0h	R/W-0h						

Table 37-12. SPIBRR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6-0	SPI_BIT_RATE	R/W	0h	SPI Baud Rate Control These bits determine the bit transfer rate if the SPI is the network SPI BIT RATE 0 master. There are 125 data-transfer rates (each a function of the CPU clock, LSPCLK) that can be selected. One data bit is shifted per SPICLK cycle. (SPICLK is the baud rate clock output on the SPICLK pin.) If the SPI is a network slave, the module receives a clock on the SPICLK pin from the network master. Therefore, these bits have no effect on the SPICLK signal. The frequency of the input clock from the master should not exceed the slave SPI's LSPCLK signal divided by 4. In master mode, the SPI clock is generated by the SPI and is output on the SPICLK pin. The SPI baud rates are determined by the following formula: For SPIBRR = 3 to 127: SPI Baud Rate = LSPCLK / (SPIBRR + 1) For SPIBRR = 0, 1, or 2: SPI Baud Rate = LSPCLK / 4 Reset type: SYSRSn 3h (R/W) = SPI Baud Rate = LSPCLK/4 4h (R/W) = SPI Baud Rate = LSPCLK/5 7Eh (R/W) = SPI Baud Rate = LSPCLK/127 7Fh (R/W) = SPI Baud Rate = LSPCLK/128

37.6.2.5 SPIRXEMU Register (Offset = 6h) [Reset = 0h]

SPIRXEMU is shown in Figure 37-18 and described in Table 37-13.

Return to the Summary Table.

SPIRXEMU contains the received data. Reading SPIRXEMU does not clear the SPI INT FLAG bit of SPISTS. This is not a real register but a dummy address from which the contents of SPIRXBUF can be read by the emulator without clearing the SPI INT FLAG.

Figure 37-18. SPIRXEMU Register

15	14	13	12	11	10	9	8
ERXBn							
R-0h							
7	6	5	4	3	2	1	0
ERXBn							
R-0h							

Table 37-13. SPIRXEMU Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	ERXBn	R	0h	Emulation Buffer Received Data SPIRXEMU functions almost identically to SPIRXBUF, except that reading SPIRXEMU does not clear the SPI INT FLAG bit (SPISTS.6). Once the SPIDAT has received the complete character, the character is transferred to SPIRXEMU and SPIRXBUF, where it can be read. At the same time, SPI INT FLAG is set. This mirror register was created to support emulation. Reading SPIRXBUF clears the SPI INT FLAG bit (SPISTS.6). In the normal operation of the emulator, the control registers are read to continually update the contents of these registers on the display screen. SPIRXEMU was created so that the emulator can read this register and properly update the contents on the display screen. Reading SPIRXEMU does not clear the SPI INT FLAG bit, but reading SPIRXBUF clears this flag. In other words, SPIRXEMU enables the emulator to emulate the true operation of the SPI more accurately. It is recommended that you view SPIRXEMU in the normal emulator run mode. Reset type: SYSRSn

37.6.2.6 SPIRXBUF Register (Offset = 7h) [Reset = 0h]

SPIRXBUF is shown in Figure 37-19 and described in Table 37-14.

Return to the Summary Table.

SPIRXBUF contains the received data. Reading SPIRXBUF clears the SPI INT FLAG bit in SPISTS. If FIFO mode is enabled, reading this register will also decrement the RXFFST counter in SPIFFRX.

Figure 37-19. SPIRXBUF Register

15	14	13	12	11	10	9	8
RXBn							
R-0h							
7	6	5	4	3	2	1	0
RXBn							
R-0h							

Table 37-14. SPIRXBUF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RXBn	R	0h	Received Data Once SPIDAT has received the complete character, the character is transferred to SPIRXBUF, where it can be read. At the same time, the SPI INT FLAG bit (SPISTS.6) is set. Since data is shifted into the SPI's most significant bit first, it is stored right-justified in this register. Reset type: SYSRSn

37.6.2.7 SPITXBUF Register (Offset = 8h) [Reset = 0h]

SPITXBUF is shown in Figure 37-20 and described in Table 37-15.

Return to the Summary Table.

SPITXBUF stores the next character to be transmitted. Writing to this register sets the TX BUF FULL Flag bit in SPISTS. When the transmission of the current character is complete, the contents of this register are automatically loaded in SPIDAT and the TX BUF FULL Flag is cleared. If no transmission is currently active, data written to this register falls through into the SPIDAT register and the TX BUF FULL Flag is not set.
In master mode, if no transmission is currently active, writing to this register initiates a transmission in the same manner that writing to SPIDAT does.

Figure 37-20. SPITXBUF Register

15	14	13	12	11	10	9	8
TXBn							
R/W-0h							
7	6	5	4	3	2	1	0
TXBn							
R/W-0h							

Table 37-15. SPITXBUF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TXBn	R/W	0h	Transmit Data Buffer This is where the next character to be transmitted is stored. When the transmission of the current character has completed, if the TX BUF FULL Flag bit is set, the contents of this register is automatically transferred to SPIDAT, and the TX BUF FULL Flag is cleared. Writes to SPITXBUF must be left-justified. Reset type: SYSRSn

37.6.2.8 SPIDAT Register (Offset = 9h) [Reset = 0h]

SPIDAT is shown in Figure 37-21 and described in Table 37-16.

Return to the Summary Table.

SPIDAT is the transmit and receive shift register. Data written to SPIDAT is shifted out (MSB) on subsequent SPICLK cycles. For every bit (MSB) shifted out of the SPI, a bit is shifted into the LSB end of the shift register.

Figure 37-21. SPIDAT Register

15	14	13	12	11	10	9	8
SDATn							
R/W-0h							
7	6	5	4	3	2	1	0
SDATn							
R/W-0h							

Table 37-16. SPIDAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SDATn	R/W	0h	Serial Data Shift Register - It provides data to be output on the serial output pin if the TALK bit (SPICTL.1) is set. - When the SPI is operating as a master, a data transfer is initiated. When initiating a transfer, check the CLOCK POLARITY bit (SPICCR.6) described in Section 10.2.1.1 and the CLOCK PHASE bit (SPICTL.3) described in Section 10.2.1.2, for the requirements. In master mode, writing dummy data to SPIDAT initiates a receiver sequence. Since the data is not hardware-justified for characters shorter than sixteen bits, transmit data must be written in left-justified form, and received data read in right-justified form. Reset type: SYSRSn

37.6.2.9 SPIFFTX Register (Offset = Ah) [Reset = A000h]

SPIFFTX is shown in Figure 37-22 and described in Table 37-17.

Return to the Summary Table.

SPIFFTX contains both control and status bits related to the output FIFO buffer. This includes FIFO reset control, FIFO interrupt level control, FIFO level status, as well as FIFO interrupt enable and clear bits.

Figure 37-22. SPIFFTX Register

15	14	13	12	11	10	9	8
SPIRST	SPIFFENA	TXFIFO	TXFFST				
R/W-1h	R/W-0h	R/W-1h	R-0h				
7	6	5	4	3	2	1	0
TXFFINT	TXFFINTCLR	TXFFIENA	TXFFIL				
R-0h	W-0h	R/W-0h	R/W-0h				

Table 37-17. SPIFFTX Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SPIRST	R/W	1h	SPI Reset Reset type: SYSRSn 0h (R/W) = Write 0 to reset the SPI transmit and receive channels. The SPI FIFO register configuration bits will be left as is. 1h (R/W) = SPI FIFO can resume transmit or receive. No effect to the SPI registers bits.
14	SPIFFENA	R/W	0h	SPI FIFO Enhancements Enable Reset type: SYSRSn 0h (R/W) = SPI FIFO enhancements are disabled. 1h (R/W) = SPI FIFO enhancements are enabled.
13	TXFIFO	R/W	1h	TX FIFO Reset Reset type: SYSRSn 0h (R/W) = Write 0 to reset the FIFO pointer to zero, and hold in reset. 1h (R/W) = Release transmit FIFO from reset.
12-8	TXFFST	R	0h	Transmit FIFO Status Reset type: SYSRSn 0h (R/W) = Transmit FIFO is empty. 1h (R/W) = Transmit FIFO has 1 word. 2h (R/W) = Transmit FIFO has 2 words. 10h (R/W) = Transmit FIFO has 16 words, which is the maximum. 1Fh (R/W) = Reserved.
7	TXFFINT	R	0h	TX FIFO Interrupt Flag Reset type: SYSRSn 0h (R/W) = TXFIFO interrupt has not occurred, This is a read-only bit. 1h (R/W) = TXFIFO interrupt has occurred, This is a read-only bit.
6	TXFFINTCLR	W	0h	TXFIFO Interrupt Clear Reset type: SYSRSn 0h (R/W) = Write 0 has no effect on TXFIFINT flag bit, Bit reads back a zero. 1h (R/W) = Write 1 to clear SPIFFTX[TXFFINT] flag.
5	TXFFIENA	R/W	0h	TX FIFO Interrupt Enable Reset type: SYSRSn 0h (R/W) = TX FIFO interrupt based on TXFFIL match (less than or equal to) will be disabled. 1h (R/W) = TX FIFO interrupt based on TXFFIL match (less than or equal to) will be enabled.

Table 37-17. SPIFFTX Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	TXFFIL	R/W	0h	Transmit FIFO Interrupt Level Bits Transmit FIFO will generate interrupt when the FIFO status bits (TXFFST4-0) and FIFO level bits (TXFFIL4-0) match (less than or equal to). Reset type: SYSRSn 0h (R/W) = A TX FIFO interrupt request is generated when there are no words remaining in the TX buffer. 1h (R/W) = A TX FIFO interrupt request is generated when there is 1 word or no words remaining in the TX buffer. 2h (R/W) = A TX FIFO interrupt request is generated when there is 2 words or fewer remaining in the TX buffer. 10h (R/W) = A TX FIFO interrupt request is generated when there are 16 words or fewer remaining in the TX buffer. 1Fh (R/W) = Reserved.

37.6.2.10 SPIFFRX Register (Offset = Bh) [Reset = 201Fh]

SPIFFRX is shown in Figure 37-23 and described in Table 37-18.

Return to the Summary Table.

SPIFFRX contains both control and status bits related to the input FIFO buffer. This includes FIFO reset control, FIFO interrupt level control, FIFO level status, as well as FIFO interrupt enable and clear bits.

Figure 37-23. SPIFFRX Register

15	14	13	12	11	10	9	8
RXFFOVF	RXFFOVFCLR	RXFIFORESET	RXFFST				
R-0h	W-0h	R/W-1h	R-0h				
7	6	5	4	3	2	1	0
RXFFINT	RXFFINTCLR	RXFFIENA	RXFFIL				
R-0h	W-0h	R/W-0h	R/W-1Fh				

Table 37-18. SPIFFRX Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RXFFOVF	R	0h	Receive FIFO Overflow Flag Reset type: SYSRSn 0h (R/W) = Receive FIFO has not overflowed. This is a read-only bit. 1h (R/W) = Receive FIFO has overflowed, read-only bit. More than 16 words have been received in to the FIFO, and the first received word is lost.
14	RXFFOVFCLR	W	0h	Receive FIFO Overflow Clear Reset type: SYSRSn 0h (R/W) = Write 0 does not affect RXFFOVF flag bit, Bit reads back a zero. 1h (R/W) = Write 1 to clear SPIFFRX[RXFFOVF].
13	RXFIFORESET	R/W	1h	Receive FIFO Reset Reset type: SYSRSn 0h (R/W) = Write 0 to reset the FIFO pointer to zero, and hold in reset. 1h (R/W) = Re-enable receive FIFO operation.
12-8	RXFFST	R	0h	Receive FIFO Status Reset type: SYSRSn 0h (R/W) = Receive FIFO is empty. 1h (R/W) = Receive FIFO has 1 word. 2h (R/W) = Receive FIFO has 2 words. 10h (R/W) = Receive FIFO has 16 words, which is the maximum. 1Fh (R/W) = Reserved.
7	RXFFINT	R	0h	Receive FIFO Interrupt Flag Reset type: SYSRSn 0h (R/W) = RXFIFO interrupt has not occurred. This is a read-only bit. 1h (R/W) = RXFIFO interrupt has occurred. This is a read-only bit.
6	RXFFINTCLR	W	0h	Receive FIFO Interrupt Clear Reset type: SYSRSn 0h (R/W) = Write 0 has no effect on RXFIFINT flag bit, Bit reads back a zero. 1h (R/W) = Write 1 to clear SPIFFRX[RXFFINT] flag
5	RXFFIENA	R/W	0h	RX FIFO Interrupt Enable Reset type: SYSRSn 0h (R/W) = RX FIFO interrupt based on RXFFIL match (greater than or equal to) will be disabled. 1h (R/W) = RX FIFO interrupt based on RXFFIL match (greater than or equal to) will be enabled.

Table 37-18. SPIFFRX Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	RXFFIL	R/W	1Fh	Receive FIFO Interrupt Level Bits Receive FIFO generates an interrupt when the FIFO status bits (RXFFST4-0) are greater than or equal to the FIFO level bits (RXFFIL4-0). The default value of these bits after reset is 11111. This avoids frequent interrupts after reset, as the receive FIFO will be empty most of the time. Reset type: SYSRSn 0h (R/W) = A RX FIFO interrupt request is generated when there is 0 or more words in the RX buffer. 1h (R/W) = A RX FIFO interrupt request is generated when there are 1 or more words in the RX buffer. 2h (R/W) = A RX FIFO interrupt request is generated when there are 2 or more words in the RX buffer. 10h (R/W) = A RX FIFO interrupt request is generated when there are 16 words in the RX buffer. 1Fh (R/W) = Reserved.

37.6.2.11 SPIFFCT Register (Offset = Ch) [Reset = 0h]

SPIFFCT is shown in Figure 37-24 and described in Table 37-19.

Return to the Summary Table.

SPIFFCT controls the FIFO transmit delay bits.

Figure 37-24. SPIFFCT Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
TXDLY							
R/W-0h							

Table 37-19. SPIFFCT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	TXDLY	R/W	0h	FIFO Transmit Delay Bits These bits define the delay between every transfer from FIFO transmit buffer to transmit shift register. The delay is defined in number SPI serial clock cycles. The 8-bit register could define a minimum delay of 0 serial clock cycles and a maximum of 255 serial clock cycles. In FIFO mode, the buffer (TXBUF) between the shift register and the FIFO should be filled only after the shift register has completed shifting of the last bit. This is required to pass on the delay between transfers to the data stream. In the FIFO mode TXBUF should not be treated as one additional level of buffer. Reset type: SYSRSn 0h (R/W) = The next word in the TX FIFO buffer is transferred to SPITXBUF immediately upon completion of transmission of the previous word. 1h (R/W) = The next word in the TX FIFO buffer is transferred to SPITXBUF1 serial clock cycle after completion of transmission of the previous word. 2h (R/W) = The next word in the TX FIFO buffer is transferred to SPITXBUF 2 serial clock cycles after completion of transmission of the previous word. FFh (R/W) = The next word in the TX FIFO buffer is transferred to SPITXBUF 255 serial clock cycles after completion of transmission of the previous word.

37.6.2.12 SPIPRI Register (Offset = Fh) [Reset = 0h]

SPIPRI is shown in Figure 37-25 and described in Table 37-20.

Return to the Summary Table.

SPIPRI controls auxiliary functions for the SPI including emulation control, SPISTE inversion, and 3-wire control.

Figure 37-25. SPIPRI Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	SOFT	FREE	RESERVED		STEINV	TRIWIRE
R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h		R/W-0h	R/W-0h

Table 37-20. SPIPRI Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	SOFT	R/W	0h	Emulation Soft Run This bit only has an effect when the FREE bit is 0. Reset type: SYSRSn 0h (R/W) = Transmission stops midway in the bit stream while TSUSPEND is asserted. Once TSUSPEND is deasserted without a system reset, the remainder of the bits pending in the DATBUF are shifted. Example: If SPIDAT has shifted 3 out of 8 bits, the communication freezes right there. However, if TSUSPEND is later deasserted without resetting the SPI, SPI starts transmitting from where it had stopped (fourth bit in this case) and will transmit 8 bits from that point. 1h (R/W) = If the emulation suspend occurs before the start of a transmission, (that is, before the first SPICLK pulse) then the transmission will not occur. If the emulation suspend occurs after the start of a transmission, then the data will be shifted out to completion. When the start of transmission occurs is dependent on the baud rate used. Standard SPI mode: Stop after transmitting the words in the shift register and buffer. That is, after TXBUF and SPIDAT are empty. In FIFO mode: Stop after transmitting the words in the shift register and buffer. That is, after TX FIFO and SPIDAT are empty.
4	FREE	R/W	0h	Emulation Free Run These bits determine what occurs when an emulation suspend occurs (for example, when the debugger hits a breakpoint). The peripheral can continue whatever it is doing (free-run mode) or, if in stop mode, it can either stop immediately or stop when the current operation (the current receive/transmit sequence) is complete. Reset type: SYSRSn 0h (R/W) = Emulation mode is selected by the SOFT bit 1h (R/W) = Free run, continue SPI operation regardless of suspend or when the suspend occurred.
3-2	RESERVED	R	0h	Reserved
1	STEINV	R/W	0h	SPISTEn Inversion Bit On devices with 2 SPI modules, inverting the SPISTE signal on one of the modules allows the device to receive left and right- channel digital audio data. This bit is only applicable to slave mode. Writing to this bit while configured as master (MASTER_SLAVE = 1) has no effect Reset type: SYSRSn 0h (R/W) = SPISTEn is active low (normal) 1h (R/W) = SPISTE is active high (inverted)

Table 37-20. SPIPRI Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	TRIWIRE	R/W	0h	SPI 3-wire Mode Enable Reset type: SYSRSn 0h (R/W) = Normal 4-wire SPI mode. 1h (R/W) = 3-wire SPI mode enabled. The unused pin becomes a GPIO pin. In master mode, the SPISIMO pin becomes the SPIMOMI (master receive and transmit) pin and SPISOMI is free for non-SPI use. In slave mode, the SPISOMI pin becomes the SPISISO (slave receive and transmit) pin and SPISIMO is free for non-SPI use.

37.6.3 SPI Registers to Driverlib Functions

Table 37-21. SPI Registers to Driverlib Functions

File	Driverlib Function
SPICCR	
spi.c	SPI_setConfig
spi.c	SPI_clearInterruptStatus
spi.h	SPI_enableModule
spi.h	SPI_disableModule
spi.h	SPI_setcharLength
spi.h	SPI_enableLoopback
spi.h	SPI_disableLoopback
spi.h	SPI_enableHighSpeedMode
spi.h	SPI_disableHighSpeedMode
SPICTL	
spi.c	SPI_setConfig
spi.c	SPI_enableInterrupt
spi.c	SPI_disableInterrupt
spi.h	SPI_enableTalk
spi.h	SPI_disableTalk
SPISTS	
spi.c	SPI_getInterruptStatus
spi.c	SPI_clearInterruptStatus
spi.h	SPI_writeDataBlockingNonFIFO
spi.h	SPI_readDataBlockingNonFIFO
SPIBRR	
spi.c	SPI_setConfig
spi.c	SPI_setBaudRate
SPIRXEMU	
spi.h	SPI_readRxEmulationBuffer
SPIRXBUF	
spi.h	SPI_readDataNonBlocking
spi.h	SPI_readDataBlockingFIFO
spi.h	SPI_readDataBlockingNonFIFO
SPITXBUF	
spi.h	SPI_writeDataNonBlocking
spi.h	SPI_writeDataBlockingFIFO
spi.h	SPI_writeDataBlockingNonFIFO
SPIDAT	

Table 37-21. SPI Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
SPIFFTX	
spi.c	SPI_enableInterrupt
spi.c	SPI_disableInterrupt
spi.c	SPI_getInterruptStatus
spi.c	SPI_clearInterruptStatus
spi.h	SPI_enableFIFO
spi.h	SPI_disableFIFO
spi.h	SPI_resetTxFIFO
spi.h	SPI_setFIFOInterruptLevel
spi.h	SPI_getFIFOInterruptLevel
spi.h	SPI_getTxFIFOStatus
spi.h	SPI_isBusy
spi.h	SPI_reset
SPIFFRX	
spi.c	SPI_enableInterrupt
spi.c	SPI_disableInterrupt
spi.c	SPI_getInterruptStatus
spi.c	SPI_clearInterruptStatus
spi.h	SPI_enableFIFO
spi.h	SPI_disableFIFO
spi.h	SPI_resetRxFIFO
spi.h	SPI_setFIFOInterruptLevel
spi.h	SPI_getFIFOInterruptLevel
spi.h	SPI_getRxFIFOStatus
SPIFFCT	
spi.h	SPI_setTxFifoTransmitDelay
SPIPRI	
spi.h	SPI_enableTriWire
spi.h	SPI_disableTriWire
spi.h	SPI_setPTESignalPolarity
spi.h	SPI_setEmulationMode

This page intentionally left blank.

Chapter 38

Universal Serial Bus (USB) Controller

This chapter discusses the features and functions of the universal serial bus (USB) controller.

38.1 Introduction

The USB controller operates as a full-speed function controller during point-to-point communications with the USB host. The controller complies with the USB 2.0 standard, which includes SUSPEND and RESUME signaling. The USB controller has thirty-two endpoints, one-half of them being for IN transactions and one-half of them being for OUT transactions. One IN and one OUT endpoint are fixed-function endpoints used for control transfers; the others are defined by firmware. A dynamically sizeable FIFO supports queuing multiple packets. Software-controlled connect and disconnect allow flexibility during USB device startup.

38.1.1 Features

The USB module has the following features:

- Complies with USB-IF certification standards
- USB 2.0 full-speed (12 Mbps) operation in host and device modes as well as low-speed (1.5 Mbps) operation in host mode
- Integrated PHY
- Three transfer types: Control, Interrupt, and Bulk
- Thirty-two endpoints
  - One dedicated control IN endpoint and one dedicated control OUT endpoint
  - Fifteen configurable IN endpoints and fifteen configurable OUT endpoints
- Four KB dedicated endpoint memory

38.1.2 USB Related Collateral

Foundational Materials

- C2000 Academy - USB
- USB Precision Labs (Video)

Expert Materials

- *High-Speed Interface Layout Guidelines Application Report*
- *USB Flash Programming of C2000 Microcontrollers Application Report*

38.1.3 Block Diagram

The USB block diagram is shown in Figure 38-1.

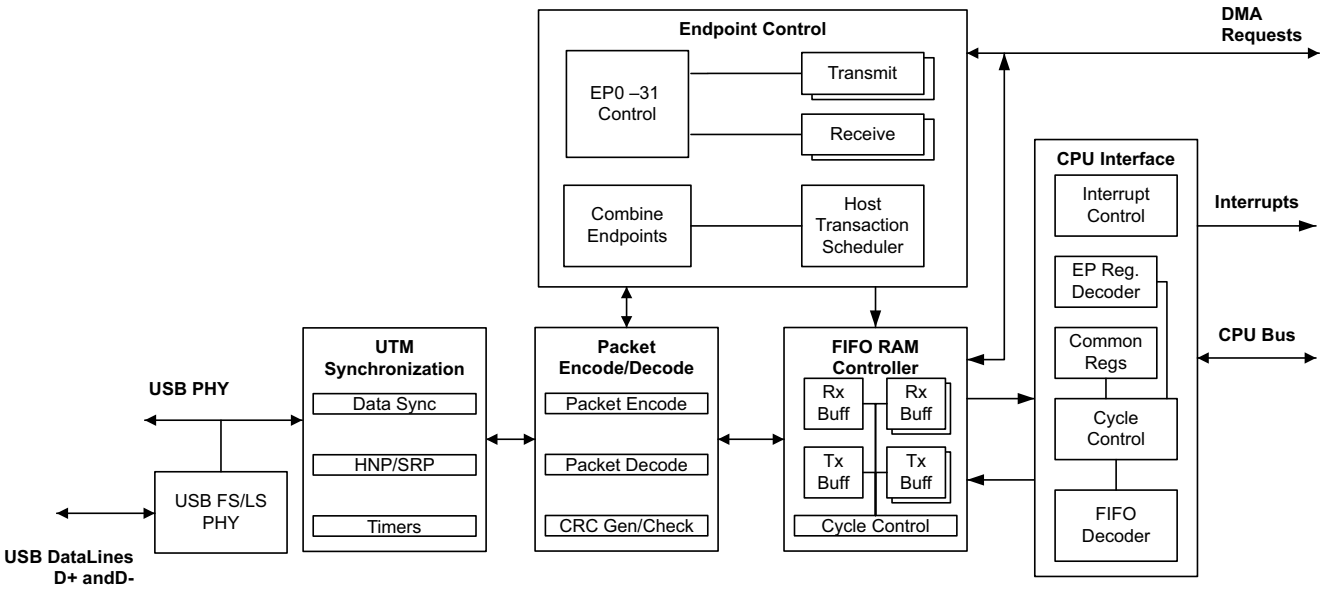


Figure 38-1. USB Block Diagram

38.1.3.1 Signal Description

The USB controller requires a total of three signals (D+, D-, and V_{Bus}) to operate in device mode and two signals (D+, D-) to operate in embedded host mode. Because of the differential signaling needed for USB, the D+ and D- pins have special buffers to support USB. As such, their position on the chip is not user-selectable. The D+ and D- pins at reset are, by default, GPIOs and must be configured before being used as USB function pins. Bits 10 and 11 in the GPIO B Analog Mode Select register (GPBAMSEL) must be set to choose the USB function. The signals USB bus voltage (V_{BUS}), external power enable (EPEN), and power fault (PFLT) are not hardwired to any pin and some applications require these signals be implemented in software using a GPIO. Software that implements these signals is available in the USB software library.

38.1.3.2 VBus Recommendations

Most applications do not need to monitor V_{Bus}. Because of this, a dedicated V_{Bus} monitoring pin was not included on this microcontroller. If you are designing a bus-powered device application or an embedded host application, you do not need to monitor V_{Bus}. If you are designing a self-powered device, you need to actively monitor the state of the V_{Bus} pin to make sure compliance with the USB specification. In Section 7.1.5 and Section 7.2.1 of the USB Specification Revision 2.0™:

- "The voltage source on the [speed identification] pull-up resistor must be derived from or controlled by the power supplied on the USB cable such that when V_{BUS} is removed, the pull-up resistor does not supply current on the data line to which it is attached.
- When V_{BUS} is removed, the device must remove power from the D+/D- pull-up resistor within 10 seconds.
- Later in the timing tables (Section 7.3.2) of the USB Specification 2.0, it is also stated that the D+/D- pull-up resistor must be applied within 100 ms of V_{Bus} reaching a valid level."

Meeting the above specification is easy because of the slow timing requirements. The hardware part of the V_{Bus} monitoring is discussed in this chapter. The corresponding software is discussed briefly, but for examples and an explanation, consult the USB software guide.

The pins of this microcontroller are not 5-V tolerant, and because of this, the V_{Bus} signal cannot be directly connected to a GPIO pin. Directly connecting 5 V to a pin of the microcontroller destroys the I/O buffer of the pin

and possibly more of the chip. The most cost-effective way of making any pin capable of reading a 5-V input is to use a series resistance in conjunction with the ESD diode clamps already present inside the device on every pin. It is recommended to use a 100-kΩ series resistor between the V_{Bus} signal and the pin chosen to monitor it. A diagram of this setup is shown in Figure 38-2.

In Figure 38-2, if V_{Bus} is above or below 3.3 V and 0 V, respectively, one of the ESD clamp diodes is forward-biased, allowing current to flow through the 100-kΩ resistor. The purpose of the diode clamps is to protect the pins of the microcontroller from very short over voltage spikes of a high magnitude. The diode clamps do this by clamping the voltage excursion to one of the supply rails. We are effectively requiring the ESD clamps to do the same thing the clamps were designed to do, but instead of a short high magnitude pulse, we are giving the clamps a long low magnitude static value using the 100-kΩ resistor.

Any pin that has digital input and output functionality can potentially be used to monitor V_{Bus}, but the use of an interrupt-capable GPIO is recommended. A pin that does not have external interrupt capability can also be used, but the input state of the pin must be polled periodically by the application software to make sure appropriate action is taken whenever V_{Bus} is applied or removed. If an interrupt-capable GPIO is chosen, the GPIO can be configured to generate an interrupt on both the rising and falling edge. More information on external interrupts can be found in the *System Control and Interrupts* chapter. Example code that implements V_{Bus} monitoring using external interrupts and takes the appropriate actions is documented in the USB Software Guide and can be found in the associated USB software package.

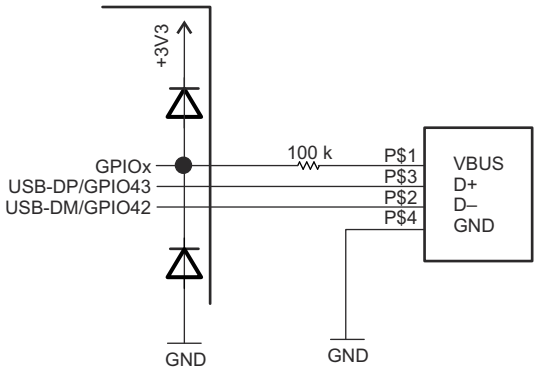


Figure 38-2. USB Scheme

38.2 Functional Description

The USB controller can be configured to act as either a dedicated host or device. However, when the USB controller is acting as a self-powered device, a GPIO input or analog comparator input must be connected to V_{BUS} and configured to generate an interrupt when the V_{BUS} level drops. This interrupt is used to disable the pullup resistor on the USB0DP signal.

Note

When USB is used in the system, the minimum system frequency is 30 MHz.

38.2.1 Operation as a Device

This section describes how the USB controller performs when the USB controller is being used as a USB device. IN endpoints, OUT endpoints, entry into and exit from SUSPEND mode, and recognition of start of frame (SOF) are all described.

When in device mode, IN transactions are controlled by the endpoint transmit interface and uses the transmit endpoint registers for the given endpoint. OUT transactions are handled with the endpoints receive interface and use the receive endpoint registers for the given endpoint. When configuring the size of the FIFOs for endpoints, take into account the maximum packet size for an endpoint. Note the following:

- Bulk endpoints must be the size of the maximum packet (up to 64 bytes) or twice the maximum packet size if double buffering is used (described further in the following section).
- Interrupt endpoints must be the size of the maximum packet (up to 64 bytes) or twice the maximum packet size if double buffering is used.
- It is also possible to specify a separate control endpoint for a USB device. However, in most cases the USB device must use the dedicated control endpoint on the USB controller’s endpoint 0.

38.2.1.1 Control and Configurable Endpoints

When operating as a device, the USB controller provides two dedicated control endpoints (IN and OUT). The remaining available configurable endpoints (one-half IN and one-half OUT) can be used for communications with a host controller. The endpoint number and direction associated with an endpoint is directly related to the register designation. For example, when the Host is transmitting to endpoint 1, all configuration and data is in the endpoint 1 transmit register interface. Endpoint 0 is a dedicated control endpoint used for all control transactions to endpoint 0 during enumeration or when any other control requests are made to endpoint 0. Endpoint 0 uses the first 64 bytes of the USB controller's FIFO RAM as a shared memory for both IN and OUT transactions. The remaining six endpoints can be configured as control, bulk, or interrupt endpoints. The six endpoints can be treated as three configurable IN and three configurable OUT endpoints. The endpoint pairs are not required to have the same type for their IN and OUT endpoint configuration. For example, the OUT portion of an endpoint pair can be a bulk endpoint, while the IN portion of that endpoint pair can be an interrupt endpoint. The address and size of the FIFOs attached to each endpoint can be modified to fit the application's needs.

38.2.1.1.1 IN Transactions as a Device

When operating as a USB device, data for IN transactions is handled through the FIFOs attached to the transmit endpoints. The sizes of the FIFOs for the configurable IN endpoints are determined by the USB Transmit FIFO Start Address (USBTXFIFOADD) register. The maximum size of a data packet that can be placed in a transmit endpoint's FIFO for transmission is programmable and is determined by the value written to the USB Maximum Transmit Data Endpoint n (USBTXMAXPn) register for that endpoint. The endpoint's FIFO can also be configured to use double-packet or single-packet buffering. When double-packet buffering is enabled, two data packets can be buffered in the FIFO, which also requires that the FIFO is at least two packets in size. When double-packet buffering is disabled, only one packet can be buffered, even if the packet size is less than half the FIFO size.

Note

The maximum packet size set for any endpoint must not exceed the FIFO size. The USBTXMAXPn register cannot be written to while data is in the FIFO as unexpected results can occur.

Single-Packet Buffering

If the size of the transmit endpoint's FIFO is less than twice the maximum packet size for this endpoint (as set in the USB Transmit Dynamic FIFO Sizing (USBTXFIFOSZ) register), only one packet can be buffered in the FIFO and single-packet buffering is required. When each packet is completely loaded into the transmit FIFO, the TXRDY bit in the USB Transmit Control and Status Endpoint n Low (USBTXCSRLn) register must be set. If the AUTOSET bit in the USB Transmit Control and Status Endpoint n High (USBTXCSRHn) register is set, the TXRDY bit is automatically set when a maximum-sized packet is loaded into the FIFO. For packet sizes less than the maximum, the TXRDY bit must be set manually. When the TXRDY bit is set, either manually or automatically, the packet is ready to be sent. When the packet has been successfully sent, both TXRDY and FIFONE are cleared, and the appropriate transmit endpoint interrupt signaled. At this point, the next packet can be loaded into the FIFO.

Double-Packet Buffering

If the size of the transmit endpoint's FIFO is at least twice the maximum packet size for this endpoint, two packets can be buffered in the FIFO and double-packet buffering is allowed. As each packet is loaded into the transmit FIFO, the TXRDY bit in the USBTXCSRLn register must be set. If the AUTOSET bit in the USBTXCSRHn register is set, the TXRDY bit is automatically set when a maximum-sized packet is loaded into the FIFO. For packet sizes less than the maximum, TXRDY must be set manually. When the TXRDY bit is set, either manually or automatically, the packet is ready to be sent. After the first packet is loaded, TXRDY is immediately cleared and an interrupt is generated. A second packet can now be loaded into the transmit FIFO and TXRDY set again (either manually or automatically if the packet is the maximum size). At this point, both packets are ready to be sent. After each packet has been successfully sent, TXRDY is automatically cleared and the appropriate transmit endpoint interrupt signaled to indicate that another packet can now be loaded into the transmit FIFO. The state of the FIFONE bit in the USBTXCSRLn register at this point indicates how many packets can be loaded. If the FIFONE bit is set, then another packet is in the FIFO and only one more packet can be loaded. If the FIFONE bit is clear, then no packets are in the FIFO and two more packets can be loaded.

Note

Double-packet buffering is disabled if an endpoint's corresponding EPn bit is set in the USB Transmit Double Packet Buffer Disable (USBTXDPKTBUFDIS) register. This bit is set by default, so the bit must be cleared to enable double-packet buffering.

38.2.1.1.2 Out Transactions as a Device

When in device mode, OUT transactions are handled through the USB controller receive FIFOs. The sizes of the receive FIFOs for the configurable OUT endpoints are determined by the USB Receive FIFO Start Address (USBRXFIFOADD) register. The maximum amount of data received by an endpoint in any packet is determined by the value written to the USB Maximum Receive Data Endpoint n (USBRXMAXPn) register for that endpoint. When double-packet buffering is enabled, two data packets can be buffered in the FIFO. When double-packet buffering is disabled, only one packet can be buffered even if the packet is less than half the FIFO size.

Note

In all cases, the maximum packet size must not exceed the FIFO size.

Single-Packet Buffering

If the size of the receive endpoint FIFO is less than twice the maximum packet size for an endpoint, only one data packet can be buffered in the FIFO and single-packet buffering is required. When a packet is received and placed in the receive FIFO, the RXRDY and FULL bits in the USB Receive Control and Status Endpoint n Low (USBRXCSRL[*n*]) register are set and the appropriate receive endpoint is signaled, indicating that a packet can now be unloaded from the FIFO. After the packet has been unloaded, the RXRDY bit must be cleared to allow further packets to be received. This action also generates the acknowledge signaling to the Host controller. If the AUTOCL bit in the USB Receive Control and Status Endpoint n High (USBRXCSRH[*n*]) register is set and a maximum-sized packet is unloaded from the FIFO, the RXRDY and FULL bits are cleared automatically. For packet sizes less than the maximum, RXRDY must be cleared manually.

Double-Packet Buffering

If the size of the receive endpoint FIFO is at least twice the maximum packet size for the endpoint, two data packets can be buffered and double-packet buffering can be used. When the first packet is received and loaded into the receive FIFO, the RXRDY bit in the USBRXCSRL[*n*] register is set and the appropriate receive endpoint interrupt is signaled to indicate that a packet can now be unloaded from the FIFO.

Note

The FULL bit in USBRXCSRL[*n*] is not set when the first packet is received. The FULL bit is only set if a second packet is received and loaded into the receive FIFO.

After each packet has been unloaded, the RXRDY bit must be cleared to allow further packets to be received. If the AUTOCL bit in the USBRXCSRH[*n*] register is set and a maximum-sized packet is unloaded from the FIFO, the RXRDY bit is cleared automatically. For packet sizes less than the maximum, RXRDY must be cleared manually. If the FULL bit is set when RXRDY is cleared, the USB controller first clears the FULL bit, then sets RXRDY again to indicate that there is another packet waiting in the FIFO to be unloaded.

Note

Double-packet buffering is disabled if an endpoint's corresponding EPn bit is set in the USB Receive Double Packet Buffer Disable (USBRXDPKTBUFDIS) register. This bit is set by default, so the bit must be cleared to enable double-packet buffering.

38.2.1.1.3 Scheduling

The device has no control over the scheduling of transactions as scheduling is determined by the Host controller. The USB controller can set up a transaction at any time. The USB controller waits for the request from the Host controller and generates an interrupt when the transaction is complete or if it was terminated due to some error. If the Host controller makes a request and the device controller is not ready, the USB controller sends a busy response (NAK) to all requests until it is ready.

38.2.1.1.4 Additional Actions

The USB controller responds automatically to certain conditions on the USB bus or actions by the Host controller such as when the USB controller automatically stalls a control transfer or unexpected zero length OUT data packets.

Stalled Control Transfer

The USB controller automatically issues a STALL handshake to a control transfer under the following conditions:

1. The Host sends more data during an OUT data phase of a control transfer than was specified in the device request during the SETUP phase. This condition is detected by the USB controller when the Host sends an OUT token (instead of an IN token) after the last OUT packet has been unloaded and the DATAEND bit in the USB Control and Status Endpoint 0 Low (USBCSRL0) register has been set.
2. The Host requests more data during an IN data phase of a control transfer than was specified in the device request during the SETUP phase. This condition is detected by the USB controller when the Host sends an IN token (instead of an OUT token) after the CPU has cleared TXRDY and set DATAEND in response to the ACK issued by the Host to what must have been the last packet.
3. The Host sends more than USBRXMAXPn bytes of data with an OUT data token.
4. The Host sends more than a zero length data packet for the OUT STATUS phase.

Zero Length OUT Data Packets

A zero-length OUT data packet is used to indicate the end of a control transfer. In normal operation, such packets must only be received after the entire length of the device request has been transferred. However, if the Host sends a zero-length OUT data packet before the entire length of device request has been transferred, the Host is signaling the premature end of the transfer. In this case, the USB controller automatically flushes any IN token ready for the data phase from the FIFO and sets the DATAEND bit in the USBCSRL0 register.

Setting the Device Address

When a Host is attempting to enumerate the USB device, the Host requests that the device change the address from zero to some other value. The address is changed by writing the value that the Host requested to the USB Device Functional Address (USBFADDR) register. However, care must be taken when writing to USBFADDR to avoid changing the address before the transaction is complete. This register must only be set after the SET_ADDRESS command is complete. Like all control transactions, the transaction is only complete after the device has left the STATUS phase. In the case of a SET_ADDRESS command, the transaction is completed by responding to the IN request from the Host with a zero-byte packet. Once the device has responded to the IN request, the USBFADDR register must be programmed to the new value as soon as possible to avoid missing any new commands sent to the new address.

Note

If the USBFADDR register is set to the new value as soon as the device receives the OUT transaction with the SET_ADDRESS command in the packet, it changes the address during the control transfer. In this case, the device does not receive the IN request that allows the USB transaction to exit the STATUS phase of the control transfer because it is sent to the old address. As a result, the Host does not get a response to the IN request, and the Host fails to enumerate the device.

38.2.1.1.5 Device Mode Suspend

When no activity has occurred on the USB bus for 3 ms, the USB controller automatically enters SUSPEND mode. If the SUSPEND interrupt has been enabled in the USB Interrupt Enable (USBIE) register, an interrupt is generated at this time. When in SUSPEND mode, the PHY also goes into SUSPEND mode. When RESUME signaling is detected, the USB controller exits SUSPEND mode and takes the PHY out of SUSPEND. If the RESUME interrupt is enabled, an interrupt is generated. The USB controller can also be forced to exit SUSPEND mode by setting the RESUME bit in the USB Power (USBPOWER) register. When this bit is set, the USB controller exits SUSPEND mode and drives RESUME signaling onto the bus. The RESUME bit must be cleared after 10 ms (a maximum of 15 ms) to end RESUME signaling. To meet USB power requirements, the controller can be put into Deep Sleep mode which keeps the controller in a static state.

38.2.1.1.6 Start of Frame

When the USB controller is operating in device mode, it receives a Start-Of-Frame (SOF) packet from the Host once every millisecond. When the SOF packet is received, the 11-bit frame number contained in the packet is written into the USB Frame Value (USBFRAME) register, and an SOF interrupt is also signaled and can be handled by the application. Once the USB controller has started to receive SOF packets, it expects one every millisecond. If no SOF packet is received after 1.00358 ms, the packet is assumed to have been lost, and the USBFRAME register is not updated. The USB controller continues and resynchronizes these pulses to the received SOF packets when these packets are successfully received again.

38.2.1.1.7 USB Reset

When the USB controller is in device mode and a RESET condition is detected on the USB bus, the USB controller automatically performs the following actions:

- Clears the USBFADDR register
- Clears the USB Endpoint Index (USBEPIDX) register
- Flushes all endpoint FIFOs
- Clears all control/status registers
- Enables all endpoint interrupts
- Generates a RESET interrupt

38.2.1.1.8 Connect/Disconnect

The USB controller connection to the USB bus is handled by software. The USB PHY can be switched between normal mode and non-driving mode by setting or clearing the SOFTCONN bit of the USBPOWER register. When the SOFTCONN bit is set, the PHY is placed in the normal mode, and the USB0DP/USB0DM lines of the USB bus are enabled. At the same time, the USB controller is placed into a state that does not respond to any USB signaling except a USB RESET. When the SOFTCONN bit is cleared, the PHY is put into non-driving mode, USB0DP and USB0DM are tristated, and the USB controller appears to other devices on the USB bus as if the USB controller has been disconnected. The non-driving mode is the default so the USB controller appears disconnected until the SOFTCONN bit has been set. The application software can then choose when to set the PHY into the normal mode. Systems with a lengthy initialization procedure can use this to make sure that initialization is complete, and the system is ready to perform enumeration before connecting to the USB bus. Once the SOFTCONN bit has been set, the USB controller can be disconnected by clearing this bit.

Note

The USB controller does not generate an interrupt when the device is connected to the Host. However, an interrupt is generated when the Host terminates a session.

38.2.2 Operation as a Host

When the USB controller is operating in Host mode, the USB controller can either be used for point-to-point communications with another USB device or, when attached to a hub, for communication with multiple devices. Full-speed and low-speed USB devices are supported, both for point-to-point communication and for operation through a hub. The USB controller automatically carries out the necessary transaction translation needed to allow a low-speed or full-speed device to be used with a USB 2.0 hub. Control, bulk, and interrupt transactions

are supported. This section describes the USB controller's actions when the USB controller is being used as a USB Host. Configuration of IN endpoints, OUT endpoints, entry into and exit from SUSPEND mode, and RESET are all described.

When in Host mode, IN transactions are controlled by an endpoint's receive interface. All IN transactions use the receive endpoint registers and all OUT endpoints use the transmit endpoint registers for a given endpoint. As in device mode, the FIFOs for endpoints must take into account the maximum packet size for an endpoint.

- Bulk endpoints must be the size of the maximum packet (up to 64 bytes) or twice the maximum packet size if double buffering is used (described further in the following section).
- Interrupt endpoints must be the size of the maximum packet (up to 64 bytes) or twice the maximum packet size if double buffering is used.
- It is also possible to specify a separate control endpoint to communicate with a device. However, in most cases the USB controller must use the dedicated control endpoint to communicate with a device's endpoint 0.

38.2.2.1 Endpoint Registers

The endpoint registers are used to control the USB endpoint interfaces which communicate with devices that are connected. The endpoints consist of a dedicated control IN endpoint and a dedicated control OUT endpoint. The remaining available endpoints are configurable, with one-half of them being OUT endpoints, and one-half of them being IN endpoints. See Section 38.1.1 for the number of available endpoints on this device.

The dedicated control interface can only be used for control transactions to endpoint 0 of devices. These control transactions are used during enumeration or other control functions that communicate using endpoint 0 of devices. This control endpoint shares the first 64 bytes of the USB controller's FIFO RAM for IN and OUT transactions. The remaining IN and OUT interfaces can be configured to communicate with control, bulk, or interrupt endpoints.

These USB interfaces can be used to simultaneously schedule as many as fifteen independent OUT and fifteen independent IN transactions to any endpoints on any device. The IN and OUT controls are paired together in the same set of registers for the respective endpoints. However, the IN and OUT controls can be configured to communicate with different types of endpoints and different endpoints on devices. For example, the first pair of endpoint controls can be split so that the OUT portion is communicating with a device's bulk OUT endpoint 1, while the IN portion is communicating with a device's interrupt IN endpoint 2.

Before accessing any device, whether for point-to-point communications or for communications via a hub, the relevant USB Receive Functional Address Endpoint n (USBRXFUNCADDRn) or USB Transmit Functional Address Endpoint n (USBTXFUNCADDRn) registers must be set for each receive or transmit endpoint to record the address of the device being accessed.

The USB controller also supports connections to devices through a USB hub by providing a register that specifies the hub address and port of each USB transfer. The FIFO address and size are customizable and can be specified for each USB IN and OUT transfer. Customization includes allowing one FIFO per transaction, sharing a FIFO across transactions, and allowing for double-buffered FIFOs.

38.2.2.2 IN Transactions as a Host

IN transactions are handled in a similar manner to the way in which OUT transactions are handled when the USB controller is in device mode except that the transaction first must be initiated by setting the REQPKT bit in the USBCSRL0 register, indicating to the transaction scheduler that there is an active transaction on this endpoint. The transaction scheduler then sends an IN token to the target device. When the packet is received and placed in the receive FIFO, the RXRDY bit in the USBCSRL0 register is set, and the appropriate receive endpoint interrupt is signaled to indicate that a packet can now be unloaded from the FIFO.

When the packet has been unloaded, RXRDY must be cleared. The AUTOCL bit in the USBRXCSRHn register can be used to have RXRDY automatically cleared when a maximum-sized packet has been unloaded from the FIFO. The AUTORQ bit in USBRXCSRHn causes the REQPKT bit to be automatically set when the RXRDY bit is cleared. When the RXRDY bit is cleared, the controller sends an acknowledge to the device. When there is a known number of packets to be transferred, the USB Request Packet Count in Block Transfer Endpoint n (USBRQPKTCOUNTn) register associated with the endpoint must be configured to the number of packets to

be transferred. The USB controller decrements the value in the USBRQPKTCOUNTn register following each request. When the USBRQPKTCOUNTn value decrements to 0, the AUTORQ bit is cleared to prevent any further transactions being attempted. For cases where the size of the transfer is unknown, USBRQPKTCOUNTn must be cleared. AUTORQ then remains set until cleared by the reception of a short packet (that is, less than the MAXLOAD value in the USBRXMAXPn register) such as can occur at the end of a bulk transfer.

If the device responds to a bulk or interrupt IN token with a NAK, the USB Host controller keeps retrying the transaction until any NAK Limit that has been set has been reached. If the target device responds with a STALL, however, the USB Host controller does not retry the transaction but sets the STALLED bit in the USBCSRL0 register. If the target device does not respond to the IN token within the required time, or the packet contained a CRC or bit-stuff error, the USB Host controller retries the transaction. If after three attempts the target device has still not responded, the USB Host controller clears the REQPKT bit and sets the ERROR bit in the USBCSRL0 register.

38.2.2.3 OUT Transactions as a Host

OUT transactions are handled in a similar manner to the way in which IN transactions are handled when the USB controller is in device mode. The TXRDY bit in the USBTXCSRLn register must be set as each packet is loaded into the transmit FIFO. Again, setting the AUTOSET bit in the USBTXCSRHn register automatically sets TXRDY when a maximum-sized packet has been loaded into the FIFO.

If the target device responds to the OUT token with a NAK, the USB Host controller keeps retrying the transaction until the NAK Limit that has been set has been reached. However, if the target device responds with a STALL, the USB controller does not retry the transaction but interrupts the main processor by setting the STALLED bit in the USBTXCSRLn register. If the target device does not respond to the OUT token within the required time, or the packet contained a CRC or bit-stuff error, the USB Host controller retries the transaction. If after three attempts the target device has still not responded, the USB controller flushes the FIFO and sets the ERROR bit in the USBTXCSRLn register.

38.2.2.4 Transaction Scheduling

Scheduling of transactions is handled automatically by the USB Host controller. The Host controller allows configuration of the endpoint communication scheduling based on the type of endpoint transaction. Interrupt transactions can be scheduled to occur in the range of every frame to every 255 frames in 1 frame increments. Bulk endpoints do not allow scheduling parameters, but do allow for a NAK timeout in the event an endpoint on a device is not responding.

The USB controller maintains a frame counter. If the target device is a full-speed device, the USB controller automatically sends an SOF packet at the start of each frame and increments the frame counter. If the target device is a low-speed device, a K state is transmitted on the bus to act as a keep-alive to stop the low-speed device from going into SUSPEND mode.

After the SOF packet has been transmitted, the USB Host controller cycles through all the configured endpoints looking for active transactions. An active transaction is defined as a receive endpoint for which the REQPKT bit is set or a transmit endpoint for which the TXRDY bit and/or the FIFONE bit is set.

An interrupt transaction is started if the transaction is found on the first scheduler cycle of a frame and if the interval counter for that endpoint has counted down to zero. As a result, only one interrupt transaction occurs per endpoint every n frames, where n is the interval set via the USB Host Transmit Interval Endpoint n (USBTXINTERVAL[*n*]) or USB Host Receive Interval Endpoint n (USBRXINTERVAL[*n*]) register for that endpoint.

An active bulk transaction starts immediately, provided sufficient time is left in the frame to complete the transaction before the next SOF packet is due. If the transaction must be retried (for example, because a NAK was received or the target device did not respond), then the transaction is not retried until the transaction scheduler has first checked all the other endpoints for active transactions. This process ensures that an endpoint that is sending a lot of NAKs does not block other transactions on the bus. The controller also allows the user to specify a limit to the length of time for NAKs to be received from a target device before the endpoint times out.

38.2.2.5 USB Hubs

The following setup requirements apply to the USB Host controller only if it is used with a USB hub. When a full- or low-speed device is connected to the USB controller using a USB 2.0 hub, details of the hub address and the hub port also must be recorded in the corresponding USB Receive Hub Address Endpoint n (USBRXHUBADDRn) and USB Receive Hub Port Endpoint n (USBRXHUBPORTn) or the USB Transmit Hub Address Endpoint n (USBTXHUBADDRn) and USB Transmit Hub Port Endpoint n (USBTXHUBPORTn) registers. In addition, the speed at which the device operates (full or low) must be recorded in the USB Type Endpoint 0 (USBTYPE0) (endpoint 0), USB Host Configure Transmit Type Endpoint n (USBTXTYPEn), or USB Host Configure Receive Type Endpoint n (USBRXTYPEn) registers for each endpoint that is accessed by the device.

For hub communications, the settings in these registers record the current allocation of the endpoints to the attached USB devices. To maximize the number of devices supported, the USB Host controller allows this allocation to be changed dynamically by simply updating the address and speed information recorded in these registers. Any changes in the allocation of endpoints to device functions must be made following the completion of any on-going transactions on the endpoints affected.

38.2.2.6 Babble

The USB Host controller does not start a transaction until the bus has been inactive for at least the minimum inter-packet delay. The controller also does not start a transaction unless it can be finished before the end of the frame. If the bus is still active at the end of a frame, then the USB Host controller assumes that the target device to which it is connected has malfunctioned, and the USB controller suspends all transactions and generates a babble interrupt.

38.2.2.7 Host SUSPEND

If the SUSPEND bit in the USBPOWER register is set, the USB Host controller completes the current transaction then stops the transaction scheduler and frame counter. No further transactions are started and no SOF packets are generated.

To exit SUSPEND mode, set the RESUME bit and clear the SUSPEND bit. While the RESUME bit is set, the USB Host controller generates RESUME signaling on the bus. After 20 ms, the RESUME bit must be cleared, at which point the frame counter and transaction scheduler start. The Host supports the detection of a remote wake-up.

38.2.2.8 USB RESET

If the RESET bit in the USBPOWER register is set, the USB Host controller generates USB RESET signaling on the bus. The RESET bit must be set for at least 20 ms to ensure correct resetting of the target device. After the CPU has cleared the bit, the USB Host controller starts its frame counter and transaction scheduler.

38.2.2.9 Connect/Disconnect

A session is started by setting the SESSION bit in the USB device Control (USBDEVCTL) register, enabling the USB controller to wait for a device to be connected. When a device is detected, a connect interrupt is generated. The speed of the device that has been connected can be determined by reading the USBDEVCTL register where the FSDEV bit is set for a full-speed device, and the LSDEV bit is set for a low-speed device. The USB controller must generate a RESET to the device, and then the USB Host controller can begin device enumeration. If the device is disconnected while a session is in progress, a disconnect interrupt is generated.

38.2.3 DMA Operation

The USB module's DMA trigger signals are not supported on this device. The DMA controller may be used to read and write the USB FIFOs via software triggering. see the Direct Memory Access (DMA) chapter for more details about programming the DMA controller. See the *USB DMA Event Trigger* advisory in the device errata for more information.

38.2.4 Address/Data Bus Bridge

This USB controller was originally designed to connect to an ARM AHB bus, but has been modified to function with the C28x device bus architecture. The modifications made are largely invisible to the user application, but there are some things to note.

- The USB memory space is 8 bits wide, while the C28x memory space is 16 bits wide.
- 32- and 16-bit accesses (r/w) are completely transparent to the user application code, no changes need be made.
- The C28x core only supports 8 bit accesses through a byte intrinsic type. This can be used to perform 8 bit reads or writes to the USB controller.
  - int &__byte(int *array, unsigned int byte_index);
  - *array = ptr to address to access, byte_index = always 0 (for USB)
    See Table 38-1 for example.
  - See the *TMS320C28x Optimizing C/C++ Compiler User's Guide* and the *TMS320C28x Assembly Language Tools User's Guide*
- Because of the bridge, the memory view of the USB controller memory space in CCS is not a 1:1 representation of what is in the controller
  - When the view mode is
    - 32 bit or 16 bit, even address are effectively duplicated, ignore odd addresses.
    - 8 bit, even addresses from within the controller are duplicated into odd address in the view window; old addresses from within the controller are not displayed.
      See Table 38-2 for example.

Table 38-1. USB Memory Access from Software

USB Controller Memory			C28x 8 Bit	
Address	**Register Name**	**Data**	**Access**	**Data**
0x00	FADDR	0x00	__byte((int *)0x00,0)	0x0000
0x01	POWER	0x11	__byte((int *)0x01,0)	0x0011
0x02	TXIS (LSB)	0x22	__byte((int *)0x02,0)	0x0022
0x03	TXIS (MSB)	0x33	__byte((int *)0x03,0)	0x0033
0x04	RXIS (LSB)	0x44	__byte((int *)0x04,0)	0x0044
0x05	RXIS (MSB)	0x55	__byte((int *)0x05,0)	0x0055
0x06	TXIE (LSB)	0x66	__byte((int *)0x06,0)	0x0066
0x07	TXIE (MSB)	0x77	__byte((int *)0x07,0)	0x0077
0x08	RXIE (LSB)	0x88	__byte((int *)0x08,0)	0x0088
0x09	RXIE (MSB)	0x99	__byte((int *)0x09,0)	0x0099
0x0A	USBIS	0xAA	__byte((int *)0x0A,0)	0x00AA
0x0B	USBIE	0xBB	__byte((int *)0x0B,0)	0x00BB
0x0C	FRAME (LSB)	0xCC	__byte((int *)0x0C,0)	0x00CC
0x0D	FRAME (MSB)	0xDD	__byte((int *)0x0D,0)	0x00DD
0x0E	EPIDX	0xEE	__byte((int *)0x0E,0)	0x00EE
0x0F	TEST	0xFF	__byte((int *)0x0F,0)	0x00FF

C28x 16 Bit			C28x 32 Bit	
Access	**Data**		**Access**	**Data**
(*((short *)(0x00)))	0x1100		(*((long *)(0x00)))	0x33221100
(*((short *)(0x01)))	0x1100		(*((long *)(0x01)))	0x33221100
(*((short *)(0x02)))	0x3322		(*((long *)(0x02)))	0x33221100
(*((short *)(0x03)))	0x3322		(*((long *)(0x03)))	0x33221100
(*((short *)(0x04)))	0x5544		(*((long *)(0x04)))	0x77665544
(*((short *)(0x05)))	0x5544		(*((long *)(0x05)))	0x77665544
(*((short *)(0x06)))	0x7766		(*((long *)(0x06)))	0x77665544
(*((short *)(0x07)))	0x7766		(*((long *)(0x07)))	0x77665544
(*((short *)(0x08)))	0x9988		(*((long *)(0x08)))	0xBBAA9988
(*((short *)(0x09)))	0x9988		(*((long *)(0x09)))	0xBBAA9988
(*((short *)(0x0A)))	0xBBAA		(*((long *)(0x0A)))	0xBBAA9988
(*((short *)(0x0B)))	0xBBAA		(*((long *)(0x0B)))	0xBBAA9988
(*((short *)(0x0C)))	0xDDCC		(*((long *)(0x0C)))	0xFFEEDDCC
(*((short *)(0x0D)))	0xDDCC		(*((long *)(0x0D)))	0xFFEEDDCC
(*((short *)(0x0E)))	0xFFEE		(*((long *)(0x0E)))	0xFFEEDDCC
(*((short *)(0x0F)))	0xFFEE		(*((long *)(0x0F)))	0xFFEEDDCC

Table 38-2. USB Memory Access from CCS IDE

CCS 8 Bit			CCS 16 Bit			CCS 32 Bit	
Address	Displayed Data		Address	Displayed Data		Address	Displayed Data
0x00	0x00		0x00	0x1100		0x00	0x11001100
0x01	0x00		0x01	0x1100		0x02	0x33223322
0x02	0x22		0x02	0x3322		0x04	0x55445544
0x03	0x22		0x03	0x3322		0x06	0x77667766
0x04	0x44		0x04	0x5544		0x08	0x99889988
0x05	0x44		0x05	0x5544		0x0A	0xBBAABBAA
0x06	0x66		0x06	0x7766		0x0C	0xDDCCDDCC
0x07	0x66		0x07	0x7766		0x0E	0xFFEEFFEE
0x08	0x88		0x08	0x9988			
0x09	0x88		0x09	0x9988			
0x0A	0xAA		0x0A	0xBBAA			
0x0B	0xAA		0x0B	0xBBAA			
0x0C	0xCC		0x0C	0xDDCC			
0x0D	0xCC		0x0D	0xDDCC			
0x0E	0xEE		0x0E	0xFFEE			
0x0F	0xEE		0x0F	0xFFEE			

38.3 Initialization and Configuration

To use the USB controller, the peripheral clock must be enabled using the System Control module PCLKCR11 register. In addition, the USB PHY signals must be connected to their respective pins using the GPIO module GPBAMSEL register. Set bits 10 and 11 for USB0DM (GPIO42) and USB0DP (GPIO43).

Set up the auxiliary PLL so a 60-MHz output clock is provided to the USB module. This fixed frequency is required for all USB operations. See the *System Control and Interrupts* chapter for more details.

In host mode, the USB controller is responsible for supplying power to the bus. To avoid incorrectly supplying voltage to the bus, the external power control signal, USB0EPEN, must be kept inactive on start-up. This can be done by connecting the USB0EPEN and USB0PFLT pins to the USB controller as soon as possible.

38.3.1 Pin Configuration

To give more flexibility, the signals External Power Enable (EPEN) and Power Fault (PFLT) were not implemented in hardware and the user must implement these signals in software. Examples of how to implement these signals in software can be found in the USB Software Guide located in C2000Ware in the \libraries\communications\usb\ directory.

When using the device controller portion of the USB controller in a system that also provides host functionality, the power to V_{BUS} must be disabled to allow the external host controller to supply power. Usually, the EPEN signal is used to control the external regulator and must be negated to avoid having two devices driving the V_{BUS} power pin on the USB connector.

When the USB controller is acting as a host, the USB controller is in control of two signals that are attached to an external voltage supply that provides power to V_{BUS}. The Host controller uses the EPEN signal to enable or disable power to the V_{BUS} pin on the USB connector. An input pin, PFLT, provides feedback when there has been a power fault on V_{BUS}. The PFLT signal can be configured to either automatically negate the EPEN signal to disable power, or the PFLT signal can generate an interrupt to the interrupt controller to allow software to handle the power fault condition. The polarity and actions related to both EPEN and PFLT are fully configurable in the USB controller. The controller also provides interrupts on device insertion and removal to allow the Host controller code to respond to these external events.

38.3.2 Endpoint Configuration

To start communication in Host or device mode, the endpoint registers must first be configured. In Host mode, this configuration establishes a connection between an endpoint register and an endpoint on a device. In device mode, an endpoint must be configured before enumerating to the Host controller.

In both cases, the endpoint 0 configuration is limited because it is a fixed-function, fixed-FIFO-size endpoint. In device and Host modes, the endpoint requires little setup but does require a software-based state machine to progress through the setup, data, and status phases of a standard control transaction. In device mode, the configuration of the remaining endpoints is done once before enumerating and then only changed if an alternate configuration is selected by the Host controller. In Host mode, the endpoints must be configured to operate as control, bulk, or interrupt mode. Once the type of endpoint is configured, a FIFO area must be assigned to each endpoint. In the case of bulk, control and interrupt endpoints, each has a maximum of 64 bytes per transaction. The maximum packet size for the given endpoint must be set prior to sending or receiving data.

Configuring each endpoint’s FIFO involves reserving a portion of the overall USB FIFO RAM to each endpoint. The total FIFO RAM available is 4 Kbytes with the first 64 bytes reserved for endpoint 0. The endpoint’s FIFO must be at least as large as the maximum packet size. The FIFO can also be configured as a double-buffered FIFO so that interrupts occur at the end of each packet and allow filling the other half of the FIFO.

If operating as a device, the USB device controller's soft connect must be enabled when the device is ready to start communications, indicating to the host controller that the device is ready to start the enumeration process. If operating as a Host controller, the device soft connect must be disabled and power must be provided to V_{BUS} via the USB0EPEN signal.

38.4 USB Global Interrupts

Global interrupt enable, flag, and clear registers have been added to ensure that no interrupt is missed. The USB interrupt can be enabled or blocked using the INTEN bit. The INTFLG bit indicates whether an interrupt has occured or not. Finally the INTFLGCLR bit will clear the INTFLG when the value '1' is written to the field.

38.5 Software

38.5.1 USB Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/usb

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

38.5.1.1 Wrapper for interrupt functions and USB support pins. - CM

FILE: usb_hal.c

38.5.1.2 USB CDC serial example

FILE: usb_ex1_dev_serial.c

This example application turns the evaluation kit into a virtual serial port when connected to the USB host system. The application supports the USB Communication Device Class, Abstract Control Model to redirect SCIA traffic to and from the USB host system.

Connect USB cables from your PC to both the mini and microUSB connectors on the controlCARD.Figure out what COM ports your controlCARD is enumerating (typically done using Device Manager in Windows) and open a serial terminal to each of with the settings 115200 Baud 8-N-1. Characters typed in one terminal should be echoed in the other and vice versa.

A driver information (INF) file for use with Windows XP, Windows 7 and Windows 10 can be found in the windows_drivers directory.

38.5.1.3 USB Composite Serial Device (usb_dev_cserial) - CM

FILE: usb_ex1_device_serial_cm.c

This example application turns the evaluation kit into a virtual serial port when connected to the USB host system. The application supports the USB Communication Device Class, Abstract Control Model to redirect UART0 traffic to and from the USB host system.

Connect USB cables from your PC to both the mini and microUSB connectors on the controlCARD.Figure out what COM ports your controlCARD is enumerating (typically done using Device Manager in Windows) and open a serial terminal to each of with the settings 115200 Baud 8-N-1. Characters typed in one terminal should be echoed in the other and vice versa.

A driver information (INF) file for use with Windows XP, Windows 7 and Windows 10 can be found in the windows_drivers directory.

38.5.1.4 USB HID Mouse Device

FILE: usb_ex2_dev_mouse.c

This example application turns the evaluation board into a USB mouse supporting the Human Interface Device class. After loading and running the example simply connect the PC to the controlCARDs microUSB port using a USB cable, and the mouse pointer will move in a square pattern for the duration of the time it is plugged in.

SCIA, connected to the FTDI virtual COM port and running at 115200, 8-N-1, is used to display messages from this application.

38.5.1.5 USB HID Mouse Device - CM

FILE: usb_ex2_device_mouse_cm.c

This example application turns the evaluation board into a USB mouse supporting the Human Interface Device class. After loading and running the example simply connect the PC to the controlCARDs microUSB port using a USB cable, and the mouse pointer will move in a square pattern for the duration of the time it is plugged in.

UART0, connected to the FTDI virtual COM port and running at 115200, 8-N-1, is used to display messages from this application.

38.5.1.6 Data structures defining the USB mouse device. - CM

FILE: usb_ex2_device_mouse_structs.c

38.5.1.7 USB Device Keyboard

FILE: usb_ex3_dev_keyboard.c

This example application turns the evaluation board into a USB keyboard supporting the Human Interface Device class. The global variable ui32Button should be modified to wake up the USB. Care should be taken to ensure that the active window can safely receive the text; enter is not pressed at any point so no actions are attempted by the host if a terminal window is used.

The device implemented by this application also supports USB remote wake up allowing it to request the host to reactivate a suspended bus. If the bus is suspended (as indicated on the application display), updating ui32Button will request a remote wakeup assuming the host has not specifically disabled such requests.

To run the example compile the project, load to the target, and run the example. After the example is running, connect a USB cable from the PC to the microUSB port on the controlCARD.Modify ui32Button value in the expressions window and then focus should be on the window so that we can receive keyboard input (i.e. NotePad).

38.5.1.8 USB HID Keyboard Device (usb_dev_keyboard) - CM

FILE: usb_ex3_device_keyboard_cm.c

This example application turns the evaluation board into a USB keyboard supporting the Human Interface Device class. The global variable ui32Button should be modified to wake up the USB. Care should be taken to ensure that the active window can safely receive the text; enter is not pressed at any point so no actions are attempted by the host if a terminal window is used.

The device implemented by this application also supports USB remote wake up allowing it to request the host to reactivate a suspended bus. If the bus is suspended (as indicated on the application display), updating ui32Button will request a remote wakeup assuming the host has not specifically disabled such requests.

To run the example compile the project, load to the target, and run the example. After the example is running, connect a USB cable from the PC to the microUSB port on the controlCARD.Modify ui32Button value in the expressions window and then focus should be on the window so that we can receive keyboard input (i.e. NotePad).

38.5.1.9 Data structures defining the USB keyboard device. - CM

FILE: usb_ex3_device_keyboard_structs.c

38.5.1.10 Data structures defining this bulk USB device. - CM

FILE: usb_ex4_device_bulk_structs.c

38.5.1.11 USB Generic Bulk Device (usb_dev_bulk) - CM

FILE: usb_ex4_device_bulk_cm.c

This example provides a generic USB device offering simple bulk data transfer to and from the host. The device uses a vendor-specific class ID and supports a single bulk IN endpoint and a single bulk OUT endpoint. Data received from the host is assumed to be ASCII text and it is echoed back with the case of all alphabetic characters swapped.

UART0, connected to the FTDI virtual COM port and running at 115200, 8-N-1, is used to display messages from this application.

A Windows INF file for the device is provided in C2000Ware. This INF contains information required to install the WinUSB subsystem on Windows. WinUSB is a Windows subsystem allowing user mode applications to access the USB device without the need for a vendor-specific kernel mode driver.

A sample Windows command-line application, usb_bulk_example, illustrating how to connect to and communicate with the bulk device is also provided. Project files are included to allow the examples to be built using Microsoft VisualStudio. Source code for this application can be found in directory MWare/tools/usb_bulk_example.

38.5.1.12 USB Generic Bulk Device

FILE: usb_ex4_dev_bulk.c

This example provides a generic USB device offering simple bulk data transfer to and from the host. The device uses a vendor-specific class ID and supports a single bulk IN endpoint and a single bulk OUT endpoint. Data received from the host is assumed to be ASCII text and it is echoed back with the case of all alphabetic characters swapped.

SCIA, connected to the FTDI virtual COM port and running at 115200, 8-N-1, is used to display messages from this application.

A Windows INF file for the device is provided under the windows drivers directory. This INF contains information required to install the WinUSB subsystem on WindowsXP, Windows 7 and Windows 10. WinUSB is a Windows subsystem allowing user mode applications to access the USB device without the need for a vendor-specific kernel mode driver.

A sample Windows command-line application, usb_bulk_example, illustrating how to connect to and communicate with the bulk device is also provided. Project files are included to allow the examples to be built using Microsoft VisualStudio. Source code for this application can be found in directory ~/C2000Ware/utilities/tools/{Device}/usb_bulk_example/Release

38.5.1.13 USB HID Mouse Host

FILE: usb_ex5_host_mouse.c

This application demonstrates the handling of a USB mouse attached to the evaluation kit. Once attached, the position of the mouse pointer and the state of the mouse buttons are output to the display.

SCIA, which is connected to the FTDI virtual serial port on the controlCARD board, is configured for 115200 bits per second, and 8-N-1 mode. When a HID compliant mouse is connected to the microUSB port on the top of the controlCARD, position and button information will be displayed to the console.

38.5.1.14 USB HID Mouse Host (usb_host_mouse) - CM

FILE: usb_ex5_host_mouse_cm.c

This application demonstrates the handling of a USB mouse attached to the evaluation kit. Once attached, the position of the mouse pointer and the state of the mouse buttons are output to the display.

UART0, which is connected to the FTDI virtual serial port on the controlCARD board, is configured for 115200 bits per second, and 8-N-1 mode. When a HID compliant mouse is connected to the microUSB port on the top of the controlCARD, position and button information will be displayed to the console.

38.5.1.15 USB HID Keyboard Host (usb_host_keyboard) - CM

FILE: usb_ex6_host_keyboard_cm.c

This example application demonstrates how to support a USB keyboard attached to the evaluation kit board. The display will show if a keyboard is currently connected and the current state of the Caps Lock key on the keyboard that is connected on the bottom status area of the screen. Pressing any keys on the keyboard will cause them to be sent out the UART0 at 115200 baud with no parity, 8 bits and 1 stop bit. Any keyboard that supports the USB HID BIOS protocol should work with this demo application.

To run the example you should connect a HID compliant keyboard to the microUSB port on the top of the controlCARD and open up a serial terminal with the above settings to view the characters typed on the keyboard.

38.5.1.16 USB HID Keyboard Host

FILE: usb_ex6_host_keyboard.c

This example application demonstrates how to support a USB keyboard attached to the evaluation kit board. The display will show if a keyboard is currently connected and the current state of the Caps Lock key on the keyboard that is connected on the bottom status area of the screen. Pressing any keys on the keyboard will cause them to be sent out the SCI at 115200 baud with no parity, 8 bits and 1 stop bit. Any keyboard that supports the USB HID BIOS protocol should work with this demo application.

To run the example you should connect a HID compliant keyboard to the microUSB port on the top of the controlCARD and open up a serial terminal with the above settings to view the characters typed on the keyboard.

38.5.1.17 USB Mass Storage Class Host

FILE: usb_ex7_host_msc.c

This example application demonstrates reading a file system from a USB mass storage class device. It makes use of FatFs, a FAT file system driver. It provides a simple command console via the SCI for issuing commands to view and navigate the file system on the mass storage device.

The first SCI, which is connected to the FTDI virtual serial port on the controlCARD board, is configured for 115200 bits per second, and 8-N-1 mode. When the program is started a message will be printed to the terminal. Type `help' for command help.

After loading and running the example, open a serial terminal with the above settings to open the command prompt. Then connect a USB MSC device to the microUSB port on the top of the controlCARD.

For additional details about FatFs, see the following site: *FatFs - Generic FAT Filesystem Module*

38.5.1.18 USB Mass Storage Class Host (usb_host_msc) - CM

FILE: usb_ex7_host_msc_cm.c

This example application demonstrates reading a file system from a USB mass storage class device. It makes use of FatFs, a FAT file system driver. It provides a simple command console via the SCI for issuing commands to view and navigate the file system on the mass storage device.

The first UART, which is connected to the FTDI virtual serial port on the controlCARD board, is configured for 115200 bits per second, and 8-N-1 mode. When the program is started a message will be printed to the terminal. Type `help' for command help.

After loading and running the example, open a serial terminal with the above settings to open the command prompt. Then connect a USB MSC device to the microUSB port on the top of the controlCARD.

For additional details about FatFs, see the following site: *FatFs - Generic FAT Filesystem Module*

38.5.1.19 USB Dual Detect

FILE: usb_ex8_dual_detect.c

This program uses a GPIO to do ID detection. If a host is connected to the device's USB port, the stack will switch to device mode and enumerate as mouse. If a mouse device is connected to the device's USB port, the stack will switch to host mode and display the mouses movement and button press information in a serial terminal.

38.5.1.20 Data structures defining this bulk USB device. - CM

FILE: usb_ex8_device_bulk_structs.c

38.5.1.21 USB Throughput Bulk Device Example (usb_ex9_throughput_dev_bulk) - CM

FILE: usb_ex8_device_bulk_throughput_cm.c

This example provides a generic USB device offering simple bulk data transfer to and from the host. The device uses a vendor-specific class ID and supports a single bulk IN Endpoint and a single bulk OUT Endpoint. Data received from the host is assumed to be ASCII text and it is echoed back with the case of all alphabetic characters swapped.

UART0, connected to the FTDI virtual COM port and running at 115200, 8-N-1, is used to display messages from this application.

A Windows INF file for the device is provided under the windows drivers directory. This INF contains information required to install the WinUSB subsystem on WindowsXP, Windows 7 and Windows 10. WinUSB is a Windows subsystem allowing user mode applications to access the USB device without the need for a vendor-specific kernel mode driver.

A sample Windows command-line application, usb_throughput_bulk_example, illustrating how to connect to and communicate with the bulk device is also provided. Project files are included to allow the examples to be built using Microsoft VisualStudio. Source code for this application can be found in directory ~/utilities/tools/usb_throughput_bulk_example/Release.

38.5.1.22 USB HUB Host example - CM

FILE: usb_ex9_host_hub_cm.c

This example application demonstrates how to support a USB keyboard and USB Mouse with a USB Hub. The display will show the connected devices on the USB hub.

To run the example you should first run the usb_config_c28 example of the C28x Side. Then run the usb_ex9_host_hub_cm Example. Then connect a USB Hub to the microUSB port on the top of the controlCARD and open up a serial terminal with the above settings to view the characters typed on the keyboard. Allow the example to run with the hub connected and then connect the USB Host Mouse or Keyboard.

When a USB Mouse is connected on the Hub the position of the mouse pointer and the state of the mouse buttons are output to the display. Similarly when a USB Keyboard is connected, any key press on the keyboard will cause them to be sent out the UART at 115200 baud with no parity, 8 bits and 1 stop bit.

This example is for depicting the usage of Hub.

There are some limitations in this example :

1. The Example fails to recognize the USB Hub and the device if the Mouse/Keyboard is already connected to the USB Hub and the Hub is connected to the Micro USB of the Control Card.
2. The same port should not be used to connect a Keyboard and mouse.

38.5.1.23 USB Throughput Bulk Device Example (usb_ex9_throughput_dev_bulk)

FILE: usb_ex9_dev_bulk_throughput.c

This example provides a throughput numbers of bulk data transfer to and from the host. The device uses a vendor-specific class ID and supports a single bulk IN Endpoint and a single bulk OUT Endpoint.

SCIA, connected to the FTDI virtual COM port and running at 115200, 8-N-1, is used to display messages from this application.

A Windows INF file for the device is provided under the windows drivers directory. This INF contains information required to install the WinUSB subsystem on WindowsXP, Windows 7 and Windows 10. This is present in utilities/windows_drivers.

A sample Windows command-line application, usb_throughput_bulk_example, illustrating how to connect to and communicate with the bulk device is also provided. Project files are included to allow the examples to be built using Microsoft VisualStudio. Source code for this application can be found in directory ~/utilities/tools/usb_throughput_bulk_example/Release.

After running the example in CCS Connect the USB Micro to the PC. Then the example will wait to receive data from the application. Run the usb_throughput_bulk example, the throughput and Data Packets Transferred.

38.5.1.24 USB HUB Host example

FILE: usb_ex10_host_hub.c

This example application demonstrates how to support a USB keyboard and USB Mouse with a USB Hub. The display will show the connected devices on the USB hub.

To run the example you should connect a USB Hub to the microUSB port on the top of the controlCARD and open up a serial terminal with the above settings to view the characters typed on the keyboard. Allow the example to run with the hub connected and then connect the USB Host Mouse or Keyboard.

When a USB Mouse is connected on the Hub the position of the mouse pointer and the state of the mouse buttons are output to the display. Similarly when a USB Keyboard is connected, any key press on the keyboard will cause them to be sent out the SCI at 115200 baud with no parity, 8 bits and 1 stop bit.

This example is for depicting the usage of Hub.

There are some limitations in this example :

1. The Example fails to recognize the USB Hub and the device if the Mouse/Keyboard is already connected to the USB Hub and the Hub is connected to the Micro USB of the Control Card.
2. The same port should not be used to connect a Keyboard and mouse.

38.6 USB Registers

38.6.1 USB Base Address Table (C28)

Table 38-3. USB Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
UsbRegs	USB_REGS	USBA_BASE	0x0004_0000	YES	-	YES	-	YES

38.6.2 USB_REGS Registers

Table 38-4 lists the memory-mapped registers for the USB_REGS registers. All register offset addresses not listed in Table 38-4 should be considered as reserved locations and the register contents should not be modified.

Table 38-4. USB_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	USBFADDR	USB Device Functional Address		Go
0h	USBPOWER	USB Power		Go
1h	USBTXIS	USB Transmit Interrupt Status		Go
2h	USBRXIS	USB Receive Interrupt Status		Go
3h	USBTXIE	USB Transmit Interrupt Enable		Go
4h	USBRXIE	USB Receive Interrupt Enable		Go
5h	USBIS	USB General Interrupt Status		Go
5h	USBIE	USB Interrupt Enable		Go
6h	USBFRAME	USB Frame Value		Go
7h	USBEPIDX	USB Endpoint Index		Go
7h	USBTEST	USB Test Mode		Go
10h	USBFIFO0	USB FIFO Endpoint 0		Go
12h	USBFIFO1	USB FIFO Endpoint 1		Go
14h	USBFIFO2	USB FIFO Endpoint 2		Go
16h	USBFIFO3	USB FIFO Endpoint 3		Go
30h	USBDEVCTL	USB Device Control		Go
31h	USBTXFIFOSZ	USB Transmit Dynamic FIFO Sizing		Go
31h	USBRXFIFOSZ	USB Receive Dynamic FIFO Sizing		Go
32h	USBTXFIFOADD	USB Transmit FIFO Start Address		Go
33h	USBRXFIFOADD	USB Receive FIFO Start Address		Go
3Dh	USBCONTIM	USB Connect Timing		Go
3Eh	USBFSEOF	USB Full-Speed Last Transaction to End of Frame Timing		Go
3Fh	USBLSEOF	USB Low-Speed Last Transaction to End of Frame Timing		Go
40h	USBTXFUNCADDR0	USB Transmit Functional Address Endpoint 0		Go
41h	USBTXHUBADDR0	USB Transmit Hub Address Endpoint 0		Go
41h	USBTXHUBPORT0	USB Transmit Hub Port Endpoint 0		Go
44h	USBTXFUNCADDR1	USB Transmit Functional Address Endpoint 1		Go
45h	USBTXHUBADDR1	USB Transmit Hub Address Endpoint 1		Go
45h	USBTXHUBPORT1	USB Transmit Hub Port Endpoint 1		Go
46h	USBRXFUNCADDR1	USB Receive Functional Address Endpoint 1		Go
47h	USBRXHUBADDR1	USB Receive Hub Address Endpoint 1		Go
47h	USBRXHUBPORT1	USB Receive Hub Port Endpoint 1		Go
48h	USBTXFUNCADDR2	USB Transmit Functional Address Endpoint 2		Go
49h	USBTXHUBADDR2	USB Transmit Hub Address Endpoint 2		Go
49h	USBTXHUBPORT2	USB Transmit Hub Port Endpoint 2		Go
4Ah	USBRXFUNCADDR2	USB Receive Functional Address Endpoint 2		Go
4Bh	USBRXHUBADDR2	USB Receive Hub Address Endpoint 2		Go
4Bh	USBRXHUBPORT2	USB Receive Hub Port Endpoint 2		Go
4Ch	USBTXFUNCADDR3	USB Transmit Functional Address Endpoint 3		Go
4Dh	USBTXHUBADDR3	USB Transmit Hub Address Endpoint 3		Go

Table 38-4. USB_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
4Dh	USBTXHUBPORT3	USB Transmit Hub Port Endpoint 3		Go
4Eh	USBRXFUNCADDR3	USB Receive Functional Address Endpoint 3		Go
4Fh	USBRXHUBADDR3	USB Receive Hub Address Endpoint 3		Go
4Fh	USBRXHUBPORT3	USB Receive Hub Port Endpoint 3		Go
81h	USBCSRL0	USB Control and Status Endpoint 0 Low		Go
81h	USBCSRH0	USB Control and Status Endpoint 0 High		Go
84h	USBCOUNT0	USB Receive Byte Count Endpoint 0		Go
85h	USBTYPE0	USB Type Endpoint 0		Go
85h	USBNAKLMT	USB NAK Limit		Go
88h	USBTXMAXP1	USB Maximum Transmit Data Endpoint 1		Go
89h	USBTXCSRL1	USB Transmit Control and Status Endpoint 1 Low		Go
89h	USBTXCSRH1	USB Transmit Control and Status Endpoint 1 High		Go
8Ah	USBRXMAXP1	USB Maximum Receive Data Endpoint 1		Go
8Bh	USBRXCSRL1	USB Receive Control and Status Endpoint 1 Low		Go
8Bh	USBRXCSRH1	USB Receive Control and Status Endpoint 1 High		Go
8Ch	USBRXCOUNT1	USB Receive Byte Count Endpoint 1		Go
8Dh	USBTXTYPE1	USB Host Transmit Configure Type Endpoint 1		Go
8Dh	USBTXINTERVAL1	USB Host Transmit Interval Endpoint 1		Go
8Eh	USBRXTYPE1	USB Host Configure Receive Type Endpoint 1		Go
8Eh	USBRXINTERVAL1	USB Host Receive Polling Interval Endpoint 1		Go
90h	USBTXMAXP2	USB Maximum Transmit Data Endpoint 2		Go
91h	USBTXCSRL2	USB Transmit Control and Status Endpoint 2 Low		Go
91h	USBTXCSRH2	USB Transmit Control and Status Endpoint 2 High		Go
92h	USBRXMAXP2	USB Maximum Receive Data Endpoint 2		Go
93h	USBRXCSRL2	USB Receive Control and Status Endpoint 2 Low		Go
93h	USBRXCSRH2	USB Receive Control and Status Endpoint 2 High		Go
94h	USBRXCOUNT2	USB Receive Byte Count Endpoint 2		Go
95h	USBTXTYPE2	USB Host Transmit Configure Type Endpoint 2		Go
95h	USBTXINTERVAL2	USB Host Transmit Interval Endpoint 2		Go
96h	USBRXTYPE2	USB Host Configure Receive Type Endpoint 2		Go
96h	USBRXINTERVAL2	USB Host Receive Polling Interval Endpoint 2		Go
98h	USBTXMAXP3	USB Maximum Transmit Data Endpoint 3		Go
99h	USBTXCSRL3	USB Transmit Control and Status Endpoint 3 Low		Go
99h	USBTXCSRH3	USB Transmit Control and Status Endpoint 3 High		Go
9Ah	USBRXMAXP3	USB Maximum Receive Data Endpoint 3		Go
9Bh	USBRXCSRL3	USB Receive Control and Status Endpoint 3 Low		Go
9Bh	USBRXCSRH3	USB Receive Control and Status Endpoint 3 High		Go
9Ch	USBRXCOUNT3	USB Receive Byte Count Endpoint 3		Go
9Dh	USBTXTYPE3	USB Host Transmit Configure Type Endpoint 3		Go
9Dh	USBTXINTERVAL3	USB Host Transmit Interval Endpoint 3		Go
9Eh	USBRXTYPE3	USB Host Configure Receive Type Endpoint 3		Go
9Eh	USBRXINTERVAL3	USB Host Receive Polling Interval Endpoint 3		Go
182h	USBRQPKTCOUNT1	USB Request Packet Count in Block Transfer Endpoint 1		Go
184h	USBRQPKTCOUNT2	USB Request Packet Count in Block Transfer Endpoint 2		Go

Table 38-4. USB_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
186h	USBRQPKTCOUNT3	USB Request Packet Count in Block Transfer Endpoint 3		Go
1A0h	USBRXDPKTBUFDIS	USB Receive Double Packet Buffer Disable		Go
1A1h	USBTXDPKTBUFDIS	USB Transmit Double Packet Buffer Disable		Go
200h	USBEPC	USB External Power Control		Go
202h	USBEPCRIS	USB External Power Control Raw Interrupt Status		Go
204h	USBEPCIM	USB External Power Control Interrupt Mask		Go
206h	USBEPCISC	USB External Power Control Interrupt Status and Clear		Go
208h	USBDRRIS	USB Device RESUME Raw Interrupt Status		Go
20Ah	USBDRIM	USB Device RESUME Interrupt Mask		Go
20Ch	USBDRISC	USB Device RESUME Interrupt Status and Clear		Go
20Eh	USBGPCS	USB General-Purpose Control and Status		Go
218h	USBVDC	USB VBUS Droop Control		Go
21Ah	USBVDCRIS	USB VBUS Droop Control Raw Interrupt Status		Go
21Ch	USBVDCIM	USB VBUS Droop Control Interrupt Mask		Go
21Eh	USBVDCISC	USB VBUS Droop Control Interrupt Status and Clear		Go
222h	USBIDVRIS	USB ID Valid Detect Raw Interrupt Status		Go
224h	USBIDVIM	USB ID Valid Detect Interrupt Mask		Go
226h	USBIDVISC	USB ID Valid Detect Interrupt Status and Clear		Go
228h	USBDMASEL	USB DMA Select		Go
240h	USB_GLB_INT_EN	USB Global Interrupt Enable Register Note: This Register is applicable only when USB is mapped to CPU1		Go
242h	USB_GLB_INT_FLG	USB Global Interrupt Flag Register Note: This Register is applicable only when USB is mapped to CPU1		Go
244h	USB_GLB_INT_FLG_CLR	USB Global Interrupt Flag Clear Register Note: This Register is applicable only when USB is mapped to CPU1		Go
280h	USBDMARIS	USB uDMA Raw Interrupt Status register. Note: This Register is applicable only when USB is mapped to CM		Go
282h	USBDMAIM	USB uDMA Interrupt Mask Register Note: This Register is applicable only when USB is mapped to CM		Go
284h	USBDMAISC	USB uDMA Interrupt Status and Clear Register Note: This Register is applicable only when USB is mapped to CM		Go

Complex bit access types are encoded to fit into small table cells. Table 38-5 shows the codes that are used for access types in this section.

Table 38-5. USB_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		

Table 38-5. USB_REGS Access Type Codes (continued)

Access Type	Code	Description
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

38.6.2.1 USBFADDR Register (Offset = 0h) [Reset = 0h]

USBFADDR is shown in Figure 38-3 and described in Table 38-6.

Return to the Summary Table.

USB Device Functional Address

Figure 38-3. USBFADDR Register

7	6	5	4	3	2	1	0
RESERVED	FUNCADDR						
R-0h	R/W-0h						

Table 38-6. USBFADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	FUNCADDR	R/W	0h	Function Address of Device as received through SET_ADDRESS Reset type: SYSRSn

38.6.2.2 USBPOWER Register (Offset = 0h) [Reset = 0h]

USBPOWER is shown in Figure 38-4 and described in Table 38-7.

Return to the Summary Table.

USB Power

Figure 38-4. USBPOWER Register

7	6	5	4	3	2	1	0
ISOUP	SOFT_CONN	RESERVED		RESET	RESUME	SUSPEND	PWRDNPHY
R/W-0h	R/W-0h	R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 38-7. USBPOWER Register Field Descriptions

Bit	Field	Type	Reset	Description
7	ISOUP	R/W	0h	Isochronous Update Reset type: SYSRSn 0h (R/W) = Host Mode - Reserved Device Mode - No effect 1h (R/W) = Host Mode - Reserved Device Mode - The USB controller waits for an SOF token from the time the TXRDY bit is set in the USBTXCSRLn register before sending the packet. If an IN token is received before an SOF token, then a zerolength data packet is sent.
6	SOFT_CONN	R/W	0h	Soft Connect/Disconnect Reset type: SYSRSn 0h (R/W) = Host Mode - Reserved Device Mode - The USB D+/D- lines are tri-stated. 1h (R/W) = Host Mode - Reserved Device Mode - The USB D+/D- lines are enabled.
5-4	RESERVED	R	0h	Reserved
3	RESET	R/W	0h	Enable Reset Signaling Reset type: SYSRSn 0h (R/W) = Ends RESET signaling on the bus. 1h (R/W) = Enables RESET signaling on the bus.
2	RESUME	R/W	0h	Enable Resume Signaling. The bit should be cleared by software 20 ms after being set. Reset type: SYSRSn 0h (R/W) = Ends RESUME signaling on the bus. 1h (R/W) = Enables RESUME signaling when the Device is in SUSPEND mode.
1	SUSPEND	R/W	0h	Enable Suspend Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Enables SUSPEND mode.
0	PWRDNPHY	R/W	0h	Power Down PHY Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Powers down the internal USB PHY.

38.6.2.3 USBTXIS Register (Offset = 1h) [Reset = 0h]

USBTXIS is shown in Figure 38-5 and described in Table 38-8.

Return to the Summary Table.

USB Transmit Interrupt Status

Figure 38-5. USBTXIS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				EP3	EP2	EP1	EP0
R-0h				R-0h	R-0h	R-0h	R-0h

Table 38-8. USBTXIS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	EP3	R	0h	Transmit Endpoint 3 Interrupt Reset type: SYSRSn 0h (R/W) = No interrupt 1h (R/W) = The Endpoint 3 transmit interrupt is asserted.
2	EP2	R	0h	Transmit Endpoint 2 Interrupt Reset type: SYSRSn 0h (R/W) = No interrupt 1h (R/W) = The Endpoint 2 transmit interrupt is asserted.
1	EP1	R	0h	Transmit Endpoint 1 Interrupt Reset type: SYSRSn 0h (R/W) = No interrupt 1h (R/W) = The Endpoint 1 transmit interrupt is asserted.
0	EP0	R	0h	Transmit Endpoint 0 Interrupt Reset type: SYSRSn 0h (R/W) = No interrupt 1h (R/W) = The Endpoint 0 transmit and receive interrupt is asserted.

38.6.2.4 USBRXIS Register (Offset = 2h) [Reset = 0h]

USBRXIS is shown in Figure 38-6 and described in Table 38-9.

Return to the Summary Table.

USB Receive Interrupt Status

Figure 38-6. USBRXIS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				EP3	EP2	EP1	RESERVED
R-0h				R-0h	R-0h	R-0h	R-0h

Table 38-9. USBRXIS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	EP3	R	0h	Recieve Endpoint 3 Interrupt Reset type: SYSRSn 0h (R/W) = No interrupt 1h (R/W) = The Endpoint 3 transmit interrupt is asserted.
2	EP2	R	0h	Recieve Endpoint 2 Interrupt Reset type: SYSRSn 0h (R/W) = No interrupt 1h (R/W) = The Endpoint 2 transmit interrupt is asserted.
1	EP1	R	0h	Recieve Endpoint 1 Interrupt Reset type: SYSRSn 0h (R/W) = No interrupt 1h (R/W) = The Endpoint 1 transmit interrupt is asserted.
0	RESERVED	R	0h	Reserved

38.6.2.5 USBTXIE Register (Offset = 3h) [Reset = Fh]

USBTXIE is shown in Figure 38-7 and described in Table 38-10.

Return to the Summary Table.

USB Transmit Interrupt Enable

Figure 38-7. USBTXIE Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				EP3	EP2	EP1	EP0
R-0h				R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 38-10. USBTXIE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	EP3	R/W	1h	Transmit Endpoint 3 Interrupt Enable Reset type: SYSRSn 0h (R/W) = The EP3 transmit interrupt is suppressed and not sent to the interrupt controller. 1h (R/W) = An interrupt is sent to the interrupt controller when the EP3 bit in the USBTXIS register is set.
2	EP2	R/W	1h	Transmit Endpoint 2 Interrupt Enable Reset type: SYSRSn 0h (R/W) = The EP2 transmit interrupt is suppressed and not sent to the interrupt controller. 1h (R/W) = An interrupt is sent to the interrupt controller when the EP2 bit in the USBTXIS register is set.
1	EP1	R/W	1h	Transmit Endpoint 1 Interrupt Enable Reset type: SYSRSn 0h (R/W) = The EP1 transmit interrupt is suppressed and not sent to the interrupt controller. 1h (R/W) = An interrupt is sent to the interrupt controller when the EP1 bit in the USBTXIS register is set.
0	EP0	R/W	1h	Transmit Endpoint 0 Interrupt Enable Reset type: SYSRSn 0h (R/W) = The EP0 transmit and receive interrupt is suppressed and not sent to the interupt controller. 1h (R/W) = An interrupt is sent to the interrupt controller when the EP0 bit in the USBTXIS register is set.

38.6.2.6 USBRXIE Register (Offset = 4h) [Reset = Eh]

USBRXIE is shown in Figure 38-8 and described in Table 38-11.

Return to the Summary Table.

USB Receive Interrupt Enable

Figure 38-8. USBRXIE Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				EP3	EP2	EP1	RESERVED
R-0h				R/W-1h	R/W-1h	R/W-1h	R-0h

Table 38-11. USBRXIE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	EP3	R/W	1h	Recieve Endpoint 3 Interrupt Enable Reset type: SYSRSn 0h (R/W) = No interrupt 1h (R/W) = The Endpoint 3 transmit interrupt is asserted.
2	EP2	R/W	1h	Recieve Endpoint 2 Interrupt Enable Reset type: SYSRSn 0h (R/W) = No interrupt 1h (R/W) = The Endpoint 2 transmit interrupt is asserted.
1	EP1	R/W	1h	Recieve Endpoint 1 Interrupt Enable Reset type: SYSRSn 0h (R/W) = No interrupt 1h (R/W) = The Endpoint 1 transmit interrupt is asserted.
0	RESERVED	R	0h	Reserved

38.6.2.7 USBIS Register (Offset = 5h) [Reset = 2Eh]

USBIS is shown in Figure 38-9 and described in Table 38-12.

Return to the Summary Table.

USB General Interrupt Status

Figure 38-9. USBIS Register

7	6	5	4	3	2	1	0
RESERVED		DISCON	RESERVED	SOF	RESET	RESUME	SUSPEND
R-0h		R/W-1h	R-0h	R/W-1h	R/W-1h	R/W-1h	R-0h

Table 38-12. USBIS Register Field Descriptions

Bit	Field	Type	Reset	Description
7-6	RESERVED	R	0h	Reserved
5	DISCON	R/W	1h	Session Disconnect Reset type: SYSRSn 0h (R/W) = No interrupt 1h (R/W) = The device has been disconnected from the host.
4	RESERVED	R	0h	Reserved
3	SOF	R/W	1h	Start of frame Reset type: SYSRSn 0h (R/W) = No interrupt 1h (R/W) = A new frame has started.
2	RESET	R/W	1h	RESET Signaling Detected Reset type: SYSRSn 0h (R/W) = No interrupt 1h (R/W) = RESET signaling has been detected on the bus.
1	RESUME	R/W	1h	RESUME Signaling Detected. Reset type: SYSRSn 0h (R/W) = No interrupt 1h (R/W) = RESUME signaling has been detected on the bus while the USB controller is in SUSPEND mode.
0	SUSPEND	R	0h	SUSPEND Signaling Detected Reset type: SYSRSn 0h (R/W) = No interrupt 1h (R/W) = SUSPEND signaling has been detected on the bus.

38.6.2.8 USBIE Register (Offset = 5h) [Reset = 2Eh]

USBIE is shown in Figure 38-10 and described in Table 38-13.

Return to the Summary Table.

USB Interrupt Enable

Figure 38-10. USBIE Register

7	6	5	4	3	2	1	0
RESERVED		DISCON	RESERVED	SOF	RESET	RESUME	SUSPEND
R-0h		R/W-1h	R-0h	R/W-1h	R/W-1h	R/W-1h	R-0h

Table 38-13. USBIE Register Field Descriptions

Bit	Field	Type	Reset	Description
7-6	RESERVED	R	0h	Reserved
5	DISCON	R/W	1h	Session Disconnect Reset type: SYSRSn 0h (R/W) = The DISCON interrupt is suppressed and not sent to the interrupt controller. 1h (R/W) = An interrupt is sent to the interrupt controller when the DISCON bit in the USBIS register is set.
4	RESERVED	R	0h	Reserved
3	SOF	R/W	1h	Start of frame Reset type: SYSRSn 0h (R/W) = The SOF interrupt is suppressed and not sent to the interrupt controller. 1h (R/W) = An interrupt is sent to the interrupt controller when the SOF bit in the USBIS register is set.
2	RESET	R/W	1h	RESET Signaling Detected Reset type: SYSRSn 0h (R/W) = The RESET interrupt is suppressed and not sent to the interrupt controller. 1h (R/W) = An interrupt is sent to the interrupt controller when the RESET bit in the USBIS register is set.
1	RESUME	R/W	1h	RESUME Signaling Detected. Reset type: SYSRSn 0h (R/W) = The RESUME interrupt is suppressed and not sent to the interrupt controller. 1h (R/W) = An interrupt is sent to the interrupt controller when the RESUME bit in the USBIS register is set.
0	SUSPEND	R	0h	SUSPEND Signaling Detected Reset type: SYSRSn 0h (R/W) = The SUSPEND interrupt is suppressed and not sent to the interrupt controller. 1h (R/W) = An interrupt is sent to the interrupt controller when the DISCON bit in the USBIS register is set.

38.6.2.9 USBFRAME Register (Offset = 6h) [Reset = 0h]

USBFRAME is shown in Figure 38-11 and described in Table 38-14.

Return to the Summary Table.

USB Frame Value

Figure 38-11. USBFRAME Register

15	14	13	12	11	10	9	8
RESERVED					FRAME		
R-0h					R-0h		
7	6	5	4	3	2	1	0
FRAME							
R-0h							

Table 38-14. USBFRAME Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	Reserved
10-0	FRAME	R	0h	Frame Number Reset type: SYSRSn

38.6.2.10 USBEPIDX Register (Offset = 7h) [Reset = 0h]

USBEPIDX is shown in Figure 38-12 and described in Table 38-15.

Return to the Summary Table.

USB Endpoint Index

Figure 38-12. USBEPIDX Register

7	6	5	4	3	2	1	0
RESERVED				EPIDX			
R-0h				R/W-0h			

Table 38-15. USBEPIDX Register Field Descriptions

Bit	Field	Type	Reset	Description
7-4	RESERVED	R	0h	Reserved
3-0	EPIDX	R/W	0h	Endpoint Index. This bit field configures which endpoint is accessed when reading or writing to one of the USB controller's indexed registers. A value of 0x0 corresponds to Endpoint 0 and a value of 0xF corresponds to Endpoint 15. Reset type: SYSRSn

38.6.2.11 USBTEST Register (Offset = 7h) [Reset = 0h]

USBTEST is shown in Figure 38-13 and described in Table 38-16.

Return to the Summary Table.

USB Test Mode

Figure 38-13. USBTEST Register

7	6	5	4	3	2	1	0
FORCEH	FIFOACC	FORCEFS	RESERVED				
R/W-0h	R/W-0h	R/W-0h	R-0h				

Table 38-16. USBTEST Register Field Descriptions

Bit	Field	Type	Reset	Description
7	FORCEH	R/W	0h	Force Host Mode. While in this mode, status of the bus connection may be read using the DEV bit of the USBDEVCTL register. The operating speed is determined from the FORCEFS bit. Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Forces the USB controller to enter Host mode when the SESSION bit is set, regardless ofwhether the USB controller is connected to any peripheral. The state of the USB0DP and USB0DM signals is ignored. The USB controller then remains in Host mode until the SESSION bit is cleared, even if a Device is disconnected. If the FORCEH bit remains set, the USB controller re-enters Host mode the next time the SESSION bit is set.
6	FIFOACC	R/W	0h	FIFO Access Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Transfers the packet in the endpoint 0 transmit FIFO to the endpoint 0 receive FIFO.
5	FORCEFS	R/W	0h	Force Full Speed Upon Reset Reset type: SYSRSn 0h (R/W) = The USB controller operates at Low Speed. 1h (R/W) = Forces the USB controller into Full-Speed mode upon receiving a USB RESET.
4-0	RESERVED	R	0h	Reserved

38.6.2.12 USBFIFO0 Register (Offset = 10h) [Reset = 0h]

USBFIFO0 is shown in Figure 38-14 and described in Table 38-17.

Return to the Summary Table.

USB FIFO Endpoint 0

Figure 38-14. USBFIFO0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EPDATA																															
R/W-0h																															

Table 38-17. USBFIFO0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	EPDATA	R/W	0h	Endpoint Data. Writing to this register loads the data into the Transmit FIFO and reading unloads data from the Receive FIFO. Reset type: SYSRSn

38.6.2.13 USBFIFO1 Register (Offset = 12h) [Reset = 0h]

USBFIFO1 is shown in Figure 38-15 and described in Table 38-18.

Return to the Summary Table.

USB FIFO Endpoint 1

Figure 38-15. USBFIFO1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EPDATA																															
R/W-0h																															

Table 38-18. USBFIFO1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	EPDATA	R/W	0h	Endpoint Data. Writing to this register loads the data into the Transmit FIFO and reading unloads data from the Receive FIFO. Reset type: SYSRSn

38.6.2.14 USBFIFO2 Register (Offset = 14h) [Reset = 0h]

USBFIFO2 is shown in Figure 38-16 and described in Table 38-19.

Return to the Summary Table.

USB FIFO Endpoint 2

Figure 38-16. USBFIFO2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EPDATA																															
R/W-0h																															

Table 38-19. USBFIFO2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	EPDATA	R/W	0h	Endpoint Data. Writing to this register loads the data into the Transmit FIFO and reading unloads data from the Receive FIFO. Reset type: SYSRSn

38.6.2.15 USBFIFO3 Register (Offset = 16h) [Reset = 0h]

USBFIFO3 is shown in Figure 38-17 and described in Table 38-20.

Return to the Summary Table.

USB FIFO Endpoint 3

Figure 38-17. USBFIFO3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EPDATA																															
R/W-0h																															

Table 38-20. USBFIFO3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	EPDATA	R/W	0h	Endpoint Data. Writing to this register loads the data into the Transmit FIFO and reading unloads data from the Receive FIFO. Reset type: SYSRSn

38.6.2.16 USBDEVCTL Register (Offset = 30h) [Reset = 4Eh]

USBDEVCTL is shown in Figure 38-18 and described in Table 38-21.

Return to the Summary Table.

USB Device Control

Figure 38-18. USBDEVCTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
DEV	FSDEV	LSDEV	VBUS		HOST	HOSTREQ	SESSION
R-0h	R/W-1h	R-0h	R/W-1h		R/W-1h	R/W-1h	R-0h

Table 38-21. USBDEVCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	DEV	R	0h	Device Mode Reset type: SYSRSn 0h (R/W) = The USB controller is operating on the OTG A side of the cable. 1h (R/W) = The USB controller is operating on the OTG B side of the cable. Only valid while a session is in progress.
6	FSDEV	R/W	1h	Full Speed Device Detected Reset type: SYSRSn 0h (R/W) = A full-speed Device has not been detected on the port. 1h (R/W) = A full-speed Device has been detected on the port.
5	LSDEV	R	0h	Low Speed Device Detected Reset type: SYSRSn 0h (R/W) = A low-speed Device has not been detected on the port. 1h (R/W) = A low-speed Device has been detected on the port.
4-3	VBUS	R/W	1h	Vbus Level Reset type: SYSRSn 0h (R/W) = Above AValid, below VBusValid. VBUS is detected as above 1.5 V and below 4.75 V. 1h (R/W) = Above VBusValid. VBUS is detected as above 4.75 V.
2	HOST	R/W	1h	Host Mode Reset type: SYSRSn 0h (R/W) = The USB controller is acting as a Device. 1h (R/W) = The USB controller is acting as a Host. Only valid while a session is in progress.
1	HOSTREQ	R/W	1h	When set, the USB controller will initiate the Host Negotiation when Suspend mode is entered. It is cleared when Host Negotiation is completed. Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Initiates the Host Negotiation when SUSPENDmode is entered.

Table 38-21. USBDEVCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	SESSION	R	0h	Session Start/End Reset type: SYSRSn 0h (R/W) = When operating as a Host: When cleared by software, this bit ends a session. When operating as a Device: The USB controller has ended a session. When the USB controller is in SUSPEND mode, this bit may be cleared by software to perform a software disconnect. 1h (R/W) = When operating as a Host: When set by software, this bit starts a session. When operating as a Device: The USB controller has started a session. When set by software, the Session Request Protocol is initiated. Clearing this bit when the USB controller is not suspended results in undefined behavior.

38.6.2.17 USBTXFIFOSZ Register (Offset = 31h) [Reset = 0h]

USBTXFIFOSZ is shown in Figure 38-19 and described in Table 38-22.

Return to the Summary Table.

USB Transmit Dynamic FIFO Sizing

Figure 38-19. USBTXFIFOSZ Register

7	6	5	4	3	2	1	0
RESERVED			DPB	SIZE			
R-0h			R/W-0h	R-0h			

Table 38-22. USBTXFIFOSZ Register Field Descriptions

Bit	Field	Type	Reset	Description
7-5	RESERVED	R	0h	Reserved
4	DPB	R/W	0h	Double Packet Buffer Support Reset type: SYSRSn 0h (R/W) = Single packet buffering is supported. 1h (R/W) = Double packet buffering is enabled.
3-0	SIZE	R	0h	Max Packet Size Reset type: SYSRSn 0h (R/W) = 8.0 1h (R/W) = 16.0 2h (R/W) = 32.0 3h (R/W) = 64.0 4h (R/W) = 128.0 5h (R/W) = 256.0 6h (R/W) = 512.0 7h (R/W) = 1024.0 8h (R/W) = 2048.0 9h (R/W) = Reserved Ah (R/W) = Reserved Bh (R/W) = Reserved Ch (R/W) = Reserved Dh (R/W) = Reserved Eh (R/W) = Reserved Fh (R/W) = Reserved

38.6.2.18 USBRXFIFOSZ Register (Offset = 31h) [Reset = 0h]

USBRXFIFOSZ is shown in Figure 38-20 and described in Table 38-23.

Return to the Summary Table.

USB Receive Dynamic FIFO Sizing

Figure 38-20. USBRXFIFOSZ Register

7	6	5	4	3	2	1	0
RESERVED			DPB	SIZE			
R-0h			R/W-0h	R-0h			

Table 38-23. USBRXFIFOSZ Register Field Descriptions

Bit	Field	Type	Reset	Description
7-5	RESERVED	R	0h	Reserved
4	DPB	R/W	0h	Double Packet Buffer Support Reset type: SYSRSn 0h (R/W) = Single packet buffering is supported. 1h (R/W) = Double packet buffering is enabled.
3-0	SIZE	R	0h	Maximum packet size to be allowed. If DPB = 0, the FIFO also is this size if DPB = 1, the FIFO is twice this size. Packet size in bytes: Reset type: SYSRSn 0h (R/W) = 8.0 1h (R/W) = 16.0 2h (R/W) = 32.0 3h (R/W) = 64.0 4h (R/W) = 128.0 5h (R/W) = 256.0 6h (R/W) = 512.0 7h (R/W) = 1024.0 8h (R/W) = 2048.0 9h (R/W) = Reserved Ah (R/W) = Reserved Bh (R/W) = Reserved Ch (R/W) = Reserved Dh (R/W) = Reserved Eh (R/W) = Reserved Fh (R/W) = Reserved

38.6.2.19 USBTXFIFOADD Register (Offset = 32h) [Reset = 0h]

USBTXFIFOADD is shown in Figure 38-21 and described in Table 38-24.

Return to the Summary Table.

USB Transmit FIFO Start Address

Figure 38-21. USBTXFIFOADD Register

15	14	13	12	11	10	9	8
RESERVED							ADDR
R-0h							R/W-0h
7	6	5	4	3	2	1	0
ADDR							
R/W-0h							

Table 38-24. USBTXFIFOADD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	RESERVED	R	0h	Reserved

Table 38-24. USBTXFIFOADD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8-0	ADDR	R/W	0h	Endpoint Data Reset type: SYSRSn 0h (R/W) = 0.0 1h (R/W) = 8.0 2h (R/W) = 16.0 3h (R/W) = 24.0 4h (R/W) = 32.0 5h (R/W) = 40.0 6h (R/W) = 48.0 7h (R/W) = 56.0 8h (R/W) = 64.0 9h (R/W) = 72.0 Ah (R/W) = 80.0 Bh (R/W) = 88.0 Ch (R/W) = 96.0 Dh (R/W) = 104.0 Eh (R/W) = 112.0 Fh (R/W) = 120.0 10h (R/W) = 128.0 11h (R/W) = 136.0 12h (R/W) = 144.0 13h (R/W) = 152.0 14h (R/W) = 160.0 15h (R/W) = 168.0 16h (R/W) = 176.0 17h (R/W) = 184.0 18h (R/W) = 192.0 19h (R/W) = 200.0 1Ah (R/W) = 208.0 1Bh (R/W) = 216.0 1Ch (R/W) = 224.0 1Dh (R/W) = 232.0 1Eh (R/W) = 240.0 1Fh (R/W) = 248.0 20h (R/W) = 256.0 21h (R/W) = 264.0 22h (R/W) = 272.0 23h (R/W) = 280.0 24h (R/W) = 288.0 25h (R/W) = 296.0 26h (R/W) = 304.0 27h (R/W) = 312.0 28h (R/W) = 320.0 29h (R/W) = 328.0 2Ah (R/W) = 336.0 2Bh (R/W) = 344.0 2Ch (R/W) = 352.0 2Dh (R/W) = 360.0 2Eh (R/W) = 368.0 2Fh (R/W) = 376.0 30h (R/W) = 384.0 31h (R/W) = 392.0 32h (R/W) = 400.0 33h (R/W) = 408.0 34h (R/W) = 416.0 35h (R/W) = 424.0 36h (R/W) = 432.0 37h (R/W) = 440.0 38h (R/W) = 448.0 39h (R/W) = 456.0 3Ah (R/W) = 464.0 3Bh (R/W) = 472.0 3Ch (R/W) = 480.0 3Dh (R/W) = 488.0 3Eh (R/W) = 496.0

Table 38-24. USBTXFIFOADD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				3Fh (R/W) = 504.0 40h (R/W) = 512.0 41h (R/W) = 520.0 42h (R/W) = 528.0 43h (R/W) = 536.0 44h (R/W) = 544.0 45h (R/W) = 552.0 46h (R/W) = 560.0 47h (R/W) = 568.0 48h (R/W) = 576.0 49h (R/W) = 584.0 4Ah (R/W) = 592.0 4Bh (R/W) = 600.0 4Ch (R/W) = 608.0 4Dh (R/W) = 616.0 4Eh (R/W) = 624.0 4Fh (R/W) = 632.0 50h (R/W) = 640.0 51h (R/W) = 648.0 52h (R/W) = 656.0 53h (R/W) = 664.0 54h (R/W) = 672.0 55h (R/W) = 680.0 56h (R/W) = 688.0 57h (R/W) = 696.0 58h (R/W) = 704.0 59h (R/W) = 712.0 5Ah (R/W) = 720.0 5Bh (R/W) = 728.0 5Ch (R/W) = 736.0 5Dh (R/W) = 744.0 5Eh (R/W) = 752.0 5Fh (R/W) = 760.0 60h (R/W) = 768.0 61h (R/W) = 776.0 62h (R/W) = 784.0 63h (R/W) = 792.0 64h (R/W) = 800.0 65h (R/W) = 808.0 66h (R/W) = 816.0 67h (R/W) = 824.0 68h (R/W) = 832.0 69h (R/W) = 840.0 6Ah (R/W) = 848.0 6Bh (R/W) = 856.0 6Ch (R/W) = 864.0 6Dh (R/W) = 872.0 6Eh (R/W) = 880.0 6Fh (R/W) = 888.0 70h (R/W) = 896.0 71h (R/W) = 904.0 72h (R/W) = 912.0 73h (R/W) = 920.0 74h (R/W) = 928.0 75h (R/W) = 936.0 76h (R/W) = 944.0 77h (R/W) = 952.0 78h (R/W) = 960.0 79h (R/W) = 968.0 7Ah (R/W) = 976.0 7Bh (R/W) = 984.0 7Ch (R/W) = 992.0 7Dh (R/W) = 1000.0 7Eh (R/W) = 1008.0 7Fh (R/W) = 1016.0

Table 38-24. USBTXFIFOADD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				80h (R/W) = 1024.0 81h (R/W) = 1032.0 82h (R/W) = 1040.0 83h (R/W) = 1048.0 84h (R/W) = 1056.0 85h (R/W) = 1064.0 86h (R/W) = 1072.0 87h (R/W) = 1080.0 88h (R/W) = 1088.0 89h (R/W) = 1096.0 8Ah (R/W) = 1104.0 8Bh (R/W) = 1112.0 8Ch (R/W) = 1120.0 8Dh (R/W) = 1128.0 8Eh (R/W) = 1136.0 8Fh (R/W) = 1144.0 90h (R/W) = 1152.0 91h (R/W) = 1160.0 92h (R/W) = 1168.0 93h (R/W) = 1176.0 94h (R/W) = 1184.0 95h (R/W) = 1192.0 96h (R/W) = 1200.0 97h (R/W) = 1208.0 98h (R/W) = 1216.0 99h (R/W) = 1224.0 9Ah (R/W) = 1232.0 9Bh (R/W) = 1240.0 9Ch (R/W) = 1248.0 9Dh (R/W) = 1256.0 9Eh (R/W) = 1264.0 9Fh (R/W) = 1272.0 A0h (R/W) = 1280.0 A1h (R/W) = 1288.0 A2h (R/W) = 1296.0 A3h (R/W) = 1304.0 A4h (R/W) = 1312.0 A5h (R/W) = 1320.0 A6h (R/W) = 1328.0 A7h (R/W) = 1336.0 A8h (R/W) = 1344.0 A9h (R/W) = 1352.0 AAh (R/W) = 1360.0 ABh (R/W) = 1368.0 ACh (R/W) = 1376.0 ADh (R/W) = 1384.0 AEh (R/W) = 1392.0 AFh (R/W) = 1400.0 B0h (R/W) = 1408.0 B1h (R/W) = 1416.0 B2h (R/W) = 1424.0 B3h (R/W) = 1432.0 B4h (R/W) = 1440.0 B5h (R/W) = 1448.0 B6h (R/W) = 1456.0 B7h (R/W) = 1464.0 B8h (R/W) = 1472.0 B9h (R/W) = 1480.0 BAh (R/W) = 1488.0 BBh (R/W) = 1496.0 BCh (R/W) = 1504.0 BDh (R/W) = 1512.0 BEh (R/W) = 1520.0 BFh (R/W) = 1528.0 C0h (R/W) = 1536.0

Table 38-24. USBTXFIFOADD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				C1h (R/W) = 1544.0 C2h (R/W) = 1552.0 C3h (R/W) = 1560.0 C4h (R/W) = 1568.0 C5h (R/W) = 1576.0 C6h (R/W) = 1584.0 C7h (R/W) = 1592.0 C8h (R/W) = 1600.0 C9h (R/W) = 1608.0 CAh (R/W) = 1616.0 CBh (R/W) = 1624.0 CCh (R/W) = 1632.0 CDh (R/W) = 1640.0 CEh (R/W) = 1648.0 CFh (R/W) = 1656.0 D0h (R/W) = 1664.0 D1h (R/W) = 1672.0 D2h (R/W) = 1680.0 D3h (R/W) = 1688.0 D4h (R/W) = 1696.0 D5h (R/W) = 1704.0 D6h (R/W) = 1712.0 D7h (R/W) = 1720.0 D8h (R/W) = 1728.0 D9h (R/W) = 1736.0 DAh (R/W) = 1744.0 DBh (R/W) = 1752.0 DCh (R/W) = 1760.0 DDh (R/W) = 1768.0 DEh (R/W) = 1776.0 DFh (R/W) = 1784.0 E0h (R/W) = 1792.0 E1h (R/W) = 1800.0 E2h (R/W) = 1808.0 E3h (R/W) = 1816.0 E4h (R/W) = 1824.0 E5h (R/W) = 1832.0 E6h (R/W) = 1840.0 E7h (R/W) = 1848.0 E8h (R/W) = 1856.0 E9h (R/W) = 1864.0 EAh (R/W) = 1872.0 EBh (R/W) = 1880.0 ECh (R/W) = 1888.0 EDh (R/W) = 1896.0 EEh (R/W) = 1904.0 EFh (R/W) = 1912.0 F0h (R/W) = 1920.0 F1h (R/W) = 1928.0 F2h (R/W) = 1936.0 F3h (R/W) = 1944.0 F4h (R/W) = 1952.0 F5h (R/W) = 1960.0 F6h (R/W) = 1968.0 F7h (R/W) = 1976.0 F8h (R/W) = 1984.0 F9h (R/W) = 1992.0 FAh (R/W) = 2000.0 FBh (R/W) = 2008.0 FCh (R/W) = 2016.0 FDh (R/W) = 2024.0 FEh (R/W) = 2032.0 FFh (R/W) = 2040.0 100h (R/W) = 2048.0 101h (R/W) = 2056.0

Table 38-24. USBTXFIFOADD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				102h (R/W) = 2064.0 103h (R/W) = 2072.0 104h (R/W) = 2080.0 105h (R/W) = 2088.0 106h (R/W) = 2096.0 107h (R/W) = 2104.0 108h (R/W) = 2112.0 109h (R/W) = 2120.0 10Ah (R/W) = 2128.0 10Bh (R/W) = 2136.0 10Ch (R/W) = 2144.0 10Dh (R/W) = 2152.0 10Eh (R/W) = 2160.0 10Fh (R/W) = 2168.0 110h (R/W) = 2176.0 111h (R/W) = 2184.0 112h (R/W) = 2192.0 113h (R/W) = 2200.0 114h (R/W) = 2208.0 115h (R/W) = 2216.0 116h (R/W) = 2224.0 117h (R/W) = 2232.0 118h (R/W) = 2240.0 119h (R/W) = 2248.0 11Ah (R/W) = 2256.0 11Bh (R/W) = 2264.0 11Ch (R/W) = 2272.0 11Dh (R/W) = 2280.0 11Eh (R/W) = 2288.0 11Fh (R/W) = 2296.0 120h (R/W) = 2304.0 121h (R/W) = 2312.0 122h (R/W) = 2320.0 123h (R/W) = 2328.0 124h (R/W) = 2336.0 125h (R/W) = 2344.0 126h (R/W) = 2352.0 127h (R/W) = 2360.0 128h (R/W) = 2368.0 129h (R/W) = 2376.0 12Ah (R/W) = 2384.0 12Bh (R/W) = 2392.0 12Ch (R/W) = 2400.0 12Dh (R/W) = 2408.0 12Eh (R/W) = 2416.0 12Fh (R/W) = 2424.0 130h (R/W) = 2432.0 131h (R/W) = 2440.0 132h (R/W) = 2448.0 133h (R/W) = 2456.0 134h (R/W) = 2464.0 135h (R/W) = 2472.0 136h (R/W) = 2480.0 137h (R/W) = 2488.0 138h (R/W) = 2496.0 139h (R/W) = 2504.0 13Ah (R/W) = 2512.0 13Bh (R/W) = 2520.0 13Ch (R/W) = 2528.0 13Dh (R/W) = 2536.0 13Eh (R/W) = 2544.0 13Fh (R/W) = 2552.0 140h (R/W) = 2560.0 141h (R/W) = 2568.0 142h (R/W) = 2576.0

Table 38-24. USBTXFIFOADD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				143h (R/W) = 2584.0 144h (R/W) = 2592.0 145h (R/W) = 2600.0 146h (R/W) = 2608.0 147h (R/W) = 2616.0 148h (R/W) = 2624.0 149h (R/W) = 2632.0 14Ah (R/W) = 2640.0 14Bh (R/W) = 2648.0 14Ch (R/W) = 2656.0 14Dh (R/W) = 2664.0 14Eh (R/W) = 2672.0 14Fh (R/W) = 2680.0 150h (R/W) = 2688.0 151h (R/W) = 2696.0 152h (R/W) = 2704.0 153h (R/W) = 2712.0 154h (R/W) = 2720.0 155h (R/W) = 2728.0 156h (R/W) = 2736.0 157h (R/W) = 2744.0 158h (R/W) = 2752.0 159h (R/W) = 2760.0 15Ah (R/W) = 2768.0 15Bh (R/W) = 2776.0 15Ch (R/W) = 2784.0 15Dh (R/W) = 2792.0 15Eh (R/W) = 2800.0 15Fh (R/W) = 2808.0 160h (R/W) = 2816.0 161h (R/W) = 2824.0 162h (R/W) = 2832.0 163h (R/W) = 2840.0 164h (R/W) = 2848.0 165h (R/W) = 2856.0 166h (R/W) = 2864.0 167h (R/W) = 2872.0 168h (R/W) = 2880.0 169h (R/W) = 2888.0 16Ah (R/W) = 2896.0 16Bh (R/W) = 2904.0 16Ch (R/W) = 2912.0 16Dh (R/W) = 2920.0 16Eh (R/W) = 2928.0 16Fh (R/W) = 2936.0 170h (R/W) = 2944.0 171h (R/W) = 2952.0 172h (R/W) = 2960.0 173h (R/W) = 2968.0 174h (R/W) = 2976.0 175h (R/W) = 2984.0 176h (R/W) = 2992.0 177h (R/W) = 3000.0 178h (R/W) = 3008.0 179h (R/W) = 3016.0 17Ah (R/W) = 3024.0 17Bh (R/W) = 3032.0 17Ch (R/W) = 3040.0 17Dh (R/W) = 3048.0 17Eh (R/W) = 3056.0 17Fh (R/W) = 3064.0 180h (R/W) = 3072.0 181h (R/W) = 3080.0 182h (R/W) = 3088.0 183h (R/W) = 3096.0

Table 38-24. USBTXFIFOADD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				184h (R/W) = 3104.0 185h (R/W) = 3112.0 186h (R/W) = 3120.0 187h (R/W) = 3128.0 188h (R/W) = 3136.0 189h (R/W) = 3144.0 18Ah (R/W) = 3152.0 18Bh (R/W) = 3160.0 18Ch (R/W) = 3168.0 18Dh (R/W) = 3176.0 18Eh (R/W) = 3184.0 18Fh (R/W) = 3192.0 190h (R/W) = 3200.0 191h (R/W) = 3208.0 192h (R/W) = 3216.0 193h (R/W) = 3224.0 194h (R/W) = 3232.0 195h (R/W) = 3240.0 196h (R/W) = 3248.0 197h (R/W) = 3256.0 198h (R/W) = 3264.0 199h (R/W) = 3272.0 19Ah (R/W) = 3280.0 19Bh (R/W) = 3288.0 19Ch (R/W) = 3296.0 19Dh (R/W) = 3304.0 19Eh (R/W) = 3312.0 19Fh (R/W) = 3320.0 1A0h (R/W) = 3328.0 1A1h (R/W) = 3336.0 1A2h (R/W) = 3344.0 1A3h (R/W) = 3352.0 1A4h (R/W) = 3360.0 1A5h (R/W) = 3368.0 1A6h (R/W) = 3376.0 1A7h (R/W) = 3384.0 1A8h (R/W) = 3392.0 1A9h (R/W) = 3400.0 1AAh (R/W) = 3408.0 1ABh (R/W) = 3416.0 1ACh (R/W) = 3424.0 1ADh (R/W) = 3432.0 1AEh (R/W) = 3440.0 1AFh (R/W) = 3448.0 1B0h (R/W) = 3456.0 1B1h (R/W) = 3464.0 1B2h (R/W) = 3472.0 1B3h (R/W) = 3480.0 1B4h (R/W) = 3488.0 1B5h (R/W) = 3496.0 1B6h (R/W) = 3504.0 1B7h (R/W) = 3512.0 1B8h (R/W) = 3520.0 1B9h (R/W) = 3528.0 1BAh (R/W) = 3536.0 1BBh (R/W) = 3544.0 1BCh (R/W) = 3552.0 1BDh (R/W) = 3560.0 1BEh (R/W) = 3568.0 1BFh (R/W) = 3576.0 1C0h (R/W) = 3584.0 1C1h (R/W) = 3592.0 1C2h (R/W) = 3600.0 1C3h (R/W) = 3608.0 1C4h (R/W) = 3616.0

Table 38-24. USBTXFIFOADD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				1C5h (R/W) = 3624.0 1C6h (R/W) = 3632.0 1C7h (R/W) = 3640.0 1C8h (R/W) = 3648.0 1C9h (R/W) = 3656.0 1CAh (R/W) = 3664.0 1CBh (R/W) = 3672.0 1CCh (R/W) = 3680.0 1CDh (R/W) = 3688.0 1CEh (R/W) = 3696.0 1CFh (R/W) = 3704.0 1D0h (R/W) = 3712.0 1D1h (R/W) = 3720.0 1D2h (R/W) = 3728.0 1D3h (R/W) = 3736.0 1D4h (R/W) = 3744.0 1D5h (R/W) = 3752.0 1D6h (R/W) = 3760.0 1D7h (R/W) = 3768.0 1D8h (R/W) = 3776.0 1D9h (R/W) = 3784.0 1DAh (R/W) = 3792.0 1DBh (R/W) = 3800.0 1DCh (R/W) = 3808.0 1DDh (R/W) = 3816.0 1DEh (R/W) = 3824.0 1DFh (R/W) = 3832.0 1E0h (R/W) = 3840.0 1E1h (R/W) = 3848.0 1E2h (R/W) = 3856.0 1E3h (R/W) = 3864.0 1E4h (R/W) = 3872.0 1E5h (R/W) = 3880.0 1E6h (R/W) = 3888.0 1E7h (R/W) = 3896.0 1E8h (R/W) = 3904.0 1E9h (R/W) = 3912.0 1EAh (R/W) = 3920.0 1EBh (R/W) = 3928.0 1ECh (R/W) = 3936.0 1EDh (R/W) = 3944.0 1EEh (R/W) = 3952.0 1EFh (R/W) = 3960.0 1F0h (R/W) = 3968.0 1F1h (R/W) = 3976.0 1F2h (R/W) = 3984.0 1F3h (R/W) = 3992.0 1F4h (R/W) = 4000.0 1F5h (R/W) = 4008.0 1F6h (R/W) = 4016.0 1F7h (R/W) = 4024.0 1F8h (R/W) = 4032.0 1F9h (R/W) = 4040.0 1FAh (R/W) = 4048.0 1FBh (R/W) = 4056.0 1FCh (R/W) = 4064.0 1FDh (R/W) = 4072.0 1FEh (R/W) = 4080.0 1FFh (R/W) = 4088.0

38.6.2.20 USBRXFIFOADD Register (Offset = 33h) [Reset = 0h]

USBRXFIFOADD is shown in Figure 38-22 and described in Table 38-25.

Return to the Summary Table.

USB Receive FIFO Start Address

Figure 38-22. USBRXFIFOADD Register

15	14	13	12	11	10	9	8
RESERVED							ADDR
R-0h							R/W-0h
7	6	5	4	3	2	1	0
ADDR							
R/W-0h							

Table 38-25. USBRXFIFOADD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	RESERVED	R	0h	Reserved

Table 38-25. USBRXFIFOADD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8-0	ADDR	R/W	0h	Endpoint Data Reset type: SYSRSn 0h (R/W) = 0.0 1h (R/W) = 8.0 2h (R/W) = 16.0 3h (R/W) = 24.0 4h (R/W) = 32.0 5h (R/W) = 40.0 6h (R/W) = 48.0 7h (R/W) = 56.0 8h (R/W) = 64.0 9h (R/W) = 72.0 Ah (R/W) = 80.0 Bh (R/W) = 88.0 Ch (R/W) = 96.0 Dh (R/W) = 104.0 Eh (R/W) = 112.0 Fh (R/W) = 120.0 10h (R/W) = 128.0 11h (R/W) = 136.0 12h (R/W) = 144.0 13h (R/W) = 152.0 14h (R/W) = 160.0 15h (R/W) = 168.0 16h (R/W) = 176.0 17h (R/W) = 184.0 18h (R/W) = 192.0 19h (R/W) = 200.0 1Ah (R/W) = 208.0 1Bh (R/W) = 216.0 1Ch (R/W) = 224.0 1Dh (R/W) = 232.0 1Eh (R/W) = 240.0 1Fh (R/W) = 248.0 20h (R/W) = 256.0 21h (R/W) = 264.0 22h (R/W) = 272.0 23h (R/W) = 280.0 24h (R/W) = 288.0 25h (R/W) = 296.0 26h (R/W) = 304.0 27h (R/W) = 312.0 28h (R/W) = 320.0 29h (R/W) = 328.0 2Ah (R/W) = 336.0 2Bh (R/W) = 344.0 2Ch (R/W) = 352.0 2Dh (R/W) = 360.0 2Eh (R/W) = 368.0 2Fh (R/W) = 376.0 30h (R/W) = 384.0 31h (R/W) = 392.0 32h (R/W) = 400.0 33h (R/W) = 408.0 34h (R/W) = 416.0 35h (R/W) = 424.0 36h (R/W) = 432.0 37h (R/W) = 440.0 38h (R/W) = 448.0 39h (R/W) = 456.0 3Ah (R/W) = 464.0 3Bh (R/W) = 472.0 3Ch (R/W) = 480.0 3Dh (R/W) = 488.0 3Eh (R/W) = 496.0

Table 38-25. USBRXFIFOADD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				3Fh (R/W) = 504.0 40h (R/W) = 512.0 41h (R/W) = 520.0 42h (R/W) = 528.0 43h (R/W) = 536.0 44h (R/W) = 544.0 45h (R/W) = 552.0 46h (R/W) = 560.0 47h (R/W) = 568.0 48h (R/W) = 576.0 49h (R/W) = 584.0 4Ah (R/W) = 592.0 4Bh (R/W) = 600.0 4Ch (R/W) = 608.0 4Dh (R/W) = 616.0 4Eh (R/W) = 624.0 4Fh (R/W) = 632.0 50h (R/W) = 640.0 51h (R/W) = 648.0 52h (R/W) = 656.0 53h (R/W) = 664.0 54h (R/W) = 672.0 55h (R/W) = 680.0 56h (R/W) = 688.0 57h (R/W) = 696.0 58h (R/W) = 704.0 59h (R/W) = 712.0 5Ah (R/W) = 720.0 5Bh (R/W) = 728.0 5Ch (R/W) = 736.0 5Dh (R/W) = 744.0 5Eh (R/W) = 752.0 5Fh (R/W) = 760.0 60h (R/W) = 768.0 61h (R/W) = 776.0 62h (R/W) = 784.0 63h (R/W) = 792.0 64h (R/W) = 800.0 65h (R/W) = 808.0 66h (R/W) = 816.0 67h (R/W) = 824.0 68h (R/W) = 832.0 69h (R/W) = 840.0 6Ah (R/W) = 848.0 6Bh (R/W) = 856.0 6Ch (R/W) = 864.0 6Dh (R/W) = 872.0 6Eh (R/W) = 880.0 6Fh (R/W) = 888.0 70h (R/W) = 896.0 71h (R/W) = 904.0 72h (R/W) = 912.0 73h (R/W) = 920.0 74h (R/W) = 928.0 75h (R/W) = 936.0 76h (R/W) = 944.0 77h (R/W) = 952.0 78h (R/W) = 960.0 79h (R/W) = 968.0 7Ah (R/W) = 976.0 7Bh (R/W) = 984.0 7Ch (R/W) = 992.0 7Dh (R/W) = 1000.0 7Eh (R/W) = 1008.0 7Fh (R/W) = 1016.0

Table 38-25. USBRXFIFOADD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				80h (R/W) = 1024.0 81h (R/W) = 1032.0 82h (R/W) = 1040.0 83h (R/W) = 1048.0 84h (R/W) = 1056.0 85h (R/W) = 1064.0 86h (R/W) = 1072.0 87h (R/W) = 1080.0 88h (R/W) = 1088.0 89h (R/W) = 1096.0 8Ah (R/W) = 1104.0 8Bh (R/W) = 1112.0 8Ch (R/W) = 1120.0 8Dh (R/W) = 1128.0 8Eh (R/W) = 1136.0 8Fh (R/W) = 1144.0 90h (R/W) = 1152.0 91h (R/W) = 1160.0 92h (R/W) = 1168.0 93h (R/W) = 1176.0 94h (R/W) = 1184.0 95h (R/W) = 1192.0 96h (R/W) = 1200.0 97h (R/W) = 1208.0 98h (R/W) = 1216.0 99h (R/W) = 1224.0 9Ah (R/W) = 1232.0 9Bh (R/W) = 1240.0 9Ch (R/W) = 1248.0 9Dh (R/W) = 1256.0 9Eh (R/W) = 1264.0 9Fh (R/W) = 1272.0 A0h (R/W) = 1280.0 A1h (R/W) = 1288.0 A2h (R/W) = 1296.0 A3h (R/W) = 1304.0 A4h (R/W) = 1312.0 A5h (R/W) = 1320.0 A6h (R/W) = 1328.0 A7h (R/W) = 1336.0 A8h (R/W) = 1344.0 A9h (R/W) = 1352.0 AAh (R/W) = 1360.0 ABh (R/W) = 1368.0 ACh (R/W) = 1376.0 ADh (R/W) = 1384.0 AEh (R/W) = 1392.0 AFh (R/W) = 1400.0 B0h (R/W) = 1408.0 B1h (R/W) = 1416.0 B2h (R/W) = 1424.0 B3h (R/W) = 1432.0 B4h (R/W) = 1440.0 B5h (R/W) = 1448.0 B6h (R/W) = 1456.0 B7h (R/W) = 1464.0 B8h (R/W) = 1472.0 B9h (R/W) = 1480.0 BAh (R/W) = 1488.0 BBh (R/W) = 1496.0 BCh (R/W) = 1504.0 BDh (R/W) = 1512.0 BEh (R/W) = 1520.0 BFh (R/W) = 1528.0 C0h (R/W) = 1536.0

Table 38-25. USBRXFIFOADD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				C1h (R/W) = 1544.0 C2h (R/W) = 1552.0 C3h (R/W) = 1560.0 C4h (R/W) = 1568.0 C5h (R/W) = 1576.0 C6h (R/W) = 1584.0 C7h (R/W) = 1592.0 C8h (R/W) = 1600.0 C9h (R/W) = 1608.0 CAh (R/W) = 1616.0 CBh (R/W) = 1624.0 CCh (R/W) = 1632.0 CDh (R/W) = 1640.0 CEh (R/W) = 1648.0 CFh (R/W) = 1656.0 D0h (R/W) = 1664.0 D1h (R/W) = 1672.0 D2h (R/W) = 1680.0 D3h (R/W) = 1688.0 D4h (R/W) = 1696.0 D5h (R/W) = 1704.0 D6h (R/W) = 1712.0 D7h (R/W) = 1720.0 D8h (R/W) = 1728.0 D9h (R/W) = 1736.0 DAh (R/W) = 1744.0 DBh (R/W) = 1752.0 DCh (R/W) = 1760.0 DDh (R/W) = 1768.0 DEh (R/W) = 1776.0 DFh (R/W) = 1784.0 E0h (R/W) = 1792.0 E1h (R/W) = 1800.0 E2h (R/W) = 1808.0 E3h (R/W) = 1816.0 E4h (R/W) = 1824.0 E5h (R/W) = 1832.0 E6h (R/W) = 1840.0 E7h (R/W) = 1848.0 E8h (R/W) = 1856.0 E9h (R/W) = 1864.0 EAh (R/W) = 1872.0 EBh (R/W) = 1880.0 ECh (R/W) = 1888.0 EDh (R/W) = 1896.0 EEh (R/W) = 1904.0 EFh (R/W) = 1912.0 F0h (R/W) = 1920.0 F1h (R/W) = 1928.0 F2h (R/W) = 1936.0 F3h (R/W) = 1944.0 F4h (R/W) = 1952.0 F5h (R/W) = 1960.0 F6h (R/W) = 1968.0 F7h (R/W) = 1976.0 F8h (R/W) = 1984.0 F9h (R/W) = 1992.0 FAh (R/W) = 2000.0 FBh (R/W) = 2008.0 FCh (R/W) = 2016.0 FDh (R/W) = 2024.0 FEh (R/W) = 2032.0 FFh (R/W) = 2040.0 100h (R/W) = 2048.0 101h (R/W) = 2056.0

Table 38-25. USBRXFIFOADD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				102h (R/W) = 2064.0 103h (R/W) = 2072.0 104h (R/W) = 2080.0 105h (R/W) = 2088.0 106h (R/W) = 2096.0 107h (R/W) = 2104.0 108h (R/W) = 2112.0 109h (R/W) = 2120.0 10Ah (R/W) = 2128.0 10Bh (R/W) = 2136.0 10Ch (R/W) = 2144.0 10Dh (R/W) = 2152.0 10Eh (R/W) = 2160.0 10Fh (R/W) = 2168.0 110h (R/W) = 2176.0 111h (R/W) = 2184.0 112h (R/W) = 2192.0 113h (R/W) = 2200.0 114h (R/W) = 2208.0 115h (R/W) = 2216.0 116h (R/W) = 2224.0 117h (R/W) = 2232.0 118h (R/W) = 2240.0 119h (R/W) = 2248.0 11Ah (R/W) = 2256.0 11Bh (R/W) = 2264.0 11Ch (R/W) = 2272.0 11Dh (R/W) = 2280.0 11Eh (R/W) = 2288.0 11Fh (R/W) = 2296.0 120h (R/W) = 2304.0 121h (R/W) = 2312.0 122h (R/W) = 2320.0 123h (R/W) = 2328.0 124h (R/W) = 2336.0 125h (R/W) = 2344.0 126h (R/W) = 2352.0 127h (R/W) = 2360.0 128h (R/W) = 2368.0 129h (R/W) = 2376.0 12Ah (R/W) = 2384.0 12Bh (R/W) = 2392.0 12Ch (R/W) = 2400.0 12Dh (R/W) = 2408.0 12Eh (R/W) = 2416.0 12Fh (R/W) = 2424.0 130h (R/W) = 2432.0 131h (R/W) = 2440.0 132h (R/W) = 2448.0 133h (R/W) = 2456.0 134h (R/W) = 2464.0 135h (R/W) = 2472.0 136h (R/W) = 2480.0 137h (R/W) = 2488.0 138h (R/W) = 2496.0 139h (R/W) = 2504.0 13Ah (R/W) = 2512.0 13Bh (R/W) = 2520.0 13Ch (R/W) = 2528.0 13Dh (R/W) = 2536.0 13Eh (R/W) = 2544.0 13Fh (R/W) = 2552.0 140h (R/W) = 2560.0 141h (R/W) = 2568.0 142h (R/W) = 2576.0

Table 38-25. USBRXFIFOADD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				143h (R/W) = 2584.0 144h (R/W) = 2592.0 145h (R/W) = 2600.0 146h (R/W) = 2608.0 147h (R/W) = 2616.0 148h (R/W) = 2624.0 149h (R/W) = 2632.0 14Ah (R/W) = 2640.0 14Bh (R/W) = 2648.0 14Ch (R/W) = 2656.0 14Dh (R/W) = 2664.0 14Eh (R/W) = 2672.0 14Fh (R/W) = 2680.0 150h (R/W) = 2688.0 151h (R/W) = 2696.0 152h (R/W) = 2704.0 153h (R/W) = 2712.0 154h (R/W) = 2720.0 155h (R/W) = 2728.0 156h (R/W) = 2736.0 157h (R/W) = 2744.0 158h (R/W) = 2752.0 159h (R/W) = 2760.0 15Ah (R/W) = 2768.0 15Bh (R/W) = 2776.0 15Ch (R/W) = 2784.0 15Dh (R/W) = 2792.0 15Eh (R/W) = 2800.0 15Fh (R/W) = 2808.0 160h (R/W) = 2816.0 161h (R/W) = 2824.0 162h (R/W) = 2832.0 163h (R/W) = 2840.0 164h (R/W) = 2848.0 165h (R/W) = 2856.0 166h (R/W) = 2864.0 167h (R/W) = 2872.0 168h (R/W) = 2880.0 169h (R/W) = 2888.0 16Ah (R/W) = 2896.0 16Bh (R/W) = 2904.0 16Ch (R/W) = 2912.0 16Dh (R/W) = 2920.0 16Eh (R/W) = 2928.0 16Fh (R/W) = 2936.0 170h (R/W) = 2944.0 171h (R/W) = 2952.0 172h (R/W) = 2960.0 173h (R/W) = 2968.0 174h (R/W) = 2976.0 175h (R/W) = 2984.0 176h (R/W) = 2992.0 177h (R/W) = 3000.0 178h (R/W) = 3008.0 179h (R/W) = 3016.0 17Ah (R/W) = 3024.0 17Bh (R/W) = 3032.0 17Ch (R/W) = 3040.0 17Dh (R/W) = 3048.0 17Eh (R/W) = 3056.0 17Fh (R/W) = 3064.0 180h (R/W) = 3072.0 181h (R/W) = 3080.0 182h (R/W) = 3088.0 183h (R/W) = 3096.0

Table 38-25. USBRXFIFOADD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				184h (R/W) = 3104.0 185h (R/W) = 3112.0 186h (R/W) = 3120.0 187h (R/W) = 3128.0 188h (R/W) = 3136.0 189h (R/W) = 3144.0 18Ah (R/W) = 3152.0 18Bh (R/W) = 3160.0 18Ch (R/W) = 3168.0 18Dh (R/W) = 3176.0 18Eh (R/W) = 3184.0 18Fh (R/W) = 3192.0 190h (R/W) = 3200.0 191h (R/W) = 3208.0 192h (R/W) = 3216.0 193h (R/W) = 3224.0 194h (R/W) = 3232.0 195h (R/W) = 3240.0 196h (R/W) = 3248.0 197h (R/W) = 3256.0 198h (R/W) = 3264.0 199h (R/W) = 3272.0 19Ah (R/W) = 3280.0 19Bh (R/W) = 3288.0 19Ch (R/W) = 3296.0 19Dh (R/W) = 3304.0 19Eh (R/W) = 3312.0 19Fh (R/W) = 3320.0 1A0h (R/W) = 3328.0 1A1h (R/W) = 3336.0 1A2h (R/W) = 3344.0 1A3h (R/W) = 3352.0 1A4h (R/W) = 3360.0 1A5h (R/W) = 3368.0 1A6h (R/W) = 3376.0 1A7h (R/W) = 3384.0 1A8h (R/W) = 3392.0 1A9h (R/W) = 3400.0 1AAh (R/W) = 3408.0 1ABh (R/W) = 3416.0 1ACh (R/W) = 3424.0 1ADh (R/W) = 3432.0 1AEh (R/W) = 3440.0 1AFh (R/W) = 3448.0 1B0h (R/W) = 3456.0 1B1h (R/W) = 3464.0 1B2h (R/W) = 3472.0 1B3h (R/W) = 3480.0 1B4h (R/W) = 3488.0 1B5h (R/W) = 3496.0 1B6h (R/W) = 3504.0 1B7h (R/W) = 3512.0 1B8h (R/W) = 3520.0 1B9h (R/W) = 3528.0 1BAh (R/W) = 3536.0 1BBh (R/W) = 3544.0 1BCh (R/W) = 3552.0 1BDh (R/W) = 3560.0 1BEh (R/W) = 3568.0 1BFh (R/W) = 3576.0 1C0h (R/W) = 3584.0 1C1h (R/W) = 3592.0 1C2h (R/W) = 3600.0 1C3h (R/W) = 3608.0 1C4h (R/W) = 3616.0

Table 38-25. USBRXFIFOADD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				1C5h (R/W) = 3624.0 1C6h (R/W) = 3632.0 1C7h (R/W) = 3640.0 1C8h (R/W) = 3648.0 1C9h (R/W) = 3656.0 1CAh (R/W) = 3664.0 1CBh (R/W) = 3672.0 1CCh (R/W) = 3680.0 1CDh (R/W) = 3688.0 1CEh (R/W) = 3696.0 1CFh (R/W) = 3704.0 1D0h (R/W) = 3712.0 1D1h (R/W) = 3720.0 1D2h (R/W) = 3728.0 1D3h (R/W) = 3736.0 1D4h (R/W) = 3744.0 1D5h (R/W) = 3752.0 1D6h (R/W) = 3760.0 1D7h (R/W) = 3768.0 1D8h (R/W) = 3776.0 1D9h (R/W) = 3784.0 1DAh (R/W) = 3792.0 1DBh (R/W) = 3800.0 1DCh (R/W) = 3808.0 1DDh (R/W) = 3816.0 1DEh (R/W) = 3824.0 1DFh (R/W) = 3832.0 1E0h (R/W) = 3840.0 1E1h (R/W) = 3848.0 1E2h (R/W) = 3856.0 1E3h (R/W) = 3864.0 1E4h (R/W) = 3872.0 1E5h (R/W) = 3880.0 1E6h (R/W) = 3888.0 1E7h (R/W) = 3896.0 1E8h (R/W) = 3904.0 1E9h (R/W) = 3912.0 1EAh (R/W) = 3920.0 1EBh (R/W) = 3928.0 1ECh (R/W) = 3936.0 1EDh (R/W) = 3944.0 1EEh (R/W) = 3952.0 1EFh (R/W) = 3960.0 1F0h (R/W) = 3968.0 1F1h (R/W) = 3976.0 1F2h (R/W) = 3984.0 1F3h (R/W) = 3992.0 1F4h (R/W) = 4000.0 1F5h (R/W) = 4008.0 1F6h (R/W) = 4016.0 1F7h (R/W) = 4024.0 1F8h (R/W) = 4032.0 1F9h (R/W) = 4040.0 1FAh (R/W) = 4048.0 1FBh (R/W) = 4056.0 1FCh (R/W) = 4064.0 1FDh (R/W) = 4072.0 1FEh (R/W) = 4080.0 1FFh (R/W) = 4088.0 200h (R/W) = 4095.0

38.6.2.21 USBCONTIM Register (Offset = 3Dh) [Reset = 11h]

USBCONTIM is shown in Figure 38-23 and described in Table 38-26.

Return to the Summary Table.

USB Connect Timing

Figure 38-23. USBCONTIM Register

7	6	5	4	3	2	1	0
WTCON				WTID			
R/W-1h				R/W-1h			

Table 38-26. USBCONTIM Register Field Descriptions

Bit	Field	Type	Reset	Description
7-4	WTCON	R/W	1h	The wait ID field configures the delay required from the enable of the ID detection to when the ID value is valid, in units of 4.369 ms. The default corresponds to 52.43 ms. Reset type: SYSRSn
3-0	WTID	R/W	1h	The connect wait field configures the wait required to allow for the user's connect/disconnect filter, in units of 533.3 ns. The default corresponds to 2.667 us. Reset type: SYSRSn

38.6.2.22 USBFSEOF Register (Offset = 3Eh) [Reset = 77h]

USBFSEOF is shown in Figure 38-24 and described in Table 38-27.

Return to the Summary Table.

USB Full-Speed Last Transaction to End of Frame Timing

Figure 38-24. USBFSEOF Register

7	6	5	4	3	2	1	0
FSEOFG							
R/W-77h							

Table 38-27. USBFSEOF Register Field Descriptions

Bit	Field	Type	Reset	Description
7-0	FSEOFG	R/W	77h	The full-speed end-of-frame gap field is used during full-speed transactions to configure the gap between the last transaction and the End-of-Frame (EOF), in units of 533.3 ns. The default corresponds to 63.46 us. Reset type: SYSRSn

38.6.2.23 USBLSEOF Register (Offset = 3Fh) [Reset = 72h]

USBLSEOF is shown in Figure 38-25 and described in Table 38-28.

Return to the Summary Table.

USB Low-Speed Last Transaction to End of Frame Timing

Figure 38-25. USBLSEOF Register

7	6	5	4	3	2	1	0
LSEOFG							
R/W-72h							

Table 38-28. USBLSEOF Register Field Descriptions

Bit	Field	Type	Reset	Description
7-0	LSEOFG	R/W	72h	The low-speed end-of-frame gap field is used during low-speed transactions to set the gap between the last transaction and the End-of-Frame (EOF), in units of 1.067 us. The default corresponds to 121.6 us. Reset type: SYSRSn

38.6.2.24 USBTXFUNCADDR0 Register (Offset = 40h) [Reset = 0h]

USBTXFUNCADDR0 is shown in Figure 38-26 and described in Table 38-29.

Return to the Summary Table.

USB Transmit Functional Address Endpoint 0

Figure 38-26. USBTXFUNCADDR0 Register

7	6	5	4	3	2	1	0
RESERVED	ADDR						
R-0h	R/W-0h						

Table 38-29. USBTXFUNCADDR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	ADDR	R/W	0h	Device Address specifies the USB bus address for the target Device. Reset type: SYSRSn

38.6.2.25 USBTXHUBADDR0 Register (Offset = 41h) [Reset = 0h]

USBTXHUBADDR0 is shown in Figure 38-27 and described in Table 38-30.

Return to the Summary Table.

USB Transmit Hub Address Endpoint 0

Figure 38-27. USBTXHUBADDR0 Register

7	6	5	4	3	2	1	0
RESERVED	ADDR						
R-0h	R/W-0h						

Table 38-30. USBTXHUBADDR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	ADDR	R/W	0h	Device Address specifies the USB bus address for the target Device. Reset type: SYSRSn

38.6.2.26 USBTXHUBPORT0 Register (Offset = 41h) [Reset = 0h]

USBTXHUBPORT0 is shown in Figure 38-28 and described in Table 38-31.

Return to the Summary Table.

USB Transmit Hub Port Endpoint 0

Figure 38-28. USBTXHUBPORT0 Register

7	6	5	4	3	2	1	0
RESERVED	ADDR						
R-0h	R/W-0h						

Table 38-31. USBTXHUBPORT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	ADDR	R/W	0h	Hub Port specifies the USB hub port number. Reset type: SYSRSn

38.6.2.27 USBTXFUNCADDR1 Register (Offset = 44h) [Reset = 0h]

USBTXFUNCADDR1 is shown in Figure 38-29 and described in Table 38-32.

Return to the Summary Table.

USB Transmit Functional Address Endpoint 1

Figure 38-29. USBTXFUNCADDR1 Register

7	6	5	4	3	2	1	0
RESERVED	ADDR						
R-0h	R/W-0h						

Table 38-32. USBTXFUNCADDR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	ADDR	R/W	0h	Device Address specifies the USB bus address for the target Device. Reset type: SYSRSn

38.6.2.28 USBTXHUBADDR1 Register (Offset = 45h) [Reset = 0h]

USBTXHUBADDR1 is shown in Figure 38-30 and described in Table 38-33.

Return to the Summary Table.

USB Transmit Hub Address Endpoint 1

Figure 38-30. USBTXHUBADDR1 Register

7	6	5	4	3	2	1	0
RESERVED	ADDR						
R-0h	R/W-0h						

Table 38-33. USBTXHUBADDR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	ADDR	R/W	0h	Device Address specifies the USB bus address for the target Device. Reset type: SYSRSn

38.6.2.29 USBTXHUBPORT1 Register (Offset = 45h) [Reset = 0h]

USBTXHUBPORT1 is shown in Figure 38-31 and described in Table 38-34.

Return to the Summary Table.

USB Transmit Hub Port Endpoint 1

Figure 38-31. USBTXHUBPORT1 Register

7	6	5	4	3	2	1	0
RESERVED	ADDR						
R-0h	R/W-0h						

Table 38-34. USBTXHUBPORT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	ADDR	R/W	0h	Hub Port specifies the USB hub port number. Reset type: SYSRSn

38.6.2.30 USBRXFUNCADDR1 Register (Offset = 46h) [Reset = 0h]

USBRXFUNCADDR1 is shown in Figure 38-32 and described in Table 38-35.

Return to the Summary Table.

USB Receive Functional Address Endpoint 1

Figure 38-32. USBRXFUNCADDR1 Register

7	6	5	4	3	2	1	0
RESERVED	ADDR						
R-0h	R/W-0h						

Table 38-35. USBRXFUNCADDR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	ADDR	R/W	0h	Device Address specifies the USB bus address for the target Device. Reset type: SYSRSn

38.6.2.31 USBRXHUBADDR1 Register (Offset = 47h) [Reset = 0h]

USBRXHUBADDR1 is shown in Figure 38-33 and described in Table 38-36.

Return to the Summary Table.

USB Receive Hub Address Endpoint 1

Figure 38-33. USBRXHUBADDR1 Register

7	6	5	4	3	2	1	0
MULTTRAN	ADDR						
R/W-0h	R/W-0h						

Table 38-36. USBRXHUBADDR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	MULTTRAN	R/W	0h	Hub has Multiple Translators Reset type: SYSRSn 0h (R/W) = Clear to indicate that the hub has a single transaction translator. 1h (R/W) = Set to indicate that the hub has multiple transaction translators.
6-0	ADDR	R/W	0h	Hub Address Reset type: SYSRSn

38.6.2.32 USBRXHUBPORT1 Register (Offset = 47h) [Reset = 0h]

USBRXHUBPORT1 is shown in Figure 38-34 and described in Table 38-37.

Return to the Summary Table.

USB Receive Hub Port Endpoint 1

Figure 38-34. USBRXHUBPORT1 Register

7	6	5	4	3	2	1	0
RESERVED	ADDR						
R-0h	R/W-0h						

Table 38-37. USBRXHUBPORT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	ADDR	R/W	0h	Hub Address Reset type: SYSRSn

38.6.2.33 USBTXFUNCADDR2 Register (Offset = 48h) [Reset = 0h]

USBTXFUNCADDR2 is shown in Figure 38-35 and described in Table 38-38.

Return to the Summary Table.

USB Transmit Functional Address Endpoint 2

Figure 38-35. USBTXFUNCADDR2 Register

7	6	5	4	3	2	1	0
RESERVED	ADDR						
R-0h	R/W-0h						

Table 38-38. USBTXFUNCADDR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	ADDR	R/W	0h	Device Address specifies the USB bus address for the target Device. Reset type: SYSRSn

38.6.2.34 USBTXHUBADDR2 Register (Offset = 49h) [Reset = 0h]

USBTXHUBADDR2 is shown in Figure 38-36 and described in Table 38-39.

Return to the Summary Table.

USB Transmit Hub Address Endpoint 2

Figure 38-36. USBTXHUBADDR2 Register

7	6	5	4	3	2	1	0
RESERVED	ADDR						
R-0h	R/W-0h						

Table 38-39. USBTXHUBADDR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	ADDR	R/W	0h	Device Address specifies the USB bus address for the target Device. Reset type: SYSRSn

38.6.2.35 USBTXHUBPORT2 Register (Offset = 49h) [Reset = 0h]

USBTXHUBPORT2 is shown in Figure 38-37 and described in Table 38-40.

Return to the Summary Table.

USB Transmit Hub Port Endpoint 2

Figure 38-37. USBTXHUBPORT2 Register

7	6	5	4	3	2	1	0
RESERVED	ADDR						
R-0h	R/W-0h						

Table 38-40. USBTXHUBPORT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	ADDR	R/W	0h	Hub Port specifies the USB hub port number. Reset type: SYSRSn

38.6.2.36 USBRXFUNCADDR2 Register (Offset = 4Ah) [Reset = 0h]

USBRXFUNCADDR2 is shown in Figure 38-38 and described in Table 38-41.

Return to the Summary Table.

USB Receive Functional Address Endpoint 2

Figure 38-38. USBRXFUNCADDR2 Register

7	6	5	4	3	2	1	0
RESERVED	ADDR						
R-0h	R/W-0h						

Table 38-41. USBRXFUNCADDR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	ADDR	R/W	0h	Device Address specifies the USB bus address for the target Device. Reset type: SYSRSn

38.6.2.37 USBRXHUBADDR2 Register (Offset = 4Bh) [Reset = 0h]

USBRXHUBADDR2 is shown in Figure 38-39 and described in Table 38-42.

Return to the Summary Table.

USB Receive Hub Address Endpoint 2

Figure 38-39. USBRXHUBADDR2 Register

7	6	5	4	3	2	1	0
MULTTRAN	ADDR						
R/W-0h	R/W-0h						

Table 38-42. USBRXHUBADDR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	MULTTRAN	R/W	0h	Hub has Multiple Translators Reset type: SYSRSn 0h (R/W) = Clear to indicate that the hub has a single transaction translator. 1h (R/W) = Set to indicate that the hub has multiple transaction translators.
6-0	ADDR	R/W	0h	Hub Address Reset type: SYSRSn

38.6.2.38 USBRXHUBPORT2 Register (Offset = 4Bh) [Reset = 0h]

USBRXHUBPORT2 is shown in Figure 38-40 and described in Table 38-43.

Return to the Summary Table.

USB Receive Hub Port Endpoint 2

Figure 38-40. USBRXHUBPORT2 Register

7	6	5	4	3	2	1	0
RESERVED	ADDR						
R-0h	R/W-0h						

Table 38-43. USBRXHUBPORT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	ADDR	R/W	0h	Hub Address Reset type: SYSRSn

38.6.2.39 USBTXFUNCADDR3 Register (Offset = 4Ch) [Reset = 0h]

USBTXFUNCADDR3 is shown in Figure 38-41 and described in Table 38-44.

Return to the Summary Table.

USB Transmit Functional Address Endpoint 3

Figure 38-41. USBTXFUNCADDR3 Register

7	6	5	4	3	2	1	0
RESERVED	ADDR						
R-0h	R/W-0h						

Table 38-44. USBTXFUNCADDR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	ADDR	R/W	0h	Device Address specifies the USB bus address for the target Device. Reset type: SYSRSn

38.6.2.40 USBTXHUBADDR3 Register (Offset = 4Dh) [Reset = 0h]

USBTXHUBADDR3 is shown in Figure 38-42 and described in Table 38-45.

Return to the Summary Table.

USB Transmit Hub Address Endpoint 3

Figure 38-42. USBTXHUBADDR3 Register

7	6	5	4	3	2	1	0
RESERVED	ADDR						
R-0h	R/W-0h						

Table 38-45. USBTXHUBADDR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	ADDR	R/W	0h	Device Address specifies the USB bus address for the target Device. Reset type: SYSRSn

38.6.2.41 USBTXHUBPORT3 Register (Offset = 4Dh) [Reset = 0h]

USBTXHUBPORT3 is shown in Figure 38-43 and described in Table 38-46.

Return to the Summary Table.

USB Transmit Hub Port Endpoint 3

Figure 38-43. USBTXHUBPORT3 Register

7	6	5	4	3	2	1	0
RESERVED	ADDR						
R-0h	R/W-0h						

Table 38-46. USBTXHUBPORT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	ADDR	R/W	0h	Hub Port specifies the USB hub port number. Reset type: SYSRSn

38.6.2.42 USBRXFUNCADDR3 Register (Offset = 4Eh) [Reset = 0h]

USBRXFUNCADDR3 is shown in Figure 38-44 and described in Table 38-47.

Return to the Summary Table.

USB Receive Functional Address Endpoint 3

Figure 38-44. USBRXFUNCADDR3 Register

7	6	5	4	3	2	1	0
RESERVED	ADDR						
R-0h	R/W-0h						

Table 38-47. USBRXFUNCADDR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	ADDR	R/W	0h	Device Address specifies the USB bus address for the target Device. Reset type: SYSRSn

38.6.2.43 USBRXHUBADDR3 Register (Offset = 4Fh) [Reset = 0h]

USBRXHUBADDR3 is shown in Figure 38-45 and described in Table 38-48.

Return to the Summary Table.

USB Receive Hub Address Endpoint 3

Figure 38-45. USBRXHUBADDR3 Register

7	6	5	4	3	2	1	0
MULTTRAN	ADDR						
R/W-0h	R/W-0h						

Table 38-48. USBRXHUBADDR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	MULTTRAN	R/W	0h	Hub has Multiple Translators Reset type: SYSRSn 0h (R/W) = Clear to indicate that the hub has a single transaction translator. 1h (R/W) = Set to indicate that the hub has multiple transaction translators.
6-0	ADDR	R/W	0h	Hub Address Reset type: SYSRSn

38.6.2.44 USBRXHUBPORT3 Register (Offset = 4Fh) [Reset = 0h]

USBRXHUBPORT3 is shown in Figure 38-46 and described in Table 38-49.

Return to the Summary Table.

USB Receive Hub Port Endpoint 3

Figure 38-46. USBRXHUBPORT3 Register

7	6	5	4	3	2	1	0
RESERVED	ADDR						
R-0h	R/W-0h						

Table 38-49. USBRXHUBPORT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	ADDR	R/W	0h	Hub Address Reset type: SYSRSn

38.6.2.45 USBCSRL0 Register (Offset = 81h) [Reset = 0h]

USBCSRL0 is shown in Figure 38-47 and described in Table 38-50.

Return to the Summary Table.

USB Control and Status Endpoint 0 Low

Figure 38-47. USBCSRL0 Register

7	6	5	4	3	2	1	0
SETENDC_NAKTO	RXRDYC_STATUS	STALL_RQPKT	SETEND_ERROR	DATAEND_SETUP	STALLED	TXRDY	RXRDY
W1C-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 38-50. USBCSRL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	SETENDC_NAKTO	W1C	0h	NAK Timeout. Software must clear this bit to allow the endpoint to continue. Reset type: SYSRSn 0h (R/W) = No timeout 1h (R/W) = Indicates that endpoint 0 is halted following the receipt of NAK responses for longer than the time set by the USBNAKLMT register
6	RXRDYC_STATUS	R/W	0h	Status Packet. Setting this bit ensures that the DT bit is set in the USBCSRH0 register so that a DATA1 packet is used for the STATUS stage transaction. Reset type: SYSRSn 0h (R/W) = No transaction 1h (R/W) = The Endpoint 1 transmit interrupt is asserted.
5	STALL_RQPKT	R/W	0h	Request Packet. This bit is cleared when the RXRDY bit is set. Reset type: SYSRSn 0h (R/W) = No request 1h (R/W) = Initiates a STATUS stage transaction. This bit must be set at the same time as the TXRDY or REQPKT bit is set. This bit is automatically cleared when the STATUS stage is over.
4	SETEND_ERROR	R/W	0h	Error. Software must clear this bit. Reset type: SYSRSn 0h (R/W) = No error 1h (R/W) = Three attempts have been made to perform a transaction with no response from the peripheral. The EP0 bit in the USBTXIS register is also set in this situation.
3	DATAEND_SETUP	R/W	0h	Setup Packet. Setting this bit always clears the DT bit in the USBCSRH0 register to send a DATA0 packet. Reset type: SYSRSn 0h (R/W) = Sends an OUT token. 1h (R/W) = Sends a SETUP token instead of an OUT token for the transaction. This bit should be set at the same time as the TXRDY bit is set.
2	STALLED	R/W	0h	Endpoint Stalled. Software must clear this bit. Reset type: SYSRSn 0h (R/W) = No handshake has been received. 1h (R/W) = A STALL handshake has been received
1	TXRDY	R/W	0h	Transmit Packet Ready. If both the TXRDY and SETUP bits are set, a setup packet is sent. If just TXRDY is set, an OUT packet is sent. Reset type: SYSRSn 0h (R/W) = No transmit packet is ready. 1h (R/W) = Software sets this bit after loading a data packet into the TX FIFO. The EP0 bit in the USBTXIS register is also set in this situation.

Table 38-50. USBCSRL0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	RXRDY	R/W	0h	Receive Packet Ready. Software must clear this bit after he packet has been read from the FIFO to acknowledge that the data has been read from the FIFO. Reset type: SYSRSn 0h (R/W) = No receive packet has been received. 1h (R/W) = Indicates that a data packet has been received in the RX FIFO. The EP0 bit in the USBTXIS register is also set in this situation.

38.6.2.46 USBCSRH0 Register (Offset = 81h) [Reset = 0h]

USBCSRH0 is shown in Figure 38-48 and described in Table 38-51.

Return to the Summary Table.

USB Control and Status Endpoint 0 High

Figure 38-48. USBCSRH0 Register

7	6	5	4	3	2	1	0
RESERVED					DTWE	DT	FLUSH
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 38-51. USBCSRH0 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-3	RESERVED	R	0h	Reserved
2	DTWE	R/W	0h	Data Toggle Write Enable. This bit is automatically cleared once the new value is written. Reset type: SYSRSn 0h (R/W) = The DT bit cannot be written. 1h (R/W) = Enables the current state of the endpoint 0 data toggle to be written (see DT bit).
1	DT	R/W	0h	Data Toggle. When read, this bit indicates the current state of the endpoint 0 data toggle. If DTWE is set, this bit may be written with the required setting of the data toggle. If DTWE is Low, this bit cannot be written. Care should be taken when writing to this bit as it should only be changed to RESET USB endpoint 0. Reset type: SYSRSn
0	FLUSH	R/W	0h	Flush FIFO. This bit is automatically cleared after the flush is performed. Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Flushes the next packet to be transmitted/read from the endpoint 0 FIFO. The FIFO pointer is reset and the TXRDY/RXRDY bit is cleared. Note: This bit should only be set when TXRDY/RXRDY is set. At other times, it may cause data to be corrupted..

38.6.2.47 USBCOUNT0 Register (Offset = 84h) [Reset = 0h]

USBCOUNT0 is shown in Figure 38-49 and described in Table 38-52.

Return to the Summary Table.

USB Receive Byte Count Endpoint 0

Figure 38-49. USBCOUNT0 Register

7	6	5	4	3	2	1	0
RESERVED	COUNT						
R-0h	R/W-0h						

Table 38-52. USBCOUNT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	Reserved
6-0	COUNT	R/W	0h	FIFO Count. COUNT is a read-only value that indicates the number of received data bytes in the endpoint 0 FIFO. Reset type: SYSRSn

38.6.2.48 USBTYPE0 Register (Offset = 85h) [Reset = 0h]

USBTYPE0 is shown in Figure 38-50 and described in Table 38-53.

Return to the Summary Table.

USB Type Endpoint 0

Figure 38-50. USBTYPE0 Register

7	6	5	4	3	2	1	0
SPEED		RESERVED					
R/W-0h		R-0h					

Table 38-53. USBTYPE0 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-6	SPEED	R/W	0h	Operating Speed specifies the operating speed of the target Device. If selected, the target is assumed to have the same connection speed as the USB controller. Reset type: SYSRSn 0h (R/W) = Reserved 1h (R/W) = Reserved 2h (R/W) = Full 3h (R/W) = Low
5-0	RESERVED	R	0h	Reserved

38.6.2.49 USBNAKLMT Register (Offset = 85h) [Reset = 0h]

USBNAKLMT is shown in Figure 38-51 and described in Table 38-54.

Return to the Summary Table.

USB NAK Limit

Figure 38-51. USBNAKLMT Register

7	6	5	4	3	2	1	0
RESERVED			NAKLMT				
R-0h			R/W-0h				

Table 38-54. USBNAKLMT Register Field Descriptions

Bit	Field	Type	Reset	Description
7-5	RESERVED	R	0h	Reserved
4-0	NAKLMT	R/W	0h	EP0 NAK Limit specifies the number of frames after receiving a stream of NAK responses. Reset type: SYSRSn

38.6.2.50 USBTXMAXP1 Register (Offset = 88h) [Reset = 0h]

USBTXMAXP1 is shown in Figure 38-52 and described in Table 38-55.

Return to the Summary Table.

USB Maximum Transmit Data Endpoint 1

Figure 38-52. USBTXMAXP1 Register

15	14	13	12	11	10	9	8
RESERVED					MAXLOAD		
R-0h					R/W-0h		
7	6	5	4	3	2	1	0
MAXLOAD							
R/W-0h							

Table 38-55. USBTXMAXP1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	Reserved
10-0	MAXLOAD	R/W	0h	Maximum Payload specifies the maximum payload in bytes per transaction. Reset type: SYSRSn

38.6.2.51 USBTXCSRL1 Register (Offset = 89h) [Reset = 0h]

USBTXCSRL1 is shown in Figure 38-53 and described in Table 38-56.

Return to the Summary Table.

USB Transmit Control and Status Endpoint 1 Low

Figure 38-53. USBTXCSRL1 Register

7	6	5	4	3	2	1	0
NAKTO	CLRDT	STALLED	STALL_SETUP	FLUSH	UNDRN_ERROR1	FIFONE	TXRDY
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 38-56. USBTXCSRL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	NAKTO	R/W	0h	NAK Timeout. Software must clear this bit to allow the endpoint to continue. Reset type: SYSRSn 0h (R/W) = No timeout 1h (R/W) = Bulk endpoints only: Indicates that the transmit endpoint is halted following the receipt of NAK responses for longer than the time set by the NAKLMT field in the USBTXINTERVAL[n] register.
6	CLRDT	R/W	0h	Clear DataToggle Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Writing a 1 to this bit clears the DT bit in the USBTXCSRH[n] register.
5	STALLED	R/W	0h	Endpoint Stalled. Software must clear this bit. Reset type: SYSRSn 0h (R/W) = A STALL handshake has not been received 1h (R/W) = Indicates that a STALL handshake has been received. When this bit is set, any DMA request that is in progress is stopped, the FIFO is completely flushed, and the TXRDY bit is cleared.
4	STALL_SETUP	R/W	0h	Setup Packet. Reset type: SYSRSn 0h (R/W) = No SETUP token is sent. 1h (R/W) = Sends a SETUP token instead of an OUT token for the transaction. This bit should be set at the same time as the TXRDY bit is set. Note: Setting this bit also clears the DT bit in the USBTXCSRH[n] register.
3	FLUSH	R/W	0h	Flush FIFO. This bit can be set simultaneously with the TXRDY bit to abort the packet that is currently being loaded into the FIFO. Note that if the FIFO is double-buffered, FLUSH may have to be set twice to completely clear the FIFO. Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Flushes the latest packet from the endpoint transmit FIFO. The FIFO pointer is reset and the TXRDY bit is cleared. The EPn bit in the USBTXIS register is also set in this situation. Note: This bit should only be set when the TXRDY bit is set. At other times, it may cause data to be corrupted.
2	UNDRN_ERROR1	R/W	0h	Error. Software must clear this bit. Reset type: SYSRSn 0h (R/W) = No error 1h (R/W) = Three attempts have been made to send a packet and no handshake packet has been received. The TXRDY bit is cleared, the EPn bit in the USBTXIS register is set, and the FIFO is completely flushed inthis situation. Note: This bit is valid only when the endpoint is operating in Bulk or Interrupt mode.

Table 38-56. USBTXCSRL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	FIFONE	R/W	0h	FIFO Not Empty Reset type: SYSRSn 0h (R/W) = The FIFO is empty 1h (R/W) = At least one packet is in the transmit FIFO.
0	TXRDY	R/W	0h	Transmit Packet Ready. This bit is cleared automatically when a data packet has been transmitted. The EPn bit in the USBTXIS register is also set at this point. TXRDY is also automatically cleared prior to loading a second packet into a double-buffered FIFO. Reset type: SYSRSn 0h (R/W) = No transmit packet is ready. 1h (R/W) = Software sets this bit after loading a data packet into the TX FIFO.

38.6.2.52 USBTXCSRH1 Register (Offset = 89h) [Reset = 0h]

USBTXCSRH1 is shown in Figure 38-54 and described in Table 38-57.

Return to the Summary Table.

USB Transmit Control and Status Endpoint 1 High

Figure 38-54. USBTXCSRH1 Register

7	6	5	4	3	2	1	0
AUTOSET	ISO	MODE	DMAEN	FDT	DMAMOD	DTWE	DT
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 38-57. USBTXCSRH1 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	AUTOSET	R/W	0h	Auto Set Reset type: SYSRSn 0h (R/W) = The TXRDY bit must be set manually. 1h (R/W) = Enables the TXRDY bit to be automatically set when data of the maximum packet size (value in USBTXMAXP[n]) is loaded into the transmit FIFO. If a packet of less than the maximum packet size is loaded, then the TXRDY bit must be set manually.
6	ISO	R/W	0h	Isochronous Transfers Reset type: SYSRSn 0h (R/W) = Enables the transmit endpoint for bulk or interrupt transfers. 1h (R/W) = Enables the transmit endpoint for isochronous transfers.
5	MODE	R/W	0h	Mode Note: This bit only has an effect when the same endpoint FIFO is used for both transmit and receive transactions. Reset type: SYSRSn 0h (R/W) = Enables the endpoint direction as RX. 1h (R/W) = Enables the endpoint direction as TX.
4	DMAEN	R/W	0h	DMA Request Enable Note: Three TX and three /RX endpoints can be connected to the DMA module. If this bit is set for a particular endpoint, the DMAATX, DMABTX, or DMACTX field in the USB DMA Select (USBDMASEL) register must be programmed correspondingly Reset type: SYSRSn 0h (R/W) = Disables the DMA request for the transmit endpoint. 1h (R/W) = Enables the DMA request for the transmit endpoint.
3	FDT	R/W	0h	Force Data Toggle Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Forces the endpoint DT bit to switch and the data packet to be cleared from the FIFO, regardless of whether an ACK was received. This bit can be used by interrupt transmit endpoints that are used to communicate rate feedback for isochronous endpoints. Note: This bit should only be set when the TXRDY bit is set. At other times, it may cause data to be corrupted.
2	DMAMOD	R/W	0h	DMA Request Mode Reset type: SYSRSn 0h (R/W) = An interrupt is generated after every DMA packet transfer. 1h (R/W) = An interrupt is generated only after the entire DMA transfer is complete. Note: This bit is valid only when the endpoint is operating in Bulk or Interrupt mode.

Table 38-57. USBTXCSRH1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	DTWE	R/W	0h	Data Toggle Write Enable. This bit is automatically cleared once the new value is written. Reset type: SYSRSn 0h (R/W) = The DT bit cannot be written. 1h (R/W) = Enables the current state of the transmit endpoint data to be written (see DT bit).
0	DT	R/W	0h	Data Toggle. When read, this bit indicates the current state of the transmit endpoint data toggle. If DTWE is High, this bit can be written with the required setting of the data toggle. If DTWE is Low, any value written to this bit is ignored. Care should be taken when writing to this bit as it should only be changed to RESET the transmit endpoint. Reset type: SYSRSn

38.6.2.53 USBRXMAXP1 Register (Offset = 8Ah) [Reset = 0h]

USBRXMAXP1 is shown in Figure 38-55 and described in Table 38-58.

Return to the Summary Table.

USB Maximum Receive Data Endpoint 1

Figure 38-55. USBRXMAXP1 Register

15	14	13	12	11	10	9	8
RESERVED					MAXLOAD		
R-0h					R/W-0h		
7	6	5	4	3	2	1	0
MAXLOAD							
R/W-0h							

Table 38-58. USBRXMAXP1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	Reserved
10-0	MAXLOAD	R/W	0h	Maximum Payload specifies the maximum payload in bytes per transaction. Reset type: SYSRSn

38.6.2.54 USBRXCSRL1 Register (Offset = 8Bh) [Reset = 0h]

USBRXCSRL1 is shown in Figure 38-56 and described in Table 38-59.

Return to the Summary Table.

USB Receive Control and Status Endpoint 1 Low

Figure 38-56. USBRXCSRL1 Register

7	6	5	4	3	2	1	0
CLRDT	STALLED	STALLREQPKT	FLUSH	DATAERRNAKTO	OVERERROR1	FULL	RXRDY
W1C-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 38-59. USBRXCSRL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	CLRDT	W1C	0h	Clear Data Toggle. Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Writing a 1 to this bit clears the DT bit in the USBRXCSRH[n] register.
6	STALLED	R/W	0h	Endpoint Stalled. Software must clear this bit. Reset type: SYSRSn 0h (R/W) = No handshake has been received. 1h (R/W) = A STALL handshake has been received. The EPn bit in the USBRXIS register is also set.
5	STALLREQPKT	R/W	0h	Request Packet. This bit is cleared when the RXRDY bit is set. Reset type: SYSRSn 0h (R/W) = No request 1h (R/W) = Requests an IN transaction.
4	FLUSH	R/W	0h	Flush FIFO. If the FIFO is double-buffered, FLUSH may have to be set twice to completely clear the FIFO. Note:This bit should only be set when the RXRDY bit is set. At other times, it may cause data to be corrupted. Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Flushes the next packet to be read from the endpoint receive FIFO. The FIFO pointer is reset and the RXRDY bit is cleared
3	DATAERRNAKTO	R/W	0h	Data Error / NAK Timeout Reset type: SYSRSn 0h (R/W) = Normal operation 1h (R/W) = Isochronous endpoints only: I ndicates that RXRDY is set and the data packet has a CRC or bit-stuff error. This bit is cleared when RXRDY is cleared. Bulk endpoints only: Indicates that the receive endpoint is halted following the receipt of NAK responses for longer than the time set by the NAKLMT field in the USBRXINTERVAL[n] register. Software must clear this bit to allow the endpoint to continue.
2	OVERERROR1	R/W	0h	Error. Software must clear this bit. Reset type: SYSRSn 0h (R/W) = No error 1h (R/W) = Three attempts have been made to receive a packet and no data packet has been received. The Epn bit in the USBRXIS register is set in this situation.
1	FULL	R/W	0h	FIFO Full Reset type: SYSRSn 0h (R/W) = The receive FIFO is not full. 1h (R/W) = No more packets can be loaded into the receive FIFO.

Table 38-59. USBRXCSRL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	RXRDY	R/W	0h	Receive Packet Ready. If the AUTOCLR bit in the USBRXCSRH[n] register is set, then the this bit is automatically cleared when a packet of USBRXMAXP[n] bytes has been unloaded from the receive FIFO. If the AUTOCLR bit is clear, or if packets of less than the maximum packet size are unloaded, then software must clear this bit manually when the packet has been unloaded from the receive FIFO Reset type: SYSRSn 0h (R/W) = No data packet has been received. 1h (R/W) = Indicates that a data packet has been received. The EPn bit in the USBTXIS register is also set in this situation

38.6.2.55 USBRXCSRH1 Register (Offset = 8Bh) [Reset = 0h]

USBRXCSRH1 is shown in Figure 38-57 and described in Table 38-60.

Return to the Summary Table.

USB Receive Control and Status Endpoint 1 High

Figure 38-57. USBRXCSRH1 Register

7	6	5	4	3	2	1	0
AUTOCL	ISOAUTORQ	DMAEN	DISNYETPIDERR	DMAMOD	DTWE	DT	RESERVED
W1C-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 38-60. USBRXCSRH1 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	AUTOCL	W1C	0h	Auto Clear Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Enables the RXRDY bit to be automatically cleared when a packet of USBRXMAXP[n] bytes has been unloaded from the receive FIFO. When packets of less than the maximum packet size are unloaded, RXRDY must be cleared manually. Care must be taken when using DMA to unload the receive FIFO as data is read from the receive FIFO in 4-byte chunks regardless of the value of the MAXLOAD field in the USBRXMAXP[n] register,
6	ISOAUTORQ	R/W	0h	Auto Request Note: This bit is automatically cleared when a short packet is received. Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Enables the REQPKT bit to be automatically set when the RXRDY bit is cleared
5	DMAEN	R/W	0h	DMA Request Enable Note: Three TX and three RX endpoints can be connected to the DMA module. If this bit is set for a particular endpoint, the DMAARX, DMABRX, or DMACRX field in the USB DMA Select (USBDMASEL) register must be programmed correspondingly Reset type: SYSRSn 0h (R/W) = Disables the DMA request for the receive endpoint. 1h (R/W) = Enables the DMA request for the receive endpoint.
4	DISNYETPIDERR	R/W	0h	PID Error. This bit is ignored in bulk or interrupt transactions. Reset type: SYSRSn 0h (R/W) = No error 1h (R/W) = Indicates a PID error in the received packet of an isochronous transaction.
3	DMAMOD	R/W	0h	DMAMOD Note: This bit must not be cleared either before or in the same cycle as the above DMAEN bit is cleared. Reset type: SYSRSn 0h (R/W) = An interrupt is generated after every DMA packet transfer. 1h (R/W) = An interrupt is generated only after the entire DMA transfer is complete.
2	DTWE	R/W	0h	Data Toggle Write Enable. This bit is automatically cleared once the new value is written. Reset type: SYSRSn 0h (R/W) = The DT bit cannot be written. 1h (R/W) = Enables the current state of the receive endpoint data to be written (see DT bit).

Table 38-60. USBRXCSRH1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	DT	R/W	0h	Data Toggle. When read, this bit indicates the current state of the receive data toggle. If DTWE is High, this bit may be written with the required setting of the data toggle. If DTWE is Low, any value written to this bit is ignored. Care should be taken when writing to this bit as it should only be changed to RESET the receive endpoint. Reset type: SYSRSn
0	RESERVED	R/W	0h	Reserved

38.6.2.56 USBRXCOUNT1 Register (Offset = 8Ch) [Reset = 0h]

USBRXCOUNT1 is shown in Figure 38-58 and described in Table 38-61.

Return to the Summary Table.

USB Receive Byte Count Endpoint 1

Figure 38-58. USBRXCOUNT1 Register

15	14	13	12	11	10	9	8
RESERVED			COUNT				
R-0h			R-0h				
7	6	5	4	3	2	1	0
COUNT							
R-0h							

Table 38-61. USBRXCOUNT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-0	COUNT	R	0h	Receive Packet Count indicates the number of bytes in the receive packet. Reset type: SYSRSn

38.6.2.57 USBTXTYPE1 Register (Offset = 8Dh) [Reset = 0h]

USBTXTYPE1 is shown in Figure 38-59 and described in Table 38-62.

Return to the Summary Table.

USB Host Transmit Configure Type Endpoint 1

Figure 38-59. USBTXTYPE1 Register

7	6	5	4	3	2	1	0
SPEED		PROTO		TEP			
R/W-0h		R/W-0h		R/W-0h			

Table 38-62. USBTXTYPE1 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-6	SPEED	R/W	0h	Operating Speed. This bit field specifies the operating speed of the target Device: Reset type: SYSRSn 0h (R/W) = Default. The target is assumed to be using the same connection speed as the USB controller. 1h (R/W) = Reserved 2h (R/W) = Full 3h (R/W) = Low
5-4	PROTO	R/W	0h	Protocol. Software must configure this bit field to select the required protocol for the transmit endpoint: Reset type: SYSRSn 0h (R/W) = Control 1h (R/W) = isochronous 2h (R/W) = Bulk 3h (R/W) = Interrupt
3-0	TEP	R/W	0h	Target Endpoint Number. Software must configure this value to the endpoint number contained in the transmit endpoint descriptor returned to the USB controller during Device enumeration. Reset type: SYSRSn

38.6.2.58 USBTXINTERVAL1 Register (Offset = 8Dh) [Reset = 0h]

USBTXINTERVAL1 is shown in Figure 38-60 and described in Table 38-63.

Return to the Summary Table.

USB Host Transmit Interval Endpoint 1

Figure 38-60. USBTXINTERVAL1 Register

7	6	5	4	3	2	1	0
TXPOLLNAKLMT							
R/W-0h							

Table 38-63. USBTXINTERVAL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-0	TXPOLLNAKLMT	R/W	0h	TX Polling / NAK Limit The polling interval for interrupt/isochronous transfers the NAK limit for bulk transfers. Reset type: SYSRSn

38.6.2.59 USBRXTYPE1 Register (Offset = 8Eh) [Reset = 0h]

USBRXTYPE1 is shown in Figure 38-61 and described in Table 38-64.

Return to the Summary Table.

USB Host Configure Receive Type Endpoint 1

Figure 38-61. USBRXTYPE1 Register

7	6	5	4	3	2	1	0
SPEED		PROTO		TEP			
R/W-0h		R/W-0h		R/W-0h			

Table 38-64. USBRXTYPE1 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-6	SPEED	R/W	0h	Operating Speed. This bit field specifies the operating speed of the target Device: Reset type: SYSRSn 0h (R/W) = Default. The target is assumed to be using the same connection speed as the USB controller. 1h (R/W) = Reserved 2h (R/W) = Full 3h (R/W) = Low
5-4	PROTO	R/W	0h	Protocol. Software must configure this bit field to select the required protocol for the transmit endpoint: Reset type: SYSRSn 0h (R/W) = Control 1h (R/W) = isochronous 2h (R/W) = Bulk 3h (R/W) = Interrupt
3-0	TEP	R/W	0h	Target Endpoint Number. Software must configure this value to the endpoint number contained in the transmit endpoint descriptor returned to the USB controller during Device enumeration. Reset type: SYSRSn

38.6.2.60 USBRXINTERVAL1 Register (Offset = 8Eh) [Reset = 0h]

USBRXINTERVAL1 is shown in Figure 38-62 and described in Table 38-65.

Return to the Summary Table.

USB Host Receive Polling Interval Endpoint 1

Figure 38-62. USBRXINTERVAL1 Register

7	6	5	4	3	2	1	0
RXPOLLNAKLMT							
R/W-0h							

Table 38-65. USBRXINTERVAL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-0	RXPOLLNAKLMT	R/W	0h	RX Polling / NAK Limit The polling interval for interrupt/isochronous transfers the NAK limit for bulk transfers. Reset type: SYSRSn

38.6.2.61 USBTXMAXP2 Register (Offset = 90h) [Reset = 0h]

USBTXMAXP2 is shown in Figure 38-63 and described in Table 38-66.

Return to the Summary Table.

USB Maximum Transmit Data Endpoint 2

Figure 38-63. USBTXMAXP2 Register

15	14	13	12	11	10	9	8
RESERVED					MAXLOAD		
R-0h					R/W-0h		
7	6	5	4	3	2	1	0
MAXLOAD							
R/W-0h							

Table 38-66. USBTXMAXP2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	Reserved
10-0	MAXLOAD	R/W	0h	Maximum Payload specifies the maximum payload in bytes per transaction. Reset type: SYSRSn

38.6.2.62 USBTXCSRL2 Register (Offset = 91h) [Reset = 0h]

USBTXCSRL2 is shown in Figure 38-64 and described in Table 38-67.

Return to the Summary Table.

USB Transmit Control and Status Endpoint 2 Low

Figure 38-64. USBTXCSRL2 Register

7	6	5	4	3	2	1	0
NAKTO	CLRDT	STALLED	STALL_SETUP	FLUSH	UNDRNERROR2	FIFONE	TXRDY
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 38-67. USBTXCSRL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	NAKTO	R/W	0h	NAK Timeout. Software must clear this bit to allow the endpoint to continue. Reset type: SYSRSn 0h (R/W) = No timeout 1h (R/W) = Bulk endpoints only: Indicates that the transmit endpoint is halted following the receipt of NAK responses for longer than the time set by the NAKLMT field in the USBTXINTERVAL[n] register.
6	CLRDT	R/W	0h	Clear DataToggle Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Writing a 1 to this bit clears the DT bit in the USBTXCSRH[n] register.
5	STALLED	R/W	0h	Endpoint Stalled. Software must clear this bit. Reset type: SYSRSn 0h (R/W) = A STALL handshake has not been received 1h (R/W) = Indicates that a STALL handshake has been received. When this bit is set, any DMA request that is in progress is stopped, the FIFO is completely flushed, and the TXRDY bit is cleared.
4	STALL_SETUP	R/W	0h	Setup Packet. Reset type: SYSRSn 0h (R/W) = No SETUP token is sent. 1h (R/W) = Sends a SETUP token instead of an OUT token for the transaction. This bit should be set at the same time as the TXRDY bit is set. Note: Setting this bit also clears the DT bit in the USBTXCSRH[n] register.
3	FLUSH	R/W	0h	Flush FIFO. This bit can be set simultaneously with the TXRDY bit to abort the packet that is currently being loaded into the FIFO. Note that if the FIFO is double-buffered, FLUSH may have to be set twice to completely clear the FIFO. Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Flushes the latest packet from the endpoint transmit FIFO. The FIFO pointer is reset and the TXRDY bit is cleared. The EPn bit in the USBTXIS register is also set in this situation. Note: This bit should only be set when the TXRDY bit is set. At other times, it may cause data to be corrupted.
2	UNDRNERROR2	R/W	0h	Error. Software must clear this bit. Reset type: SYSRSn 0h (R/W) = No error 1h (R/W) = Three attempts have been made to send a packet and no handshake packet has been received. The TXRDY bit is cleared, the EPn bit in the USBTXIS register is set, and the FIFO is completely flushed inthis situation. Note: This bit is valid only when the endpoint is operating in Bulk or Interrupt mode.

Table 38-67. USBTXCSRL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	FIFONE	R/W	0h	FIFO Not Empty Reset type: SYSRSn 0h (R/W) = The FIFO is empty 1h (R/W) = At least one packet is in the transmit FIFO.
0	TXRDY	R/W	0h	Transmit Packet Ready. This bit is cleared automatically when a data packet has been transmitted. The EPn bit in the USBTXIS register is also set at this point. TXRDY is also automatically cleared prior to loading a second packet into a double-buffered FIFO. Reset type: SYSRSn 0h (R/W) = No transmit packet is ready. 1h (R/W) = Software sets this bit after loading a data packet into the TX FIFO.

38.6.2.63 USBTXCSRH2 Register (Offset = 91h) [Reset = 0h]

USBTXCSRH2 is shown in Figure 38-65 and described in Table 38-68.

Return to the Summary Table.

USB Transmit Control and Status Endpoint 2 High

Figure 38-65. USBTXCSRH2 Register

7	6	5	4	3	2	1	0
AUTOSET	ISO	MODE	DMAEN	FDT	DMAMOD	DTWE	DT
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 38-68. USBTXCSRH2 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	AUTOSET	R/W	0h	Auto Set Reset type: SYSRSn 0h (R/W) = The TXRDY bit must be set manually. 1h (R/W) = Enables the TXRDY bit to be automatically set when data of the maximum packet size (value in USBTXMAXP[n]) is loaded into the transmit FIFO. If a packet of less than the maximum packet size is loaded, then the TXRDY bit must be set manually.
6	ISO	R/W	0h	Isochronous Transfers Reset type: SYSRSn 0h (R/W) = Enables the transmit endpoint for bulk or interrupt transfers. 1h (R/W) = Enables the transmit endpoint for isochronous transfers.
5	MODE	R/W	0h	Mode Note: This bit only has an effect when the same endpoint FIFO is used for both transmit and receive transactions. Reset type: SYSRSn 0h (R/W) = Enables the endpoint direction as RX. 1h (R/W) = Enables the endpoint direction as TX.
4	DMAEN	R/W	0h	DMA Request Enable Note: Three TX and three /RX endpoints can be connected to the DMA module. If this bit is set for a particular endpoint, the DMAATX, DMABTX, or DMACTX field in the USB DMA Select (USBDMASEL) register must be programmed correspondingly Reset type: SYSRSn 0h (R/W) = Disables the DMA request for the transmit endpoint. 1h (R/W) = Enables the DMA request for the transmit endpoint.
3	FDT	R/W	0h	Force Data Toggle Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Forces the endpoint DT bit to switch and the data packet to be cleared from the FIFO, regardless of whether an ACK was received. This bit can be used by interrupt transmit endpoints that are used to communicate rate feedback for isochronous endpoints. Note: This bit should only be set when the TXRDY bit is set. At other times, it may cause data to be corrupted.
2	DMAMOD	R/W	0h	DMA Request Mode Reset type: SYSRSn 0h (R/W) = An interrupt is generated after every DMA packet transfer. 1h (R/W) = An interrupt is generated only after the entire DMA transfer is complete. Note: This bit is valid only when the endpoint is operating in Bulk or Interrupt mode.

Table 38-68. USBTXCSRH2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	DTWE	R/W	0h	Data Toggle Write Enable. This bit is automatically cleared once the new value is written. Reset type: SYSRSn 0h (R/W) = The DT bit cannot be written. 1h (R/W) = Enables the current state of the transmit endpoint data to be written (see DT bit).
0	DT	R/W	0h	Data Toggle. When read, this bit indicates the current state of the transmit endpoint data toggle. If DTWE is High, this bit can be written with the required setting of the data toggle. If DTWE is Low, any value written to this bit is ignored. Care should be taken when writing to this bit as it should only be changed to RESET the transmit endpoint. Reset type: SYSRSn

38.6.2.64 USBRXMAXP2 Register (Offset = 92h) [Reset = 0h]

USBRXMAXP2 is shown in Figure 38-66 and described in Table 38-69.

Return to the Summary Table.

USB Maximum Receive Data Endpoint 2

Figure 38-66. USBRXMAXP2 Register

15	14	13	12	11	10	9	8
RESERVED					MAXLOAD		
R-0h					R/W-0h		
7	6	5	4	3	2	1	0
MAXLOAD							
R/W-0h							

Table 38-69. USBRXMAXP2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	Reserved
10-0	MAXLOAD	R/W	0h	Maximum Payload specifies the maximum payload in bytes per transaction. Reset type: SYSRSn

38.6.2.65 USBRXCSRL2 Register (Offset = 93h) [Reset = 0h]

USBRXCSRL2 is shown in Figure 38-67 and described in Table 38-70.

Return to the Summary Table.

USB Receive Control and Status Endpoint 2 Low

Figure 38-67. USBRXCSRL2 Register

7	6	5	4	3	2	1	0
CLRDT	STALLED	STALLREQPKT	FLUSH	DATAERRNAKTO	OVERERROR2	FULL	RXRDY
W1C-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 38-70. USBRXCSRL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	CLRDT	W1C	0h	Clear Data Toggle. Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Writing a 1 to this bit clears the DT bit in the USBRXCSRH[n] register.
6	STALLED	R/W	0h	Endpoint Stalled. Software must clear this bit. Reset type: SYSRSn 0h (R/W) = No handshake has been received. 1h (R/W) = A STALL handshake has been received. The EPn bit in the USBRXIS register is also set.
5	STALLREQPKT	R/W	0h	Request Packet. This bit is cleared when the RXRDY bit is set. Reset type: SYSRSn 0h (R/W) = No request 1h (R/W) = Requests an IN transaction.
4	FLUSH	R/W	0h	Flush FIFO. If the FIFO is double-buffered, FLUSH may have to be set twice to completely clear the FIFO. Note:This bit should only be set when the RXRDY bit is set. At other times, it may cause data to be corrupted. Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Flushes the next packet to be read from the endpoint receive FIFO. The FIFO pointer is reset and the RXRDY bit is cleared
3	DATAERRNAKTO	R/W	0h	Data Error / NAK Timeout Reset type: SYSRSn 0h (R/W) = Normal operation 1h (R/W) = Isochronous endpoints only: I ndicates that RXRDY is set and the data packet has a CRC or bit-stuff error. This bit is cleared when RXRDY is cleared. Bulk endpoints only: Indicates that the receive endpoint is halted following the receipt of NAK responses for longer than the time set by the NAKLMT field in the USBRXINTERVAL[n] register. Software must clear this bit to allow the endpoint to continue.
2	OVERERROR2	R/W	0h	Error. Software must clear this bit. Reset type: SYSRSn 0h (R/W) = No error 1h (R/W) = Three attempts have been made to receive a packet and no data packet has been received. The Epn bit in the USBRXIS register is set in this situation.
1	FULL	R/W	0h	FIFO Full Reset type: SYSRSn 0h (R/W) = The receive FIFO is not full. 1h (R/W) = No more packets can be loaded into the receive FIFO.

Table 38-70. USBRXCSRL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	RXRDY	R/W	0h	Receive Packet Ready. If the AUTOCLR bit in the USBRXCSRH[n] register is set, then the this bit is automatically cleared when a packet of USBRXMAXP[n] bytes has been unloaded from the receive FIFO. If the AUTOCLR bit is clear, or if packets of less than the maximum packet size are unloaded, then software must clear this bit manually when the packet has been unloaded from the receive FIFO Reset type: SYSRSn 0h (R/W) = No data packet has been received. 1h (R/W) = Indicates that a data packet has been received. The EPn bit in the USBTXIS register is also set in this situation

38.6.2.66 USBRXCSRH2 Register (Offset = 93h) [Reset = 0h]

USBRXCSRH2 is shown in Figure 38-68 and described in Table 38-71.

Return to the Summary Table.

USB Receive Control and Status Endpoint 2 High

Figure 38-68. USBRXCSRH2 Register

7	6	5	4	3	2	1	0
AUTOCL	ISOAUTORQ	DMAEN	DISNYETPIDERR	DMAMOD	DTWE	DT	RESERVED
W1C-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 38-71. USBRXCSRH2 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	AUTOCL	W1C	0h	Auto Clear Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Enables the RXRDY bit to be automatically cleared when a packet of USBRXMAXP[n] bytes has been unloaded from the receive FIFO. When packets of less than the maximum packet size are unloaded, RXRDY must be cleared manually. Care must be taken when using DMA to unload the receive FIFO as data is read from the receive FIFO in 4-byte chunks regardless of the value of the MAXLOAD field in the USBRXMAXP[n] register,
6	ISOAUTORQ	R/W	0h	Auto Request Note: This bit is automatically cleared when a short packet is received. Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Enables the REQPKT bit to be automatically set when the RXRDY bit is cleared
5	DMAEN	R/W	0h	DMA Request Enable Note: Three TX and three RX endpoints can be connected to the DMA module. If this bit is set for a particular endpoint, the DMAARX, DMABRX, or DMACRX field in the USB DMA Select (USBDMASEL) register must be programmed correspondingly Reset type: SYSRSn 0h (R/W) = Disables the DMA request for the receive endpoint. 1h (R/W) = Enables the DMA request for the receive endpoint.
4	DISNYETPIDERR	R/W	0h	PID Error. This bit is ignored in bulk or interrupt transactions. Reset type: SYSRSn 0h (R/W) = No error 1h (R/W) = Indicates a PID error in the received packet of an isochronous transaction.
3	DMAMOD	R/W	0h	DMAMOD Note: This bit must not be cleared either before or in the same cycle as the above DMAEN bit is cleared. Reset type: SYSRSn 0h (R/W) = An interrupt is generated after every DMA packet transfer. 1h (R/W) = An interrupt is generated only after the entire DMA transfer is complete.
2	DTWE	R/W	0h	Data Toggle Write Enable. This bit is automatically cleared once the new value is written. Reset type: SYSRSn 0h (R/W) = The DT bit cannot be written. 1h (R/W) = Enables the current state of the receive endpoint data to be written (see DT bit).

Table 38-71. USBRXCSRH2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	DT	R/W	0h	Data Toggle. When read, this bit indicates the current state of the receive data toggle. If DTWE is High, this bit may be written with the required setting of the data toggle. If DTWE is Low, any value written to this bit is ignored. Care should be taken when writing to this bit as it should only be changed to RESET the receive endpoint. Reset type: SYSRSn
0	RESERVED	R/W	0h	Reserved

38.6.2.67 USBRXCOUNT2 Register (Offset = 94h) [Reset = 0h]

USBRXCOUNT2 is shown in Figure 38-69 and described in Table 38-72.

Return to the Summary Table.

USB Receive Byte Count Endpoint 2

Figure 38-69. USBRXCOUNT2 Register

15	14	13	12	11	10	9	8
RESERVED			COUNT				
R-0h			R-0h				
7	6	5	4	3	2	1	0
COUNT							
R-0h							

Table 38-72. USBRXCOUNT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-0	COUNT	R	0h	Receive Packet Count indicates the number of bytes in the receive packet. Reset type: SYSRSn

38.6.2.68 USBTXTYPE2 Register (Offset = 95h) [Reset = 0h]

USBTXTYPE2 is shown in Figure 38-70 and described in Table 38-73.

Return to the Summary Table.

USB Host Transmit Configure Type Endpoint 2

Figure 38-70. USBTXTYPE2 Register

7	6	5	4	3	2	1	0
SPEED		PROTO		TEP			
R/W-0h		R/W-0h		R/W-0h			

Table 38-73. USBTXTYPE2 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-6	SPEED	R/W	0h	Operating Speed. This bit field specifies the operating speed of the target Device: Reset type: SYSRSn 0h (R/W) = Default. The target is assumed to be using the same connection speed as the USB controller. 1h (R/W) = Reserved 2h (R/W) = Full 3h (R/W) = Low
5-4	PROTO	R/W	0h	Protocol. Software must configure this bit field to select the required protocol for the transmit endpoint: Reset type: SYSRSn 0h (R/W) = Control 1h (R/W) = isochronous 2h (R/W) = Bulk 3h (R/W) = Interrupt
3-0	TEP	R/W	0h	Target Endpoint Number. Software must configure this value to the endpoint number contained in the transmit endpoint descriptor returned to the USB controller during Device enumeration. Reset type: SYSRSn

38.6.2.69 USBTXINTERVAL2 Register (Offset = 95h) [Reset = 0h]

USBTXINTERVAL2 is shown in Figure 38-71 and described in Table 38-74.

Return to the Summary Table.

USB Host Transmit Interval Endpoint 2

Figure 38-71. USBTXINTERVAL2 Register

7	6	5	4	3	2	1	0
TXPOLLNAKLMT							
R/W-0h							

Table 38-74. USBTXINTERVAL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-0	TXPOLLNAKLMT	R/W	0h	TX Polling / NAK Limit The polling interval for interrupt/isochronous transfers the NAK limit for bulk transfers. Reset type: SYSRSn

38.6.2.70 USBRXTYPE2 Register (Offset = 96h) [Reset = 0h]

USBRXTYPE2 is shown in Figure 38-72 and described in Table 38-75.

Return to the Summary Table.

USB Host Configure Receive Type Endpoint 2

Figure 38-72. USBRXTYPE2 Register

7	6	5	4	3	2	1	0
SPEED		PROTO		TEP			
R/W-0h		R/W-0h		R/W-0h			

Table 38-75. USBRXTYPE2 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-6	SPEED	R/W	0h	Operating Speed. This bit field specifies the operating speed of the target Device: Reset type: SYSRSn 0h (R/W) = Default. The target is assumed to be using the same connection speed as the USB controller. 1h (R/W) = Reserved 2h (R/W) = Full 3h (R/W) = Low
5-4	PROTO	R/W	0h	Protocol. Software must configure this bit field to select the required protocol for the transmit endpoint: Reset type: SYSRSn 0h (R/W) = Control 1h (R/W) = isochronous 2h (R/W) = Bulk 3h (R/W) = Interrupt
3-0	TEP	R/W	0h	Target Endpoint Number. Software must configure this value to the endpoint number contained in the transmit endpoint descriptor returned to the USB controller during Device enumeration. Reset type: SYSRSn

38.6.2.71 USBRXINTERVAL2 Register (Offset = 96h) [Reset = 0h]

USBRXINTERVAL2 is shown in Figure 38-73 and described in Table 38-76.

Return to the Summary Table.

USB Host Receive Polling Interval Endpoint 2

Figure 38-73. USBRXINTERVAL2 Register

7	6	5	4	3	2	1	0
RXPOLLNAKLMT							
R/W-0h							

Table 38-76. USBRXINTERVAL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-0	RXPOLLNAKLMT	R/W	0h	RX Polling / NAK Limit The polling interval for interrupt/isochronous transfers the NAK limit for bulk transfers. Reset type: SYSRSn

38.6.2.72 USBTXMAXP3 Register (Offset = 98h) [Reset = 0h]

USBTXMAXP3 is shown in Figure 38-74 and described in Table 38-77.

Return to the Summary Table.

USB Maximum Transmit Data Endpoint 3

Figure 38-74. USBTXMAXP3 Register

15	14	13	12	11	10	9	8
RESERVED					MAXLOAD		
R-0h					R/W-0h		
7	6	5	4	3	2	1	0
MAXLOAD							
R/W-0h							

Table 38-77. USBTXMAXP3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	Reserved
10-0	MAXLOAD	R/W	0h	Maximum Payload specifies the maximum payload in bytes per transaction. Reset type: SYSRSn

38.6.2.73 USBTXCSRL3 Register (Offset = 99h) [Reset = 0h]

USBTXCSRL3 is shown in Figure 38-75 and described in Table 38-78.

Return to the Summary Table.

USB Transmit Control and Status Endpoint 3 Low

Figure 38-75. USBTXCSRL3 Register

7	6	5	4	3	2	1	0
NAKTO	CLRDT	STALLED	STALL_SETUP	FLUSH	UNDRNERROR3	FIFONE	TXRDY
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 38-78. USBTXCSRL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	NAKTO	R/W	0h	NAK Timeout. Software must clear this bit to allow the endpoint to continue. Reset type: SYSRSn 0h (R/W) = No timeout 1h (R/W) = Bulk endpoints only: Indicates that the transmit endpoint is halted following the receipt of NAK responses for longer than the time set by the NAKLMT field in the USBTXINTERVAL[n] register.
6	CLRDT	R/W	0h	Clear DataToggle Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Writing a 1 to this bit clears the DT bit in the USBTXCSRH[n] register.
5	STALLED	R/W	0h	Endpoint Stalled. Software must clear this bit. Reset type: SYSRSn 0h (R/W) = A STALL handshake has not been received 1h (R/W) = Indicates that a STALL handshake has been received. When this bit is set, any DMA request that is in progress is stopped, the FIFO is completely flushed, and the TXRDY bit is cleared.
4	STALL_SETUP	R/W	0h	Setup Packet. Reset type: SYSRSn 0h (R/W) = No SETUP token is sent. 1h (R/W) = Sends a SETUP token instead of an OUT token for the transaction. This bit should be set at the same time as the TXRDY bit is set. Note: Setting this bit also clears the DT bit in the USBTXCSRH[n] register.
3	FLUSH	R/W	0h	Flush FIFO. This bit can be set simultaneously with the TXRDY bit to abort the packet that is currently being loaded into the FIFO. Note that if the FIFO is double-buffered, FLUSH may have to be set twice to completely clear the FIFO. Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Flushes the latest packet from the endpoint transmit FIFO. The FIFO pointer is reset and the TXRDY bit is cleared. The EPn bit in the USBTXIS register is also set in this situation. Note: This bit should only be set when the TXRDY bit is set. At other times, it may cause data to be corrupted.
2	UNDRNERROR3	R/W	0h	Error. Software must clear this bit. Reset type: SYSRSn 0h (R/W) = No error 1h (R/W) = Three attempts have been made to send a packet and no handshake packet has been received. The TXRDY bit is cleared, the EPn bit in the USBTXIS register is set, and the FIFO is completely flushed inthis situation. Note: This bit is valid only when the endpoint is operating in Bulk or Interrupt mode.

Table 38-78. USBTXCSRL3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	FIFONE	R/W	0h	FIFO Not Empty Reset type: SYSRSn 0h (R/W) = The FIFO is empty 1h (R/W) = At least one packet is in the transmit FIFO.
0	TXRDY	R/W	0h	Transmit Packet Ready. This bit is cleared automatically when a data packet has been transmitted. The EPn bit in the USBTXIS register is also set at this point. TXRDY is also automatically cleared prior to loading a second packet into a double-buffered FIFO. Reset type: SYSRSn 0h (R/W) = No transmit packet is ready. 1h (R/W) = Software sets this bit after loading a data packet into the TX FIFO.

38.6.2.74 USBTXCSRH3 Register (Offset = 99h) [Reset = 0h]

USBTXCSRH3 is shown in Figure 38-76 and described in Table 38-79.

Return to the Summary Table.

USB Transmit Control and Status Endpoint 3 High

Figure 38-76. USBTXCSRH3 Register

7	6	5	4	3	2	1	0
AUTOSET	ISO	MODE	DMAEN	FDT	DMAMOD	DTWE	DT
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 38-79. USBTXCSRH3 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	AUTOSET	R/W	0h	Auto Set Reset type: SYSRSn 0h (R/W) = The TXRDY bit must be set manually. 1h (R/W) = Enables the TXRDY bit to be automatically set when data of the maximum packet size (value in USBTXMAXP[n]) is loaded into the transmit FIFO. If a packet of less than the maximum packet size is loaded, then the TXRDY bit must be set manually.
6	ISO	R/W	0h	Isochronous Transfers Reset type: SYSRSn 0h (R/W) = Enables the transmit endpoint for bulk or interrupt transfers. 1h (R/W) = Enables the transmit endpoint for isochronous transfers.
5	MODE	R/W	0h	Mode Note: This bit only has an effect when the same endpoint FIFO is used for both transmit and receive transactions. Reset type: SYSRSn 0h (R/W) = Enables the endpoint direction as RX. 1h (R/W) = Enables the endpoint direction as TX.
4	DMAEN	R/W	0h	DMA Request Enable Note: Three TX and three /RX endpoints can be connected to the DMA module. If this bit is set for a particular endpoint, the DMAATX, DMABTX, or DMACTX field in the USB DMA Select (USBDMASEL) register must be programmed correspondingly Reset type: SYSRSn 0h (R/W) = Disables the DMA request for the transmit endpoint. 1h (R/W) = Enables the DMA request for the transmit endpoint.
3	FDT	R/W	0h	Force Data Toggle Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Forces the endpoint DT bit to switch and the data packet to be cleared from the FIFO, regardless of whether an ACK was received. This bit can be used by interrupt transmit endpoints that are used to communicate rate feedback for isochronous endpoints. Note: This bit should only be set when the TXRDY bit is set. At other times, it may cause data to be corrupted.
2	DMAMOD	R/W	0h	DMA Request Mode Reset type: SYSRSn 0h (R/W) = An interrupt is generated after every DMA packet transfer. 1h (R/W) = An interrupt is generated only after the entire DMA transfer is complete. Note: This bit is valid only when the endpoint is operating in Bulk or Interrupt mode.

Table 38-79. USBTXCSRH3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	DTWE	R/W	0h	Data Toggle Write Enable. This bit is automatically cleared once the new value is written. Reset type: SYSRSn 0h (R/W) = The DT bit cannot be written. 1h (R/W) = Enables the current state of the transmit endpoint data to be written (see DT bit).
0	DT	R/W	0h	Data Toggle. When read, this bit indicates the current state of the transmit endpoint data toggle. If DTWE is High, this bit can be written with the required setting of the data toggle. If DTWE is Low, any value written to this bit is ignored. Care should be taken when writing to this bit as it should only be changed to RESET the transmit endpoint. Reset type: SYSRSn

38.6.2.75 USBRXMAXP3 Register (Offset = 9Ah) [Reset = 0h]

USBRXMAXP3 is shown in Figure 38-77 and described in Table 38-80.

Return to the Summary Table.

USB Maximum Receive Data Endpoint 3

Figure 38-77. USBRXMAXP3 Register

15	14	13	12	11	10	9	8
RESERVED					MAXLOAD		
R-0h					R/W-0h		
7	6	5	4	3	2	1	0
MAXLOAD							
R/W-0h							

Table 38-80. USBRXMAXP3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	Reserved
10-0	MAXLOAD	R/W	0h	Maximum Payload specifies the maximum payload in bytes per transaction. Reset type: SYSRSn

38.6.2.76 USBRXCSRL3 Register (Offset = 9Bh) [Reset = 0h]

USBRXCSRL3 is shown in Figure 38-78 and described in Table 38-81.

Return to the Summary Table.

USB Receive Control and Status Endpoint 3 Low

Figure 38-78. USBRXCSRL3 Register

7	6	5	4	3	2	1	0
CLRDT	STALLED	STALLREQPKT	FLUSH	DATAERRNAKTO	OVERERROR3	FULL	RXRDY
W1C-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 38-81. USBRXCSRL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	CLRDT	W1C	0h	Clear Data Toggle. Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Writing a 1 to this bit clears the DT bit in the USBRXCSRH[n] register.
6	STALLED	R/W	0h	Endpoint Stalled. Software must clear this bit. Reset type: SYSRSn 0h (R/W) = No handshake has been received. 1h (R/W) = A STALL handshake has been received. The EPn bit in the USBRXIS register is also set.
5	STALLREQPKT	R/W	0h	Request Packet. This bit is cleared when the RXRDY bit is set. Reset type: SYSRSn 0h (R/W) = No request 1h (R/W) = Requests an IN transaction.
4	FLUSH	R/W	0h	Flush FIFO. If the FIFO is double-buffered, FLUSH may have to be set twice to completely clear the FIFO. Note:This bit should only be set when the RXRDY bit is set. At other times, it may cause data to be corrupted. Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Flushes the next packet to be read from the endpoint receive FIFO. The FIFO pointer is reset and the RXRDY bit is cleared
3	DATAERRNAKTO	R/W	0h	Data Error / NAK Timeout Reset type: SYSRSn 0h (R/W) = Normal operation 1h (R/W) = Isochronous endpoints only: I ndicates that RXRDY is set and the data packet has a CRC or bit-stuff error. This bit is cleared when RXRDY is cleared. Bulk endpoints only: Indicates that the receive endpoint is halted following the receipt of NAK responses for longer than the time set by the NAKLMT field in the USBRXINTERVAL[n] register. Software must clear this bit to allow the endpoint to continue.
2	OVERERROR3	R/W	0h	Error. Software must clear this bit. Reset type: SYSRSn 0h (R/W) = No error 1h (R/W) = Three attempts have been made to receive a packet and no data packet has been received. The Epn bit in the USBRXIS register is set in this situation.
1	FULL	R/W	0h	FIFO Full Reset type: SYSRSn 0h (R/W) = The receive FIFO is not full. 1h (R/W) = No more packets can be loaded into the receive FIFO.

Table 38-81. USBRXCSRL3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	RXRDY	R/W	0h	Receive Packet Ready. If the AUTOCLR bit in the USBRXCSRH[n] register is set, then the this bit is automatically cleared when a packet of USBRXMAXP[n] bytes has been unloaded from the receive FIFO. If the AUTOCLR bit is clear, or if packets of less than the maximum packet size are unloaded, then software must clear this bit manually when the packet has been unloaded from the receive FIFO Reset type: SYSRSn 0h (R/W) = No data packet has been received. 1h (R/W) = Indicates that a data packet has been received. The EPn bit in the USBTXIS register is also set in this situation

38.6.2.77 USBRXCSRH3 Register (Offset = 9Bh) [Reset = 0h]

USBRXCSRH3 is shown in Figure 38-79 and described in Table 38-82.

Return to the Summary Table.

USB Receive Control and Status Endpoint 3 High

Figure 38-79. USBRXCSRH3 Register

7	6	5	4	3	2	1	0
AUTOCL	ISOAUTORQ	DMAEN	DISNYETPIDERR	DMAMOD	DTWE	DT	RESERVED
W1C-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 38-82. USBRXCSRH3 Register Field Descriptions

Bit	Field	Type	Reset	Description
7	AUTOCL	W1C	0h	Auto Clear Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Enables the RXRDY bit to be automatically cleared when a packet of USBRXMAXP[n] bytes has been unloaded from the receive FIFO. When packets of less than the maximum packet size are unloaded, RXRDY must be cleared manually. Care must be taken when using DMA to unload the receive FIFO as data is read from the receive FIFO in 4-byte chunks regardless of the value of the MAXLOAD field in the USBRXMAXP[n] register,
6	ISOAUTORQ	R/W	0h	Auto Request Note: This bit is automatically cleared when a short packet is received. Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Enables the REQPKT bit to be automatically set when the RXRDY bit is cleared
5	DMAEN	R/W	0h	DMA Request Enable Note: Three TX and three RX endpoints can be connected to the DMA module. If this bit is set for a particular endpoint, the DMAARX, DMABRX, or DMACRX field in the USB DMA Select (USBDMASEL) register must be programmed correspondingly Reset type: SYSRSn 0h (R/W) = Disables the DMA request for the receive endpoint. 1h (R/W) = Enables the DMA request for the receive endpoint.
4	DISNYETPIDERR	R/W	0h	PID Error. This bit is ignored in bulk or interrupt transactions. Reset type: SYSRSn 0h (R/W) = No error 1h (R/W) = Indicates a PID error in the received packet of an isochronous transaction.
3	DMAMOD	R/W	0h	DMAMOD Note: This bit must not be cleared either before or in the same cycle as the above DMAEN bit is cleared. Reset type: SYSRSn 0h (R/W) = An interrupt is generated after every DMA packet transfer. 1h (R/W) = An interrupt is generated only after the entire DMA transfer is complete.
2	DTWE	R/W	0h	Data Toggle Write Enable. This bit is automatically cleared once the new value is written. Reset type: SYSRSn 0h (R/W) = The DT bit cannot be written. 1h (R/W) = Enables the current state of the receive endpoint data to be written (see DT bit).

Table 38-82. USBRXCSRH3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	DT	R/W	0h	Data Toggle. When read, this bit indicates the current state of the receive data toggle. If DTWE is High, this bit may be written with the required setting of the data toggle. If DTWE is Low, any value written to this bit is ignored. Care should be taken when writing to this bit as it should only be changed to RESET the receive endpoint. Reset type: SYSRSn
0	RESERVED	R/W	0h	Reserved

38.6.2.78 USBRXCOUNT3 Register (Offset = 9Ch) [Reset = 0h]

USBRXCOUNT3 is shown in Figure 38-80 and described in Table 38-83.

Return to the Summary Table.

USB Receive Byte Count Endpoint 3

Figure 38-80. USBRXCOUNT3 Register

15	14	13	12	11	10	9	8
RESERVED			COUNT				
R-0h			R-0h				
7	6	5	4	3	2	1	0
COUNT							
R-0h							

Table 38-83. USBRXCOUNT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-0	COUNT	R	0h	Receive Packet Count indicates the number of bytes in the receive packet. Reset type: SYSRSn

38.6.2.79 USBTXTYPE3 Register (Offset = 9Dh) [Reset = 0h]

USBTXTYPE3 is shown in Figure 38-81 and described in Table 38-84.

Return to the Summary Table.

USB Host Transmit Configure Type Endpoint 3

Figure 38-81. USBTXTYPE3 Register

7	6	5	4	3	2	1	0
SPEED		PROTO		TEP			
R/W-0h		R/W-0h		R/W-0h			

Table 38-84. USBTXTYPE3 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-6	SPEED	R/W	0h	Operating Speed. This bit field specifies the operating speed of the target Device: Reset type: SYSRSn 0h (R/W) = Default. The target is assumed to be using the same connection speed as the USB controller. 1h (R/W) = Reserved 2h (R/W) = Full 3h (R/W) = Low
5-4	PROTO	R/W	0h	Protocol. Software must configure this bit field to select the required protocol for the transmit endpoint: Reset type: SYSRSn 0h (R/W) = Control 1h (R/W) = isochronous 2h (R/W) = Bulk 3h (R/W) = Interrupt
3-0	TEP	R/W	0h	Target Endpoint Number. Software must configure this value to the endpoint number contained in the transmit endpoint descriptor returned to the USB controller during Device enumeration. Reset type: SYSRSn

38.6.2.80 USBTXINTERVAL3 Register (Offset = 9Dh) [Reset = 0h]

USBTXINTERVAL3 is shown in Figure 38-82 and described in Table 38-85.

Return to the Summary Table.

USB Host Transmit Interval Endpoint 3

Figure 38-82. USBTXINTERVAL3 Register

7	6	5	4	3	2	1	0
TXPOLLNAKLMT							
R/W-0h							

Table 38-85. USBTXINTERVAL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-0	TXPOLLNAKLMT	R/W	0h	TX Polling / NAK Limit The polling interval for interrupt/isochronous transfers the NAK limit for bulk transfers. Reset type: SYSRSn

38.6.2.81 USBRXTYPE3 Register (Offset = 9Eh) [Reset = 0h]

USBRXTYPE3 is shown in Figure 38-83 and described in Table 38-86.

Return to the Summary Table.

USB Host Configure Receive Type Endpoint 3

Figure 38-83. USBRXTYPE3 Register

7	6	5	4	3	2	1	0
SPEED		PROTO		TEP			
R/W-0h		R/W-0h		R/W-0h			

Table 38-86. USBRXTYPE3 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-6	SPEED	R/W	0h	Operating Speed. This bit field specifies the operating speed of the target Device: Reset type: SYSRSn 0h (R/W) = Default. The target is assumed to be using the same connection speed as the USB controller. 1h (R/W) = Reserved 2h (R/W) = Full 3h (R/W) = Low
5-4	PROTO	R/W	0h	Protocol. Software must configure this bit field to select the required protocol for the transmit endpoint: Reset type: SYSRSn 0h (R/W) = Control 1h (R/W) = isochronous 2h (R/W) = Bulk 3h (R/W) = Interrupt
3-0	TEP	R/W	0h	Target Endpoint Number. Software must configure this value to the endpoint number contained in the transmit endpoint descriptor returned to the USB controller during Device enumeration. Reset type: SYSRSn

38.6.2.82 USBRXINTERVAL3 Register (Offset = 9Eh) [Reset = 0h]

USBRXINTERVAL3 is shown in Figure 38-84 and described in Table 38-87.

Return to the Summary Table.

USB Host Receive Polling Interval Endpoint 3

Figure 38-84. USBRXINTERVAL3 Register

7	6	5	4	3	2	1	0
RXPOLLNAKLMT							
R/W-0h							

Table 38-87. USBRXINTERVAL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
7-0	RXPOLLNAKLMT	R/W	0h	RX Polling / NAK Limit The polling interval for interrupt/isochronous transfers the NAK limit for bulk transfers. Reset type: SYSRSn

38.6.2.83 USBRQPKTCOUNT1 Register (Offset = 182h) [Reset = 0h]

USBRQPKTCOUNT1 is shown in Figure 38-85 and described in Table 38-88.

Return to the Summary Table.

USB Request Packet Count in Block Transfer Endpoint 1

Figure 38-85. USBRQPKTCOUNT1 Register

15	14	13	12	11	10	9	8
RESERVED			COUNT				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
COUNT							
R/W-0h							

Table 38-88. USBRQPKTCOUNT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-0	COUNT	R/W	0h	Block Transfer Packet Count sets the number of packets of the size defined by the MAXLOAD bit field that are to be transferred in a block transfer. Note: This is only used in Host mode when AUTORQ is set. The bit has no effect in Device mode or when AUTORQ is not set. Reset type: SYSRSn

38.6.2.84 USBRQPKTCOUNT2 Register (Offset = 184h) [Reset = 0h]

USBRQPKTCOUNT2 is shown in Figure 38-86 and described in Table 38-89.

Return to the Summary Table.

USB Request Packet Count in Block Transfer Endpoint 2

Figure 38-86. USBRQPKTCOUNT2 Register

15	14	13	12	11	10	9	8
RESERVED			COUNT				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
COUNT							
R/W-0h							

Table 38-89. USBRQPKTCOUNT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-0	COUNT	R/W	0h	Block Transfer Packet Count sets the number of packets of the size defined by the MAXLOAD bit field that are to be transferred in a block transfer. Note: This is only used in Host mode when AUTORQ is set. The bit has no effect in Device mode or when AUTORQ is not set. Reset type: SYSRSn

38.6.2.85 USBRQPKTCOUNT3 Register (Offset = 186h) [Reset = 0h]

USBRQPKTCOUNT3 is shown in Figure 38-87 and described in Table 38-90.

Return to the Summary Table.

USB Request Packet Count in Block Transfer Endpoint 3

Figure 38-87. USBRQPKTCOUNT3 Register

15	14	13	12	11	10	9	8
RESERVED			COUNT				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
COUNT							
R/W-0h							

Table 38-90. USBRQPKTCOUNT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-0	COUNT	R/W	0h	Block Transfer Packet Count sets the number of packets of the size defined by the MAXLOAD bit field that are to be transferred in a block transfer. Note: This is only used in Host mode when AUTORQ is set. The bit has no effect in Device mode or when AUTORQ is not set. Reset type: SYSRSn

38.6.2.86 USBRXDPKTBUFDIS Register (Offset = 1A0h) [Reset = 0h]

USBRXDPKTBUFDIS is shown in Figure 38-88 and described in Table 38-91.

Return to the Summary Table.

USB Receive Double Packet Buffer Disable

Figure 38-88. USBRXDPKTBUFDIS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				EP3	EP2	EP1	RESERVED
R-0h				R-0h	R-0h	R-0h	R-0h

Table 38-91. USBRXDPKTBUFDIS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	EP3	R	0h	EP3 RX Double Packet Buffer Disable Reset type: SYSRSn 0h (R/W) = Disables double-packet buffering. 1h (R/W) = Enables double-packet buffering.
2	EP2	R	0h	EP2 RX Double Packet Buffer Disable Reset type: SYSRSn 0h (R/W) = Disables double-packet buffering. 1h (R/W) = Enables double-packet buffering.
1	EP1	R	0h	EP1 RX Double Packet Buffer Disable Reset type: SYSRSn 0h (R/W) = Disables double-packet buffering. 1h (R/W) = Enables double-packet buffering.
0	RESERVED	R	0h	Reserved

38.6.2.87 USBTXDPKTBUFDIS Register (Offset = 1A1h) [Reset = 0h]

USBTXDPKTBUFDIS is shown in Figure 38-89 and described in Table 38-92.

Return to the Summary Table.

USB Transmit Double Packet Buffer Disable

Figure 38-89. USBTXDPKTBUFDIS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				EP3	EP2	EP1	RESERVED
R-0h				R-0h	R-0h	R-0h	R-0h

Table 38-92. USBTXDPKTBUFDIS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	EP3	R	0h	EP3 TX Double Packet Buffer Disable Reset type: SYSRSn 0h (R/W) = Disables double-packet buffering. 1h (R/W) = Enables double-packet buffering.
2	EP2	R	0h	EP2 TX Double Packet Buffer Disable Reset type: SYSRSn 0h (R/W) = Disables double-packet buffering. 1h (R/W) = Enables double-packet buffering.
1	EP1	R	0h	EP1 TX Double Packet Buffer Disable Reset type: SYSRSn 0h (R/W) = Disables double-packet buffering. 1h (R/W) = Enables double-packet buffering.
0	RESERVED	R	0h	Reserved

38.6.2.88 USBEPC Register (Offset = 200h) [Reset = 0h]

USBEPC is shown in Figure 38-90 and described in Table 38-93.

Return to the Summary Table.

USB External Power Control

Figure 38-90. USBEPC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						PFLTACT	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
RESERVED	PFLTAEN	PFLTSEN	PFLTEN	RESERVED	EPENDE	EPEN	
R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	

Table 38-93. USBEPC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-10	RESERVED	R	0h	Reserved
9-8	PFLTACT	R/W	0h	Power Fault Action. This bit field specifies how the USB0EPEN signal is changed when detecting a USB power fault Reset type: SYSRSn 0h (R/W) = Unchanged. USB0EPEN is controlled by the combination of the EPEN and EPENDE bits. 1h (R/W) = Tristate. USB0EPEN is undriven (tristate). 2h (R/W) = Low. USB0EPEN is driven Low. 3h (R/W) = High. USB0EPEN is driven High.
7	RESERVED	R	0h	Reserved
6	PFLTAEN	R/W	0h	Power Fault Action Enable. This bit specifies whether a USB power fault triggers any automatic corrective action regarding the driven state of the USB0EPEN signal. Reset type: SYSRSn 0h (R/W) = Disabled. USB0EPEN is controlled by the combination of the EPEN and EPENDE bits. 1h (R/W) = Enabled. The USB0EPEN output is automatically changed to the state specified by the PFLTACT field.
5	PFLTSEN	R/W	0h	Power Fault Sense. This bit specifies the logical sense of the USB0PFLT input signal that indicates an error condition. The complementary state is the inactive state. Reset type: SYSRSn 0h (R/W) = Low Fault. If USB0PFLT is driven Low, the power fault is signaled internally (if enabled by the PFLTEN bit). 1h (R/W) = High Fault. If USB0PFLT is driven High, the power fault is signaled internally (if enabled by the PFLTEN bit).
4	PFLTEN	R/W	0h	Power Fault Input Enable. This bit specifies whether the USB0PFLT input signal is used in internal logic. Reset type: SYSRSn 0h (R/W) = Not Used. The USB0PFLT signal is ignored. 1h (R/W) = Used. The USB0PFLT signal is used internally

Table 38-93. USBEPC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	RESERVED	R	0h	Reserved
2	EPENDE	R/W	0h	EPEN Drive Enable. This bit specifies whether the USB0EPEN signal is driven or undriven (tristate). When driven, the signal value is specified by the EPEN field. When not driven, the EPEN field is ignored and the USB0EPEN signal is placed in a high-impedance state. The USB0EPEN signal is undriven at reset because the sense of the external power supply enable is unknown. By adding the high-impedance state, system designers can bias the power supply enable to the disabled state using a large resistor (100 kOhm) and later configure and drive the output signal to enable the power supply. Reset type: SYSRSn 0h (R/W) = Not Driven. The USB0EPEN signal is high impedance. 1h (R/W) = Driven. The USB0EPEN signal is driven to the logical value specified by the value of the EPEN field.
1-0	EPEN	R/W	0h	External Power Supply Enable Configuration. This bit field specifies and controls the logical value driven on the USB0EPEN signal. Reset type: SYSRSn 0h (R/W) = Power Enable Active Low. The USB0EPEN signal is driven Low if the EPENDE bit is set. 1h (R/W) = Power Enable Active High. The USB0EPEN signal is driven High if the EPENDE bit is set. 2h (R/W) = Power Enable High if VBUS Low. The USB0EPEN signal is driven High when the A device is not recognized. 3h (R/W) = Power Enable High if VBUS High. The USB0EPEN signal is driven High when the A device is recognized.

38.6.2.89 USBEPCRIS Register (Offset = 202h) [Reset = 0h]

USBEPCRIS is shown in Figure 38-91 and described in Table 38-94.

Return to the Summary Table.

USB External Power Control Raw Interrupt Status

Figure 38-91. USBEPCRIS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															PF
R-0h															R-0h

Table 38-94. USBEPCRIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	PF	R	0h	USB Power Fault Interrupt Status. This bit is cleared by writing a 1 to the PF bit in the USBEPCISC register Reset type: SYSRSn 0h (R/W) = A Power Fault status has been detected. 1h (R/W) = An interrupt has not occurred.

38.6.2.90 USBEPCIM Register (Offset = 204h) [Reset = 0h]

USBEPCIM is shown in Figure 38-92 and described in Table 38-95.

Return to the Summary Table.

USB External Power Control Interrupt Mask

Figure 38-92. USBEPCIM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															PF
R-0h															R-0h

Table 38-95. USBEPCIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	PF	R	0h	USB Power Fault Interrupt Mask. Reset type: SYSRSn 0h (R/W) = The raw interrupt signal from a detected power fault is sent to the interrupt controller. 1h (R/W) = A detected power fault does not affect the interrupt status.

38.6.2.91 USBEPCISC Register (Offset = 206h) [Reset = 0h]

USBEPCISC is shown in Figure 38-93 and described in Table 38-96.

Return to the Summary Table.

USB External Power Control Interrupt Status and Clear

Figure 38-93. USBEPCISC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															PF
R-0h															R-0h

Table 38-96. USBEPCISC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	PF	R	0h	Power Fault Interrupt Status and Clear This bit is cleared by writing a 1. Clearing this bit also clears the PF bit in the USBEPCISC register. Reset type: SYSRSn 0h (R/W) = The PF bits in the USBEPCRIS and USBEPCIM registers are set, providing an interrupt to the interrupt controller 1h (R/W) = No interrupt has occurred or the interrupt is masked.

38.6.2.92 USBDRRIS Register (Offset = 208h) [Reset = 0h]

USBDRRIS is shown in Figure 38-94 and described in Table 38-97.

Return to the Summary Table.

USB Device RESUME Raw Interrupt Status

Figure 38-94. USBDRRIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							RESUME
R-0h							R-0h

Table 38-97. USBDRRIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	RESUME	R	0h	RESUME Interrupt Status This bit is cleared by writing a 1 to the RESUME bit in the USBDRISC register. Reset type: SYSRSn 0h (R/W) = A RESUME status has been detected. 1h (R/W) = An interrupt has not occurred.

38.6.2.93 USBDRIM Register (Offset = 20Ah) [Reset = 0h]

USBDRIM is shown in Figure 38-95 and described in Table 38-98.

Return to the Summary Table.

USB Device RESUME Interrupt Mask

Figure 38-95. USBDRIM Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							RESUME
R-0h							R-0h

Table 38-98. USBDRIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	RESUME	R	0h	Resume Interrupt Mask Reset type: SYSRSn 0h (R/W) = The raw interrupt signal from a detected RESUME is sent to the interrupt controller. This bit should only be set when a SUSPEND has been detected (the SUSPEND bit in the USBIS register is set). 1h (R/W) = A detected RESUME does not affect the interrupt status.

38.6.2.94 USBDRISC Register (Offset = 20Ch) [Reset = 0h]

USBDRISC is shown in Figure 38-96 and described in Table 38-99.

Return to the Summary Table.

USB Device RESUME Interrupt Status and Clear

Figure 38-96. USBDRISC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							RESUME
R-0h							W1C-0h

Table 38-99. USBDRISC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	RESUME	W1C	0h	RESUME Interrupt Status and Clear. This bit is cleared by writing a 1. Clearing this bit also clears the RESUME bit in the USBDRCRIS register Reset type: SYSRSn 0h (R/W) = The RESUME bits in the USBDRRIS and USBDRCIM registers are set, providing an interrupt to the interrupt controller. 1h (R/W) = No interrupt has occurred or the interrupt is masked.

38.6.2.95 USBGPCS Register (Offset = 20Eh) [Reset = 0h]

USBGPCS is shown in Figure 38-97 and described in Table 38-100.

Return to the Summary Table.

USB General-Purpose Control and Status

Figure 38-97. USBGPCS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						DEVMODOTG	DEVMOD
R-0h						R/W-0h	R/W-0h

Table 38-100. USBGPCS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-2	RESERVED	R	0h	Reserved
1	DEVMODOTG	R/W	0h	Enable Device Mode. This bit enables the DEVMOD bit to control the state of the internal ID signal in G OTG mode. Reset type: SYSRSn 0h (R/W) = The RESUME bits in the USBDRRIS and USBDRCIM registers are set, providing an interrupt to the interrupt controller. 1h (R/W) = No interrupt has occurred or the interrupt is masked.
0	DEVMOD	R/W	0h	Device Mode This bit specifies the state of the internal ID signal in Host mode and in OTG mode when the DEVMODOTG bit is set. In Device mode this bit is ignored (assumed set). Reset type: SYSRSn 0h (R/W) = The RESUME bits in the USBDRRIS and USBDRCIM registers are set, providing an interrupt to the interrupt controller. 1h (R/W) = No interrupt has occurred or the interrupt is masked.

38.6.2.96 USBVDC Register (Offset = 218h) [Reset = 0h]

USBVDC is shown in Figure 38-98 and described in Table 38-101.

Return to the Summary Table.

USB VBUS Droop Control

Figure 38-98. USBVDC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							VBDEN
R-0h							R/W-0h

Table 38-101. USBVDC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	VBDEN	R/W	0h	Vbus Droop Enable Reset type: SYSRSn

38.6.2.97 USBVDCRIS Register (Offset = 21Ah) [Reset = 0h]

USBVDCRIS is shown in Figure 38-99 and described in Table 38-102.

Return to the Summary Table.

USB VBUS Droop Control Raw Interrupt Status

Figure 38-99. USBVDCRIS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															VD
R-0h															R-0h

Table 38-102. USBVDCRIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	VD	R	0h	Vbus Droop Raw Interrupt Status Reset type: SYSRSn

38.6.2.98 USBVDCIM Register (Offset = 21Ch) [Reset = 0h]

USBVDCIM is shown in Figure 38-100 and described in Table 38-103.

Return to the Summary Table.

USB VBUS Droop Control Interrupt Mask

Figure 38-100. USBVDCIM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															VD
R-0h															R-0h

Table 38-103. USBVDCIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	VD	R	0h	Vbus Droop Interrupt Mask Reset type: SYSRSn

38.6.2.99 USBVDCISC Register (Offset = 21Eh) [Reset = 0h]

USBVDCISC is shown in Figure 38-101 and described in Table 38-104.

Return to the Summary Table.

USB VBUS Droop Control Interrupt Status and Clear

Figure 38-101. USBVDCISC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															VD
R-0h															W1C-0h

Table 38-104. USBVDCISC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	VD	W1C	0h	Vbus Droop Interrupt Status and Clear Reset type: SYSRSn

38.6.2.100 USBIDVRIS Register (Offset = 222h) [Reset = 0h]

USBIDVRIS is shown in Figure 38-102 and described in Table 38-105.

Return to the Summary Table.

USB ID Valid Detect Raw Interrupt Status

Figure 38-102. USBIDVRIS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ID
R-0h															W1C-0h

Table 38-105. USBIDVRIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	ID	W1C	0h	ID Valid Detect Raw Interrupt Status Reset type: SYSRSn

38.6.2.101 USBIDVIM Register (Offset = 224h) [Reset = 0h]

USBIDVIM is shown in Figure 38-103 and described in Table 38-106.

Return to the Summary Table.

USB ID Valid Detect Interrupt Mask

Figure 38-103. USBIDVIM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ID
R-0h															W1C-0h

Table 38-106. USBIDVIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	ID	W1C	0h	ID Valid Detect Interrupt mask Reset type: SYSRSn

38.6.2.102 USBIDVISC Register (Offset = 226h) [Reset = 0h]

USBIDVISC is shown in Figure 38-104 and described in Table 38-107.

Return to the Summary Table.

USB ID Valid Detect Interrupt Status and Clear

Figure 38-104. USBIDVISC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ID
R-0h															W1C-0h

Table 38-107. USBIDVISC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	ID	W1C	0h	ID Valid Detect Interrupt Status and Clear Reset type: SYSRSn

38.6.2.103 USBDMASEL Register (Offset = 228h) [Reset = 0h]

USBDMASEL is shown in Figure 38-105 and described in Table 38-108.

Return to the Summary Table.

USB DMA Select

Figure 38-105. USBDMASEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								DMACTX				DMACRX			
R-0h								R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMABTX				DMABRX				DMAATX				DMAARX			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 38-108. USBDMASEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-20	DMACTX	R/W	0h	DMA C TX Select specifies the TX mapping of the third USB endpoint on DMA channel 5 Reset type: SYSRSn 0h (R/W) = Reserved 1h (R/W) = Endpoint 1 TX 2h (R/W) = Endpoint 2 TX 3h (R/W) = Endpoint 3 TX
19-16	DMACRX	R/W	0h	DMA C RX Select specifies the RX and TX mapping of the third USB endpoint on DMA channel 4 Reset type: SYSRSn 0h (R/W) = Reserved 1h (R/W) = Endpoint 1 RX 2h (R/W) = Endpoint 2 RX 3h (R/W) = Endpoint 3 RX
15-12	DMABTX	R/W	0h	DMA B TX Select specifies the TX mapping of the second USB endpoint on DMA channel 3 Reset type: SYSRSn 0h (R/W) = Reserved 1h (R/W) = Endpoint 1 TX 2h (R/W) = Endpoint 2 TX 3h (R/W) = Endpoint 3 TX
11-8	DMABRX	R/W	0h	DMA B RX Select Specifies the RX mapping of the second USB endpoint on DMA channel 2 Reset type: SYSRSn 0h (R/W) = Reserved 1h (R/W) = Endpoint 1 RX 2h (R/W) = Endpoint 2 RX 3h (R/W) = Endpoint 3 RX
7-4	DMAATX	R/W	0h	DMA A TX Select specifies the TX mapping of the first USB endpoint on DMA channel 1 Reset type: SYSRSn 0h (R/W) = Reserved 1h (R/W) = Endpoint 1 TX 2h (R/W) = Endpoint 2 TX 3h (R/W) = Endpoint 3 TX
3-0	DMAARX	R/W	0h	DMA A RX Select specifies the RX mapping of the first USB endpoint on DMA channel 0 Reset type: SYSRSn 0h (R/W) = Reserved 1h (R/W) = Endpoint 1 RX 2h (R/W) = Endpoint 2 RX 3h (R/W) = Endpoint 3 RX

38.6.2.104 USB_GLB_INT_EN Register (Offset = 240h) [Reset = 0h]

USB_GLB_INT_EN is shown in Figure 38-106 and described in Table 38-109.

Return to the Summary Table.

USB Global Interrupt Enable Register
Note: This Register is applicable only when USB is mapped to CPU1

Figure 38-106. USB_GLB_INT_EN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTEN
R-0h							R/W-0h

Table 38-109. USB_GLB_INT_EN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	INTEN	R/W	0h	1: Interrupt enabled, USB interrupt is passed on. 0: Interrupt disabled, USB interrupt is blocked. Reset type: SYSRSn

38.6.2.105 USB_GLB_INT_FLG Register (Offset = 242h) [Reset = 0h]

USB_GLB_INT_FLG is shown in Figure 38-107 and described in Table 38-110.

Return to the Summary Table.

USB Global Interrupt Flag Register
Note: This Register is applicable only when USB is mapped to CPU1

Figure 38-107. USB_GLB_INT_FLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTFLG
R-0h							R/W-0h

Table 38-110. USB_GLB_INT_FLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	INTFLG	R/W	0h	1: Once USB interrupt has been fired, no other interrupt will be fired unless this flag is cleared by writing to USB_GLB_INT_FLG_CLR register. 0: No interrupt has been fired. Reset type: SYSRSn

38.6.2.106 USB_GLB_INT_FLG_CLR Register (Offset = 244h) [Reset = 0h]

USB_GLB_INT_FLG_CLR is shown in Figure 38-108 and described in Table 38-111.

Return to the Summary Table.

USB Global Interrupt Flag Clear Register
Note: This Register is applicable only when USB is mapped to CPU1

Figure 38-108. USB_GLB_INT_FLG_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTFLG
R-0h							R-0/W1S-0h

Table 38-111. USB_GLB_INT_FLG_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	INTFLG	R-0/W1S	0h	Write of 1 to this field clears the corresponding bit in USB_GLB_INT_FLG register. Write of 0 has no effect. Reset type: SYSRSn

38.6.2.107 USBDMARIS Register (Offset = 280h) [Reset = 0h]

USBDMARIS is shown in Figure 38-109 and described in Table 38-112.

Return to the Summary Table.

USB uDMA Raw Interrupt Status register.
Note: This Register is applicable only when USB is mapped to CM

Figure 38-109. USBDMARIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		USB_DMAC_TX_DONE	USB_DMAC_RX_DONE	USB_DMAB_TX_DONE	USB_DMAB_RX_DONE	USB_DMAA_TX_DONE	USB_DMAA_Rx_DONE
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 38-112. USBDMARIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	USB_DMAC_TX_DONE	R	0h	1: USB uDMA transfer complete indication of USB_DMAC_TX trigger 0: No USB uDMA transfer complete indication of USB_DMAC_TXtrigger Reset type: PER.RESET
4	USB_DMAC_RX_DONE	R	0h	1: USB uDMA transfer complete indication of USB_DMAC_RX trigger 0: No USB uDMA transfer complete indication of USB_DMAC_RXtrigger Reset type: PER.RESET
3	USB_DMAB_TX_DONE	R	0h	1: USB uDMA transfer complete indication of USB_DMAB_TX trigger 0: No USB uDMA transfer complete indication of USB_DMAB_TXtrigger Reset type: PER.RESET
2	USB_DMAB_RX_DONE	R	0h	1: USB uDMA transfer complete indication of USB_DMAB_RX trigger 0: No USB uDMA transfer complete indication of USB_DMAB_RXtrigger Reset type: PER.RESET
1	USB_DMAA_TX_DONE	R	0h	1: USB uDMA transfer complete indication of USB_DMAA_TX trigger 0: No USB uDMA transfer complete indication of USB_DMAA_TXtrigger Reset type: PER.RESET
0	USB_DMAA_Rx_DONE	R	0h	1: USB uDMA transfer complete indication of USB_DMAA_Rx trigger 0: No USB uDMA transfer complete indication of USB_DMAA_Rxtrigger Reset type: PER.RESET

38.6.2.108 USBDMAIM Register (Offset = 282h) [Reset = 3Fh]

USBDMAIM is shown in Figure 38-110 and described in Table 38-113.

Return to the Summary Table.

USB uDMA Interrupt Mask Register
Note: This Register is applicable only when USB is mapped to CM

Figure 38-110. USBDMAIM Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		USB_DMAC_TX_DONE	USB_DMAC_RX_DONE	USB_DMAB_TX_DONE	USB_DMAB_RX_DONE	USB_DMAA_TX_DONE	USB_DMAA_Rx_DONE
R-0h		R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 38-113. USBDMAIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	USB_DMAC_TX_DONE	R/W	1h	0: USB_DMAC_TX_DONE does not trigger a USB interrupt. 1: USB_DMAC_TX_DONE triggers a USB interrupt. Note: The reset value of this bit is 1 to keep compatibility with Concerto USB software model. Reset type: PER.RESET
4	USB_DMAC_RX_DONE	R/W	1h	0: USB_DMAC_RX_DONE does not trigger a USB interrupt. 1: USB_DMAC_RX_DONE triggers a USB interrupt. Note: The reset value of this bit is 1 to keep compatibility with Concerto USB software model. Reset type: PER.RESET
3	USB_DMAB_TX_DONE	R/W	1h	0: USB_DMAB_TX_DONE does not trigger a USB interrupt. 1: USB_DMAB_TX_DONE triggers a USB interrupt. Note: The reset value of this bit is 1 to keep compatibility with Concerto USB software model. Reset type: PER.RESET
2	USB_DMAB_RX_DONE	R/W	1h	0: USB_DMAB_RX_DONE does not trigger a USB interrupt. 1: USB_DMAB_RX_DONE triggers a USB interrupt. Note: The reset value of this bit is 1 to keep compatibility with Concerto USB software model. Reset type: PER.RESET
1	USB_DMAA_TX_DONE	R/W	1h	0: USB_DMAA_TX_DONE does not trigger a USB interrupt. 1: USB_DMAA_TX_DONE triggers a USB interrupt. Note: The reset value of this bit is 1 to keep compatibility with Concerto USB software model. Reset type: PER.RESET
0	USB_DMAA_Rx_DONE	R/W	1h	0: USB_DMAA_Rx_DONE does not trigger a USB interrupt. 1: USB_DMAA_Rx_DONE triggers a USB interrupt. Note: The reset value of this bit is 1 to keep compatibility with Concerto USB software model. Reset type: PER.RESET

38.6.2.109 USBDMAISC Register (Offset = 284h) [Reset = 3Fh]

USBDMAISC is shown in Figure 38-111 and described in Table 38-114.

Return to the Summary Table.

USB uDMA Interrupt Status and Clear Register
Note: This Register is applicable only when USB is mapped to CM

Figure 38-111. USBDMAISC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		USB_DMAC_TX_DONE	USB_DMAC_RX_DONE	USB_DMAB_TX_DONE	USB_DMAB_RX_DONE	USB_DMAA_TX_DONE	USB_DMAA_Rx_DONE
R-0h		R/W1S-1h	R/W1S-1h	R/W1S-1h	R/W1S-1h	R/W1S-1h	R/W1S-1h

Table 38-114. USBDMAISC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	USB_DMAC_TX_DONE	R/W1S	1h	0: USB_DMAC_TX_DONE has not triggered a USB interrupt. 1: USB_DMAC_TX_DONE triggered a USB interrupt. Note: This bit is cleared by writing a 1. Clearing this bit also clears the DMADONE bit in the USBDMARIS register. Reset type: PER.RESET
4	USB_DMAC_RX_DONE	R/W1S	1h	0: USB_DMAC_RX_DONE has not triggered a USB interrupt. 1: USB_DMAC_RX_DONE triggered a USB interrupt. Note: This bit is cleared by writing a 1. Clearing this bit also clears the DMADONE bit in the USBDMARIS register. Reset type: PER.RESET
3	USB_DMAB_TX_DONE	R/W1S	1h	0: USB_DMAB_TX_DONE has not triggered a USB interrupt. 1: USB_DMAB_TX_DONE triggered a USB interrupt. Note: This bit is cleared by writing a 1. Clearing this bit also clears the DMADONE bit in the USBDMARIS register. Reset type: PER.RESET
2	USB_DMAB_RX_DONE	R/W1S	1h	0: USB_DMAB_RX_DONE has not triggered a USB interrupt. 1: USB_DMAB_RX_DONE triggered a USB interrupt. Note: This bit is cleared by writing a 1. Clearing this bit also clears the DMADONE bit in the USBDMARIS register. Reset type: PER.RESET
1	USB_DMAA_TX_DONE	R/W1S	1h	0: USB_DMAA_TX_DONE has not triggered a USB interrupt. 1: USB_DMAA_TX_DONE triggered a USB interrupt. Note: This bit is cleared by writing a 1. Clearing this bit also clears the DMADONE bit in the USBDMARIS register. Reset type: PER.RESET

Table 38-114. USBDMAISC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	USB_DMAA_Rx_DONE	R/W1S	1h	0: USB_DMAA_Rx_DONE has not triggered a USB interrupt. 1: USB_DMAA_Rx_DONE triggered a USB interrupt. Note: This bit is cleared by writing a 1. Clearing this bit also clears the DMADONE bit in the USBDMARIS register. Reset type: PER.RESET

38.6.3 USB Registers to Driverlib Functions

Table 38-115. USB Registers to Driverlib Functions

File	Driverlib Function
FADDR	
usb.c	voidUSBDevAddrSet
usb.c	uint32_tUSBDevAddrGet
POWER	
usb.c	voidUSBHostSuspend
usb.c	voidUSBHostReset
usb.c	voidUSBHostResume
usb.c	voidUSBDevConnect
usb.c	voidUSBDevDisconnect
usb.c	voidUSBPHYPowerOff
usb.c	voidUSBPHYPowerOn
TXIS	
usb.c	uint32_tUSBIntStatus
usb.c	uint32_tUSBIntStatusEndpoint
RXIS	
usb.c	uint32_tUSBIntStatus
usb.c	uint32_tUSBIntStatusEndpoint
TXIE	
usb.c	voidUSBIntDisableEndpoint
usb.c	voidUSBIntEnableEndpoint
RXIE	
usb.c	voidUSBIntDisableEndpoint
usb.c	voidUSBIntEnableEndpoint
IS	
usb.c	uint32_tUSBIntStatus
usb.c	uint32_tUSBIntStatusControl
IE	
usb.c	voidUSBIntDisableControl
usb.c	voidUSBIntEnableControl
FRAME	
usb.c	uint32_tUSBFrameNumberGet
EPIDX	
usb.c	staticvoid_USBIndexWrite
usb.c	staticuint32_t_USBIndexRead
TEST	
-	
FIFO0	

Table 38-115. USB Registers to Driverlib Functions (continued)

File	Driverlib Function
usb.c	int32_tUSBEndpointDataGet
usb.c	int32_tUSBEndpointDataPut
usb.c	uint32_tUSBFIFOAddrGet
FIFO1	
-	
FIFO2	
-	
FIFO3	
-	
FIFO4	
-	
FIFO5	
-	
FIFO6	
-	
FIFO7	
-	
FIFO8	
-	
FIFO9	
-	
FIFO10	
-	
FIFO11	
-	
FIFO12	
-	
FIFO13	
-	
FIFO14	
-	
FIFO15	
-	
DEVCTL	
usb.c	uint32_tUSBHostSpeedGet
usb.c	voidUSBOTGSessionRequest
usb.c	uint32_tUSBModeGet
TXFIFOSZ	
usb.c	voidUSBFIFOConfigSet
usb.c	voidUSBFIFOConfigGet
RXFIFOSZ	
usb.c	voidUSBFIFOConfigSet
usb.c	voidUSBFIFOConfigGet
TXFIFOADD	
usb.c	voidUSBFIFOConfigSet

Table 38-115. USB Registers to Driverlib Functions (continued)

File	Driverlib Function
usb.c	voidUSBFIFOConfigGet
RXFIFOADD	
usb.c	voidUSBFIFOConfigSet
usb.c	voidUSBFIFOConfigGet
CONTIM	
-	
VPLEN	
-	
FSEOF	
-	
LSEOF	
-	
TXFUNCADDR0	
usb.c	voidUSBHostAddrSet
usb.c	uint32_tUSBHostAddrGet
TXHUBADDR0	
usb.c	voidUSBHostHubAddrSet
usb.c	uint32_tUSBHostHubAddrGet
TXHUBPORT0	
-	
TXFUNCADDR1	
-	
TXHUBADDR1	
-	
TXHUBPORT1	
-	
RXFUNCADDR1	
-	
RXHUBADDR1	
-	
RXHUBPORT1	
-	
TXFUNCADDR2	
-	
TXHUBADDR2	
-	
TXHUBPORT2	
-	
RXFUNCADDR2	
-	
RXHUBADDR2	
-	
RXHUBPORT2	
-	
TXFUNCADDR3	

Table 38-115. USB Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
TXHUBADDR3	
-	
TXHUBPORT3	
-	
RXFUNCADDR3	
-	
RXHUBADDR3	
-	
RXHUBPORT3	
-	
TXFUNCADDR4	
-	
TXHUBADDR4	
-	
TXHUBPORT4	
-	
RXFUNCADDR4	
-	
RXHUBADDR4	
-	
RXHUBPORT4	
-	
TXFUNCADDR5	
-	
TXHUBADDR5	
-	
TXHUBPORT5	
-	
RXFUNCADDR5	
-	
RXHUBADDR5	
-	
RXHUBPORT5	
-	
TXFUNCADDR6	
-	
TXHUBADDR6	
-	
TXHUBPORT6	
-	
RXFUNCADDR6	
-	
RXHUBADDR6	
-	

Table 38-115. USB Registers to Driverlib Functions (continued)

File	Driverlib Function
RXHUBPORT6	
-	
TXFUNCADDR7	
-	
TXHUBADDR7	
-	
TXHUBPORT7	
-	
RXFUNCADDR7	
-	
RXHUBADDR7	
-	
RXHUBPORT7	
-	
TXFUNCADDR8	
-	
TXHUBADDR8	
-	
TXHUBPORT8	
-	
RXFUNCADDR8	
-	
RXHUBADDR8	
-	
RXHUBPORT8	
-	
TXFUNCADDR9	
-	
TXHUBADDR9	
-	
TXHUBPORT9	
-	
RXFUNCADDR9	
-	
RXHUBADDR9	
-	
RXHUBPORT9	
-	
TXFUNCADDR10	
-	
TXHUBADDR10	
-	
TXHUBPORT10	
-	
RXFUNCADDR10	

Table 38-115. USB Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
RXHUBADDR10	
-	
RXHUBPORT10	
-	
TXFUNCADDR11	
-	
TXHUBADDR11	
-	
TXHUBPORT11	
-	
RXFUNCADDR11	
-	
RXHUBADDR11	
-	
RXHUBPORT11	
-	
TXFUNCADDR12	
-	
TXHUBADDR12	
-	
TXHUBPORT12	
-	
RXFUNCADDR12	
-	
RXHUBADDR12	
-	
RXHUBPORT12	
-	
TXFUNCADDR13	
-	
TXHUBADDR13	
-	
TXHUBPORT13	
-	
RXFUNCADDR13	
-	
RXHUBADDR13	
-	
RXHUBPORT13	
-	
TXFUNCADDR14	
-	
TXHUBADDR14	
-	

Table 38-115. USB Registers to Driverlib Functions (continued)

File	Driverlib Function
TXHUBPORT14	
-	
RXFUNCADDR14	
-	
RXHUBADDR14	
-	
RXHUBPORT14	
-	
TXFUNCADDR15	
-	
TXHUBADDR15	
-	
TXHUBPORT15	
-	
RXFUNCADDR15	
-	
RXHUBADDR15	
-	
RXHUBPORT15	
-	
CSRL0	
usb.c	voidUSBHostEndpointStatusClear
usb.c	voidUSBDevEndpointStatusClear
usb.c	voidUSBDevEndpointStall
usb.c	voidUSBDevEndpointStallClear
usb.c	uint32_tUSBEndpointDataAvail
usb.c	int32_tUSBEndpointDataGet
usb.c	voidUSBDevEndpointDataAck
usb.c	voidUSBHostEndpointDataAck
usb.c	int32_tUSBEndpointDataPut
usb.c	int32_tUSBEndpointDataSend
usb.c	voidUSBFIFOFlush
usb.c	voidUSBHostRequestIN
usb.c	voidUSBHostRequestINClear
usb.c	voidUSBHostRequestStatus
CSRH0	
usb.c	voidUSBHostEndpointDataToggle
usb.c	voidUSBFIFOFlush
COUNT0	
usb.c	uint32_tUSBEndpointDataAvail
usb.c	int32_tUSBEndpointDataGet
TYPE0	
usb.c	voidUSBHostEndpointConfig
usb.c	voidUSBHostHubAddrSet
NAKLMT	

Table 38-115. USB Registers to Driverlib Functions (continued)

File	Driverlib Function
usb.c	voidUSBHostEndpointConfig
TXMAXP1	
usb.c	voidUSBHostEndpointConfig
usb.c	voidUSBDevEndpointConfigSet
usb.c	voidUSBDevEndpointConfigGet
TXCSRL1	
usb.c	uint32_tUSBEndpointStatus
usb.c	voidUSBHostEndpointStatusClear
usb.c	voidUSBDevEndpointStatusClear
usb.c	voidUSBEndpointDataToggleClear
usb.c	voidUSBDevEndpointStall
usb.c	voidUSBDevEndpointStallClear
usb.c	voidUSBDevEndpointConfigSet
usb.c	voidUSBFIFOFlush
TXCSRH1	
usb.c	voidUSBHostEndpointDataToggle
usb.c	voidUSBHostEndpointConfig
usb.c	voidUSBDevEndpointConfigSet
usb.c	voidUSBDevEndpointConfigGet
usb.c	voidUSBEndpointDMAConfigSet
usb.c	voidUSBEndpointDMAEnable
usb.c	voidUSBEndpointDMADisable
RXMAXP1	
usb.c	voidUSBHostEndpointConfig
usb.c	voidUSBDevEndpointConfigSet
usb.c	voidUSBDevEndpointConfigGet
RXCSRL1	
usb.c	uint32_tUSBEndpointStatus
usb.c	voidUSBHostEndpointStatusClear
usb.c	voidUSBDevEndpointStatusClear
usb.c	voidUSBEndpointDataToggleClear
usb.c	voidUSBDevEndpointStall
usb.c	voidUSBDevEndpointStallClear
usb.c	voidUSBDevEndpointConfigSet
usb.c	uint32_tUSBEndpointDataAvail
usb.c	int32_tUSBEndpointDataGet
usb.c	voidUSBDevEndpointDataAck
usb.c	voidUSBHostEndpointDataAck
usb.c	voidUSBFIFOFlush
usb.c	voidUSBHostRequestIN
usb.c	voidUSBHostRequestINClear
RXCSRH1	
usb.c	voidUSBHostEndpointDataToggle
usb.c	voidUSBHostEndpointConfig
usb.c	voidUSBDevEndpointConfigSet

Table 38-115. USB Registers to Driverlib Functions (continued)

File	Driverlib Function
usb.c	voidUSBDevEndpointConfigGet
usb.c	voidUSBEndpointDMAConfigSet
usb.c	voidUSBEndpointDMAEnable
usb.c	voidUSBEndpointDMADisable
RXCOUNT1	
-	
TXTYPE1	
usb.c	voidUSBHostEndpointConfig
TXINTERVAL1	
usb.c	voidUSBHostEndpointConfig
RXTYPE1	
usb.c	voidUSBHostEndpointConfig
RXINTERVAL1	
usb.c	voidUSBHostEndpointConfig
TXMAXP2	
-	
TXCSRL2	
-	
TXCSRH2	
-	
RXMAXP2	
-	
RXCSRL2	
-	
RXCSRH2	
-	
RXCOUNT2	
-	
TXTYPE2	
-	
TXINTERVAL2	
-	
RXTYPE2	
-	
RXINTERVAL2	
-	
TXMAXP3	
-	
TXCSRL3	
-	
TXCSRH3	
-	
RXMAXP3	
-	
RXCSRL3	

Table 38-115. USB Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
RXCSRH3	
-	
RXCOUNT3	
-	
TXTYPE3	
-	
TXINTERVAL3	
-	
RXTYPE3	
-	
RXINTERVAL3	
-	
TXMAXP4	
-	
TXCSRL4	
-	
TXCSRH4	
-	
RXMAXP4	
-	
RXCSRL4	
-	
RXCSRH4	
-	
RXCOUNT4	
-	
TXTYPE4	
-	
TXINTERVAL4	
-	
RXTYPE4	
-	
RXINTERVAL4	
-	
TXMAXP5	
-	
TXCSRL5	
-	
TXCSRH5	
-	
RXMAXP5	
-	
RXCSRL5	
-	

Table 38-115. USB Registers to Driverlib Functions (continued)

File	Driverlib Function
RXCSRH5	
-	
RXCOUNT5	
-	
TXTYPE5	
-	
TXINTERVAL5	
-	
RXTYPE5	
-	
RXINTERVAL5	
-	
TXMAXP6	
-	
TXCSRL6	
-	
TXCSRH6	
-	
RXMAXP6	
-	
RXCSRL6	
-	
RXCSRH6	
-	
RXCOUNT6	
-	
TXTYPE6	
-	
TXINTERVAL6	
-	
RXTYPE6	
-	
RXINTERVAL6	
-	
TXMAXP7	
-	
TXCSRL7	
-	
TXCSRH7	
-	
RXMAXP7	
-	
RXCSRL7	
-	
RXCSRH7	

Table 38-115. USB Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
RXCOUNT7	
-	
TXTYPE7	
-	
TXINTERVAL7	
-	
RXTYPE7	
-	
RXINTERVAL7	
-	
TXMAXP8	
-	
TXCSRL8	
-	
TXCSRH8	
-	
RXMAXP8	
-	
RXCSRL8	
-	
RXCSRH8	
-	
RXCOUNT8	
-	
TXTYPE8	
-	
TXINTERVAL8	
-	
RXTYPE8	
-	
RXINTERVAL8	
-	
TXMAXP9	
-	
TXCSRL9	
-	
TXCSRH9	
-	
RXMAXP9	
-	
RXCSRL9	
-	
RXCSRH9	
-	

Table 38-115. USB Registers to Driverlib Functions (continued)

File	Driverlib Function
RXCOUNT9	
-	
TXTYPE9	
-	
TXINTERVAL9	
-	
RXTYPE9	
-	
RXINTERVAL9	
-	
TXMAXP10	
-	
TXCSRL10	
-	
TXCSRH10	
-	
RXMAXP10	
-	
RXCSRL10	
-	
RXCSRH10	
-	
RXCOUNT10	
-	
TXTYPE10	
-	
TXINTERVAL10	
-	
RXTYPE10	
-	
RXINTERVAL10	
-	
TXMAXP11	
-	
TXCSRL11	
-	
TXCSRH11	
-	
RXMAXP11	
-	
RXCSRL11	
-	
RXCSRH11	
-	
RXCOUNT11	

Table 38-115. USB Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
TXTYPE11	
-	
TXINTERVAL11	
-	
RXTYPE11	
-	
RXINTERVAL11	
-	
TXMAXP12	
-	
TXCSRL12	
-	
TXCSRH12	
-	
RXMAXP12	
-	
RXCSRL12	
-	
RXCSRH12	
-	
RXCOUNT12	
-	
TXTYPE12	
-	
TXINTERVAL12	
-	
RXTYPE12	
-	
RXINTERVAL12	
-	
TXMAXP13	
-	
TXCSRL13	
-	
TXCSRH13	
-	
RXMAXP13	
-	
RXCSRL13	
-	
RXCSRH13	
-	
RXCOUNT13	
-	

Table 38-115. USB Registers to Driverlib Functions (continued)

File	Driverlib Function
TXTYPE13	
-	
TXINTERVAL13	
-	
RXTYPE13	
-	
RXINTERVAL13	
-	
TXMAXP14	
-	
TXCSRL14	
-	
TXCSRH14	
-	
RXMAXP14	
-	
RXCSRL14	
-	
RXCSRH14	
-	
RXCOUNT14	
-	
TXTYPE14	
-	
TXINTERVAL14	
-	
RXTYPE14	
-	
RXINTERVAL14	
-	
TXMAXP15	
-	
TXCSRL15	
-	
TXCSRH15	
-	
RXMAXP15	
-	
RXCSRL15	
-	
RXCSRH15	
-	
RXCOUNT15	
-	
TXTYPE15	

Table 38-115. USB Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
TXINTERVAL15	
-	
RXTYPE15	
-	
RXINTERVAL15	
-	
RQPKTCOUNT1	
usb.c	voidUSBEndpointPacketCountSet
RQPKTCOUNT2	
-	
RQPKTCOUNT3	
-	
RQPKTCOUNT4	
-	
RQPKTCOUNT5	
-	
RQPKTCOUNT6	
-	
RQPKTCOUNT7	
-	
RQPKTCOUNT8	
-	
RQPKTCOUNT9	
-	
RQPKTCOUNT10	
-	
RQPKTCOUNT11	
-	
RQPKTCOUNT12	
-	
RQPKTCOUNT13	
-	
RQPKTCOUNT14	
-	
RQPKTCOUNT15	
-	
RXDPKTBUFDIS	
-	
TXDPKTBUFDIS	
-	
EPC	
usb.c	voidUSBIntDisableControl
usb.c	voidUSBIntEnableControl
usb.c	uint32_tUSBIntStatus

Table 38-115. USB Registers to Driverlib Functions (continued)

File	Driverlib Function
usb.c	uint32_tUSBIntStatusControl
usb.c	voidUSBHostPwrConfig
usb.c	voidUSBHostPwrFaultEnable
usb.c	voidUSBHostPwrFaultDisable
usb.c	voidUSBHostPwrEnable
usb.c	voidUSBHostPwrDisable
EPCRIS	
-	
EPCIM	
usb.c	voidUSBIntDisableControl
usb.c	voidUSBIntEnableControl
EPCISC	
usb.c	uint32_tUSBIntStatus
usb.c	uint32_tUSBIntStatusControl
DRRIS	
-	
DRIM	
-	
DRISC	
-	
GPCS	
usb.c	voidUSBHostMode
usb.c	voidUSBDevMode
usb.c	voidUSBOTGMode
VDC	
usb.c	voidUSBHostPwrConfig
VDCRIS	
-	
VDCIM	
-	
VDCISC	
-	
IDVRIS	
-	
IDVIM	
usb.c	voidUSBIntDisableControl
usb.c	voidUSBIntEnableControl
IDVISC	
usb.c	uint32_tUSBIntStatus
usb.c	uint32_tUSBIntStatusControl
DMASEL	
usb.c	voidUSBEndpointDMAChannel
GLBINTEN	
usb.c	voidUSBEnableGlobalInterrupt
usb.c	voidUSBDisableGlobalInterrupt

Table 38-115. USB Registers to Driverlib Functions (continued)

File	Driverlib Function
GLBINTFLG	
usb.c	uint16_tUSBGlobalInterruptFlagStatus
usb.c	voidUSBClearGlobalInterruptFlag
GLBINTFLGCL	
usb.c	voidUSBClearGlobalInterruptFlag
PP	
-	

Chapter 39

► CONNECTIVITY MANAGER (CM)

The following chapters describe the Connectivity Manager peripherals.

39.1 Technical Reference Manual Overview

The block diagram for the F2838x device is shown in Figure 39-1. This Technical Reference Manual is organized into five major sections:

- C28x SYSTEM RESOURCES

  These chapters describe the C28x CPU subsystem, C28x Boot ROM, device configuration, and other system peripherals.

- ANALOG PERIPHERALS

  These chapters describe the Analog-to-Digital Converter (ADC), Buffered Digital-to-Analog Converter (DAC), Comparator Subsystem (CMPSS), and general analog subsystem configuration.

- CONTROL PERIPHERALS

  These chapters describe the Enhanced Capture (eCAP), High Resolution Capture (HRCAP), Enhanced Pulse Width Modulator (ePWM), Enhanced Quadrature Encoder Pulse (eQEP), and Sigma Delta Filter Module (SDFM) peripherals.

- COMMUNICATION PERIPHERALS

  These chapters describe the communication peripherals available to the C28x subsystem such as the I2C, SCI, FSI, McBSP, PMBUS, and SPI. The CAN, EtherCAT, and USB peripherals are also described in this section and can be assigned to the CM subsystem.

- CONNECTIVITY MANAGER (CM)

  These chapters describe the Connectivity Manager (CM) subsystem as well as the AES, GCRC, CM-I2C, CM-UART, SSI, and Ethernet peripherals.

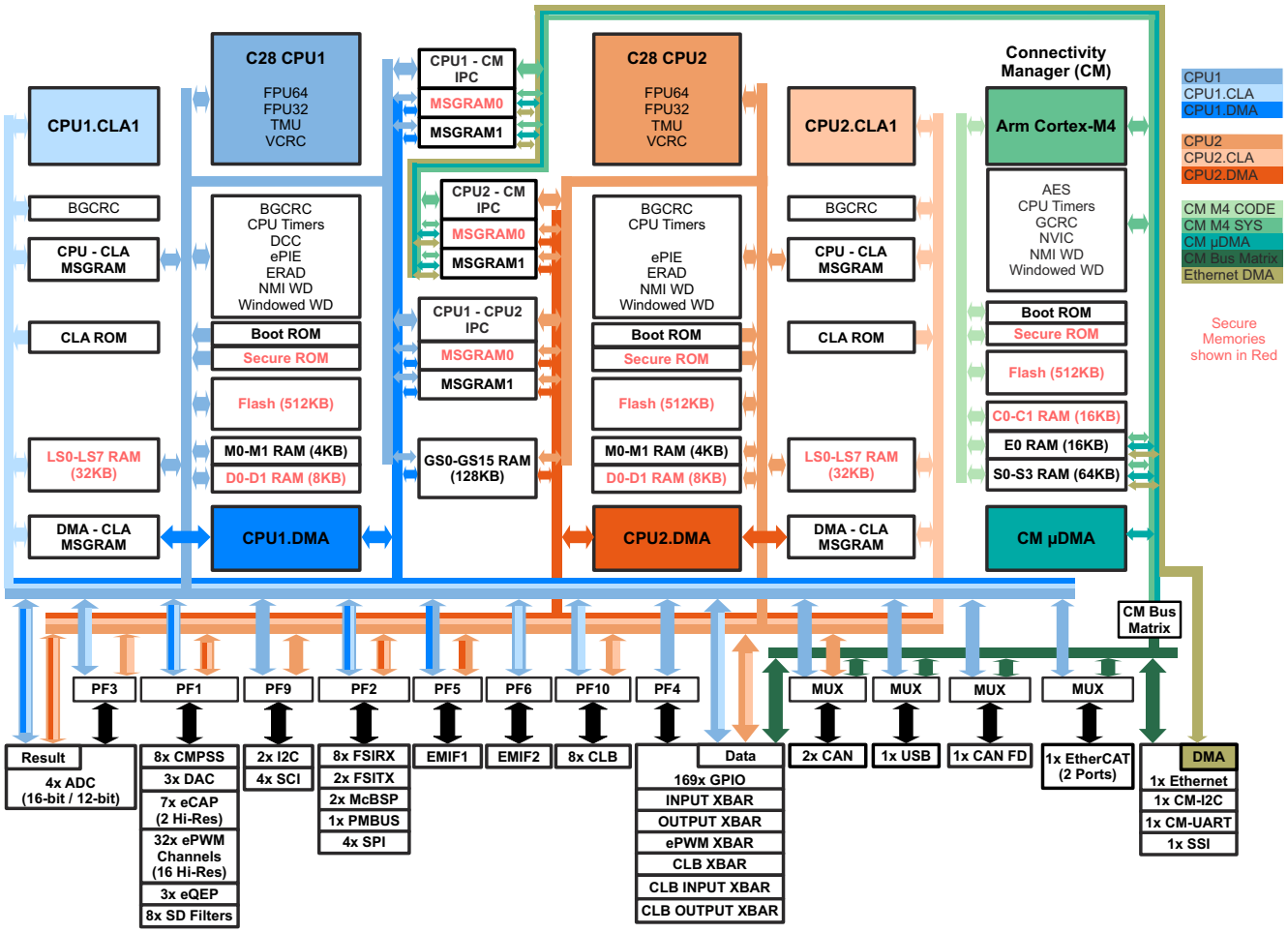


Figure 39-1. F2838x Block Diagram

Chapter 40

Connectivity Manager Subsystem

This chapter discusses the Connectivity Manager (CM) architecture and its associated peripherals.

40.1 Connectivity Manager Overview

The TMS320F2838x MCU supports dual-core C28x architecture along with a new connectivity manager (CM) subsystem. The connectivity manager subsystem is based on the industry standard 32-bit Arm® Cortex®-M4 CPU and features a wide variety of communication peripherals, including EtherCAT™, Ethernet, USB, MCAN (CAN-FD), DCAN, UART, SSI, I2C, and so on.

Targeting performance and flexibility, the connectivity manager is based on 125-MHz Cortex®-M4 architecture and provides a variety of integrated memories as well as multiple programmable GPIOs. It offers consumers compelling cost-effective solutions by integrating application-specific peripherals, and provides a comprehensive library of software tools that minimize board costs and design-cycle time. Table 40-1 lists the key architectural features of connectivity manager.

The primary goals of the Connectivity Manager (CM) are to:

- Allow easy porting of standard communication software stacks from the Arm® eco system.
- Allow easy porting of AUTomotive Open System ARchitecture (AUTOSAR) and Micro Controller Abstraction Layer (MCALs) for communication peripherals like DCAN.
- Provide additional communication MIPS.

Table 40-1. Connectivity Manager Architectural Features

Feature	Description	CPU1	CPU2
Core	Arm® Cortex®-M4	NA	NA
Frequency (MHz)	125 MHz	NA	NA
Flash	512KB	No	No
RAM	96KB	Yes	Yes
BOOTROM	96KB	No	No
µDMA	1	No	No
DCAN	2	Yes	Yes
EtherCAT	1	Yes	No
USB	1	Yes	No
MCAN (CAN-FD)	1	No	No
Ethernet	1	No	No
SSI	1	No	No
UART	1	No	No
I2C	1	No	No
AES	1	No	No
GCRC	1	No	No
Timer	3	No	No
Windowed watchdog	1	No	No

40.2 Connectivity Manager Functional Block Diagram

Figure 40-1 shows the functional block diagram and associated peripherals of the connectivity manager.

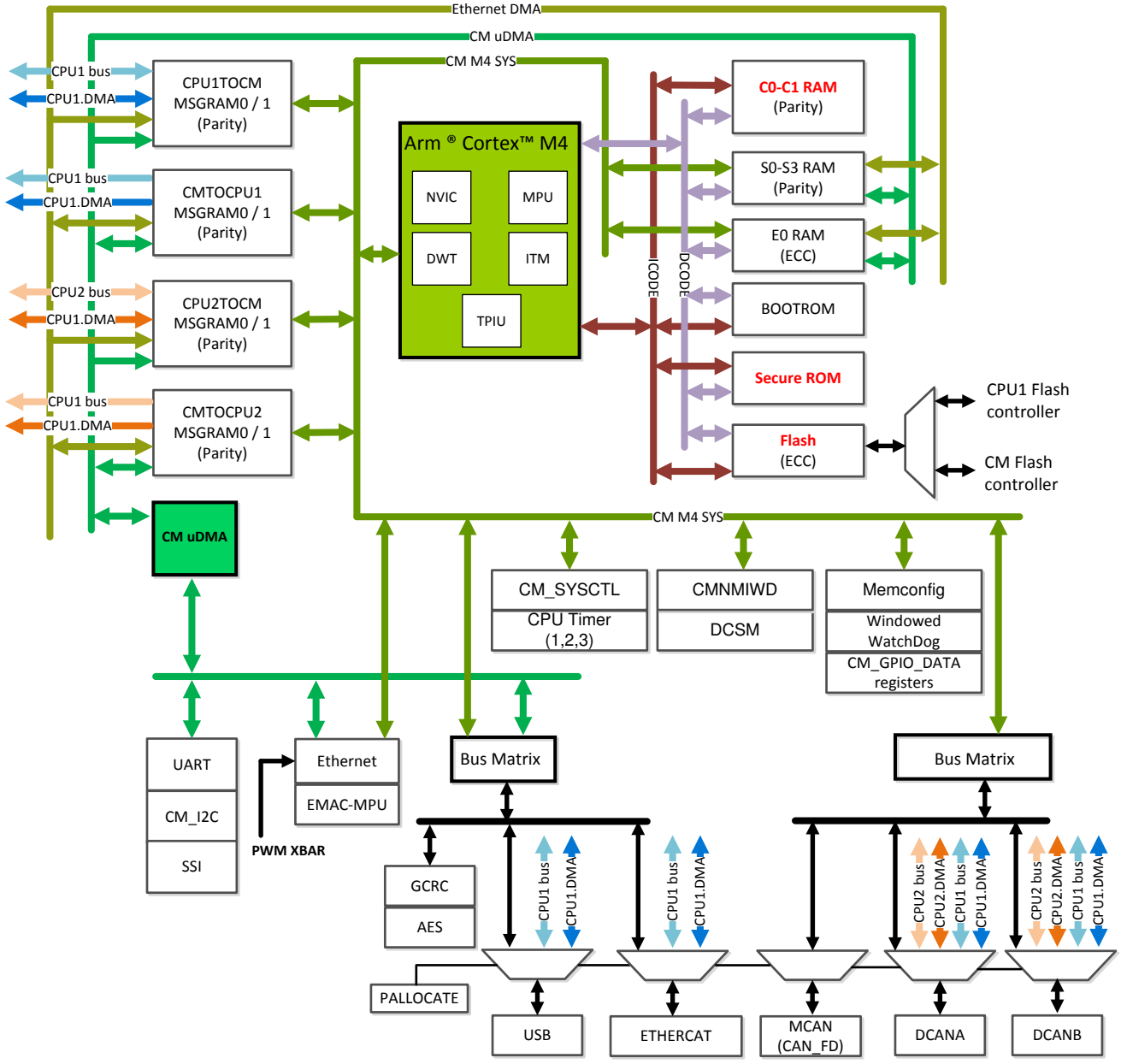


Figure 40-1. Connectivity Manager Block Diagram

40.3 Arm® Cortex®-M4 Processor Core Overview

The Arm® Cortex®-M4 processor includes the following:

- 32-bit Arm® Cortex®-M4 architecture optimized for small-footprint embedded applications
- Cortex®-M4 CPU can be operated at maximum frequency of 125-MHz
- Outstanding processing performance combined with fast interrupt handling
- Thumb-2 mixed, 16-/32-bit instruction set delivers the high performance expected of a 32-bit Arm® core in a compact memory size usually associated with 8- and 16-bit devices, typically in the range of a few kilobytes of memory for microcontroller-class applications
  - Single-cycle multiply instruction and hardware divide
  - Atomic bit manipulation (bit-banding), delivering maximum memory utilization and streamlined peripheral control
  - Unaligned data access, enabling data to be efficiently packed into memory
- Fast code execution permits slower processor clock or increases sleep mode time
- Harvard architecture characterized by separate buses for instruction and data
- Efficient processor core, system and memories
- Hardware division and fast multiplier
- Deterministic, high-performance interrupt handling for time-critical applications
- Memory protection unit (MPU) to provide a privileged mode for protected operating system functionality
- Enhanced system debug with extensive breakpoint and trace capabilities
- Migration from the Arm7® processor family for better performance and power efficiency

See the *Arm® Cortex®-M4 Processor Technical Reference Manual* for more information.

Chapter 41

Connectivity Manager System Control and Interrupts

This section describes the system-level functionality of the F2838x Connectivity Manager CPU.

41.1 Introduction

This section provides discussions that include configuration of the clocking, resets, and interrupts of the CPU and peripherals, as well as the operation of the on-chip memories, timers, and security features.

41.2 Reset

The types and effects of the different reset sources on the CM subsystem are discussed here. Reset for the CM subsystem is controlled by CPU1 and it is held in reset unless the user's application code on CPU1 releases the CM out of reset by writing '0' to the RESET bit of the CMRESCTL register. After updating the RESET bit, the user's software can read the RESETSTS bit of the CMRESCTL register to check the reset status of the CM. A Write to the CMRESCTL register is protected by KEY; therefore the user must put the correct value (0xA5A5) in the KEY field to make any updates to this register. Because of this, the user must always perform a 32-bit write to this register.

Following are the different reset sources on the CM subsystem.

41.2.1 CPU1 $\overline{\text{SYSRS}}$

Any time the CPU1 subsystem gets reset and asserts a CPU1. $\overline{\text{SYSRS}}$ signal, it resets the CM subsystem as well. The CM subsystem is held in reset unless the user application code on CPU1 releases CM out of reset by writing '0' to the RESET bit of the CMRESCTL register.

41.2.2 System Reset Request (CMSYSRESETREQ)

User software running on the CM can initiate a reset to the CM subsystem by writing '1' to the SYSRESETREQ bit of the AIRCR register of the Cortex®-M4. This action resets almost all the logic on the CM except for debug.

After this reset, the CMSYSRESETREQ bit in the CMRESC register is set. Software can read this bit to determine the cause of the reset and clear the status by writing '1' into the corresponding bit in the CMRESCCLR register.

Note

The Cortex®-M4 also has a software reset by the name VECTRESET. VECTRESET only resets the Cortex®-M4 core and not other logic in the CM subsystem; hence, this could cause unexpected behavior. You should avoid using VECTRESET.

41.2.3 CM NMI Watchdog Reset (CMNMIWDRSTn)

Like CPU1/CPU2, the CM has its own non-maskable interrupt (NMI) module that captures the hardware errors in the CM subsystem as well as some of device level error. The NMI module has a watchdog timer that triggers a reset if the CPU does not respond to an error within a user-specified amount of time. This reset activity resets almost all the logic on the CM except for debug. The CMNMIWDRSTn bit can trigger an NMI or interrupt on CPU1 (depending on the configuration of the CMNMIWDRST bit field in the CMTOCPU1NMICTL register).

After this reset, the CMNMIWDRSTn bit in the CMRESC register will be set. Software can read this bit to determine the cause of the reset and clear the status by writing '1' into the corresponding bit in the CMRESCCLR register.

41.2.4 CM Secure Code Copy Reset (CMSCCRESETn)

The dual-zone code security module (DCSM) on this device locks read access to secure memories including ROM. To facilitate CRC checks, TI provides ROM functions to securely access those memory areas. To prevent security breaches, interrupts must be disabled before calling these functions. If a vector fetch occurs in a secure copy or CRC function, the DCSM triggers a reset. This reset on the CM is the same as the CM System Reset.

After this reset, the CMEOLRESETn bit in the CMRESC register will be set. Software can read this bit to determine the cause of the reset and clear the status by writing '1' into corresponding bit in the CMRESCCLR register.

41.3 CM Clocking

This section explains the clock sources and clock domains on the Connectivity Manager, and how to configure them for application use. The figure below provides an overview of the CM clocking system.

41.3.1 CM Clock Sources

Clock sources for C28x and the CM are the same. Refer to Section 3.7.1 for more information.

41.3.2 CM Derived Clocks

Derived clock sources for the Connectivity Manager are the same as C28x derived clocks. Refer to Section 3.7.2 for more information.

41.3.3 CM Device Clock Domains

The device clock domains are applied to the clock inputs of the various modules in the device. They are connected to the derived clocks, either directly or through an additional divider. In addition to the clock domains defined in Section 3.7.3, the following clock domains are derived specifically for the CM (Cortex®-M4) Subsystem.

41.3.3.1 Connectivity Manager Clock (CMCLK)

The CM clock is derived from PLLSYSCLK or AUXPLLRAWCLK and then divided down by the CMCLK divider. This clock is asynchronous to the CPU1/CPU2 system clock. This clock is used by the Cortex®-M4 CPU (CM), GPIO, DCSM, Message RAMs, IPC and Watch Dog.

41.3.3.2 CM Peripheral Subsystem Clock (CM.PERx.SYSCLK)

This clock is the same as CMCLK, where each peripheral clock has its own independent clock gating that is controlled by the CMPCLKCRx registers.

41.3.3.3 MCAN Bit Clock

The required frequency tolerance for the MCAN bit clock depends on the bit timing setup and network configuration, and can be as tight as 0.1. Since the main system clock (in the form of CM.PERx.SYSCLK) may not be precise enough, the bit clock can also be connected to AUXCLKIN or AUXPLLRAWCLK by way of the CLKSRCCTL2.MCANxBITCLKSEL bit.

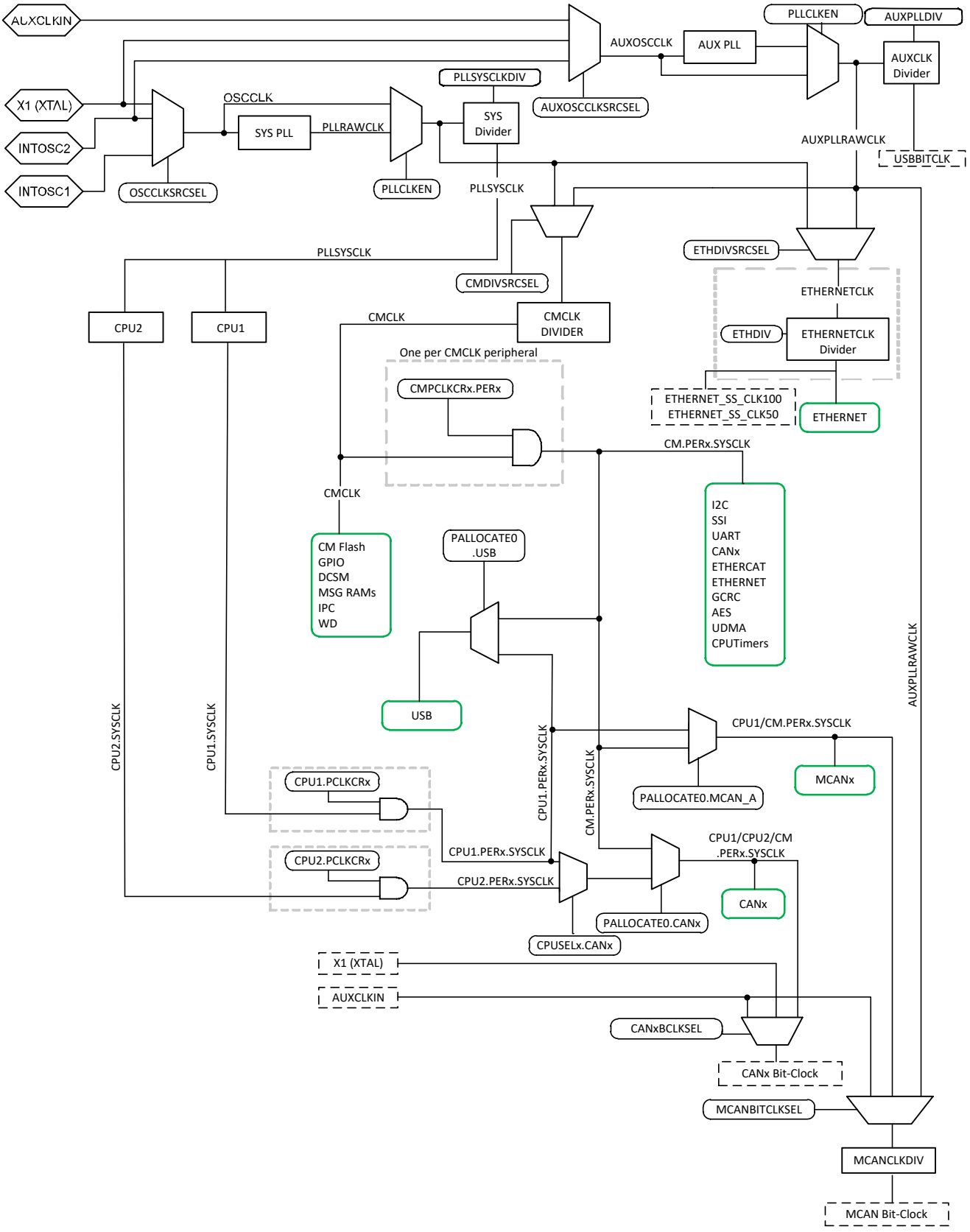


Figure 41-1. CM Clocking System

41.3.4 CM Clock Connectivity

Table 41-1 provides details on the CM clock connections.

Table 41-1. CM Clock Connections

Clock Domain	Module Name
CMCLK	GPIO DCSM Message RAMs IPC Watch Dog
CM.PERx.SYSCLK	I2C SSI UART MCANA ETHERCAT ETHERNET GCRC AESLIP µDMA CPUTIMER 0 - 2 USB CANA - B
CAN Bit Clock	CANA - B
MCAN Bit Clock	MCANA

41.4 SysTick

The Cortex®-M4 includes an integrated system timer, SysTick, which provides a simple, 24-bit clear-on-write, decrementing, wrap-on-zero counter with a flexible control mechanism. The counter can be used in several different ways, for example as:

- An RTOS tick timer that fires at a programmable rate (for example, 100 Hz) and invokes a SysTick routine.
- A high-speed alarm timer using the system clock.
- A variable rate alarm or signal timer; the duration is range-dependent on the reference clock used and the dynamic range of the counter.
- A simple counter used to measure time to completion and time used.
- An internal clock source control based on missing/meeting durations. The COUNT bit in the STCTRL control and status register can be used to determine if an action completed within a set duration, as part of a dynamic clock management control loop.

The timer consists of three registers:

- **SysTick Control and Status (STCTRL)**: A control and status counter to configure its clock, enable the counter, enable the SysTick interrupt, and determine counter status.
- **SysTick Reload Value (STRELOAD)**: The reload value for the counter, used to provide the counter's wrap value.
- **SysTick Current Value (STCURRENT)**: The current value of the counter.

When enabled, the timer counts down on each clock from the reload value to zero, reloads (wraps) to the value in the STRELOAD register on the next clock edge, then decrements on subsequent clocks. Clearing the STRELOAD register disables the counter on the next wrap. When the counter reaches zero, the COUNT status bit is set. The COUNT bit clears on reads.

Writing to the STCURRENT register clears the register and the COUNT status bit. The write does not trigger the SysTick exception logic. On a read, the current value is the value of the register at the time the register is accessed.

The SysTick counter runs on the processor clock. If this clock signal is stopped for low power mode, the SysTick counter stops. Ensure software uses aligned word accesses to access the SysTick registers.

Note: When the processor is halted for debugging, the counter does not decrement.

41.5 Watchdog Timer

The Connectivity Manager (CM) has one watchdog (also referred to as windowed watchdog) timer. The functionality of this watchdog timer is the same as the one used on CPUx subsystems. Refer to the Watchdog Timers section in the *C28x System Control* chapter for details about this module. Following are some differences in the configuration of the watchdog timer on the CM vs CPUx.

- The Watchdog timer on CM is disabled by default. Software needs to clear the WDDIS bit in the WDCR register to enable the watchdog.
- Whenever the watchdog counter (WDCR) overflows or an incorrect value is written to WDCR[WDCHK], an NMI gets generated (not reset or interrupt such as CPUx watchdog timers) to the CMNMIWD module. If software is not able to service the NMI, then the NMIWD module will trigger a reset to the CM.

The CM watchdog timer counter stops incrementing when the Cortex®-M4 is halted during the debug session.

41.6 Exceptions and NMI

This section provides details about interrupts and exception handling supported by the CM subsystem. Both the CM and C28 subsystems each have an independent NMI module which captures various exceptions that can occur in the system and trigger an NMI to the respective CPU core.

This device has exception capturing and handling ability that enables the device to be used in many safety-critical applications. Device run-time exceptions that can be detected and acted upon include clock failure detection, memory access error detection, memory uncorrectable error, Flash uncorrectable error, MCAN uncorrectable error, NMI watchdog error, EtherCAT error, bus fault detection, and interrupt handler address mismatch errors. This device also supports back-to-back, non-maskable interrupt handling capability, along with highly configurable peripheral interrupt handling.

The CM subsystem is built around the Arm® Cortex®-M4 core and includes the Nested Vectored Interrupt Controller (NVIC) module, while the C28 subsystem is built around the C28x core and includes the peripheral interrupt expansion (PIE) module. This enables the user to configure, handle, and serve interrupt requests from different subsystem peripherals and handle various exceptions that can occur in the device during its operation. Exception handling on the CM subsystem is built such that it will be able to identify and handle the errors, even if the C28 subsystem fails to handle its exceptions.

41.6.1 CM Subsystem Nested Vectored Interrupt Controller

Refer to Section 41.7 for a detailed overview of the NVIC.

As part of the CM subsystem, the Cortex®-M4 NVIC handles exceptions that can occur on the CM subsystem. The Cortex®-M4 NVIC module supports a non-maskable interrupt, which has higher priority than all other NVIC-supported interrupts or exceptions. The CM subsystem's non-maskable interrupt module (CMNMI) is responsible for generating this non-maskable interrupt to the Cortex®-M4 CPU core in the CM subsystem. Refer to Section 41.6.3 for more details on NMI handling.

The NVIC supports a HARDFAULT exception interrupt which has higher priority than any programmable interrupts but less priority than an NMI. Other programmable exceptions supported by the NVIC are memory management faults, bus faults, and usage fault programmable exceptions. These programmable exceptions are disabled by default and the system errors which can cause these exception events end up triggering a HARDFAULT exception. Section 41.6.2 provides details on the events that cause these exceptions.

On power-up and any reset that resets the CM CPU, the NVIC is mapped to the address of 0x0000 0000 in ROM. The M-Boot ROM installs predefined interrupt handlers in the default NVIC table as needed for the boot

ROM execution and C28. Once the user application is started by boot ROM, the interrupt handlers should have their own NVIC vector table and map the NVIC base address to user locations. If users fail to remap the NVIC to their application needs, any interrupt that occurs while the application is executing ends up calling interrupt and exception handlers installed by boot ROM. Refer to the *Boot ROM* chapter for more details on boot ROM handlers in the NVIC.

41.6.2 CM Subsystem Exceptions Handling

Table 41-2 provides information on different exceptions supported by NVIC on the CM subsystem.

Table 41-2. CM Subsystem Exceptions

Exception Type	Vector Number	Priority	Description
Reset	1	-3 (highest	This exception is invoked on power up and on any other reset. On the first instruction, Reset drops to the lowest priority and then is called the base level of activation. This exception is asynchronous.
Non-Maskable Interrupt(NMI)	2	-2	A non-maskable Interrupt (NMI) can be signaled using the NMI signal or triggered by software using the Interrupt Control and State (INTCTRL) register. This exception has the highest priority other than reset. NMI is permanently enabled and has a fixed priority of -2. NMIs cannot be masked or prevented from activation by any other exception or preempted by any exception other than reset.
Hard Fault	3	-1	This exception is caused by all classes of Fault when the fault cannot activate due to priority or the configurable fault handler has been disabled. This exception is synchronous.
Memory Management	4	Configurable	This exception is caused by an MPU mismatch, including access violation and no match. This exception is synchronous.
Bus Fault	5	Configurable	A bus fault is an exception that occurs because of a memory-related fault for an instruction or data memory transaction such as a prefetch fault or a memory access fault. This fault can be enabled or disabled.
Usage Fault	6	Configurable	This exception is caused by a usage fault, such as an undefined instruction executed or an illegal state transition attempt. This exception is synchronous.

From the exceptions in Table 41-2, the NMI and bus faults are generated by the digital subsystem, whereas memory management errors are generated internally by the M4 MPU.

Bus Fault Exceptions:

For all uncorrectable memory errors during M4 CPU reads or writes (address, parity, or double data error), HRESP-based and HREADY-based error responses are generated by the memory C28 logic.

Write Accesses:

When an HRESP-error is generated for write accesses, if the intended write was a stack push, STKERR status is set by the NVIC upon seeing the bus fault indication for the stack push operation. If the application so prefers, the application can treat a STKERR as a low-priority exception and pend the same, except when stacking for an exception.

For all uncorrectable errors during reads, the same HRESP-error indication is generated. The M4 core uses this error indication in the following manner:

- For bus errors during normal data reads, the M4 can generate a bus fault, but the application can treat this as a lower priority, thereby enabling a higher priority exception to be serviced by the CPU. As an example, there can be a bus fault in a user thread and still the CPU can service an interrupt to handle critical interrupts - the bus fault can be handled afterwards. But if the read occurs during an ISR bus fault, the read is treated as a hard fault, since the read is in code for an exception that must never fault.
- For bus errors during stack pops, the NVIC sets the UNSTKERR status indication and the M4 generates a bus fault.
- For bus errors during vector table reads, the NVIC sets VECTTBL and the M4 generates a bus fault. If there is a double fault when the bus fault handler tries to read the vector, the M4 CPU enters a LOCKUP condition. In this condition, the M4 WDT timers time out and issue a reset to the system.
- For bus errors during fetches, even if the M4 sees an error response, M4 does not internally bus fault unless the CPU is decoding the erroneous instruction fetched.

For more details on how the remaining exceptions are generated by the Cortex®-M4 CPU, refer to Section 41.7.

Refer to the *ROM Code and Peripheral Booting* chapter for more details on how boot ROM handles HARDFAULT exceptions, if the exception occurs during boot ROM execution.

On the CM subsystem, the following errors generate a BUSFAULT:

- RAMUNCERR - RAM Uncorrectable error. This is a double bit error generated by the RAM wrapper logic as a bus error.
- RAMACCVIOL - Ram Access violation.

41.6.3 CM Subsystem Non-Maskable Interrupt (CMNMI) Module

The CM subsystem has the capability of detecting all serious errors that can occur in the entire system including all the subsystems, and inform the main CPU core about the error. An NMI exception to the M4 CPU on the CM subsystem is generated only when at least one or more of the below NMI error sources become active. More details on each of the sources is given in Section 41.6.3.1 and the descriptions in the CMNMIFLG register.

1. RAM/ROM uncorrectable error
2. Reset request from the EtherCAT
3. Clock failure
4. MCAN uncorrectable error
5. CM windowed watchdog timed out
6. Flash uncorrectable error

All these NMI sources are ORed to generate the NMI input to the M4 NVIC. The NMI triggers a CMNMIWD counter running at the CM subsystem frequency. The CMNMIWD counter stops counting only if all the pending NMIs are acknowledged by clearing the pending flags in the CMNMIFLG register. If the pending NMI is not acknowledged before the CMNMIWD counter reaches the value programmed in the NMI Watchdog period register (CMNMIWDPRD), an NMIWD reset is generated to the CM subsystem, which resets the entire device.

Figure 41-2 shows different sources that can trigger an NMI to the Cortex®-M4 on the CM subsystem and the registers associated with them.

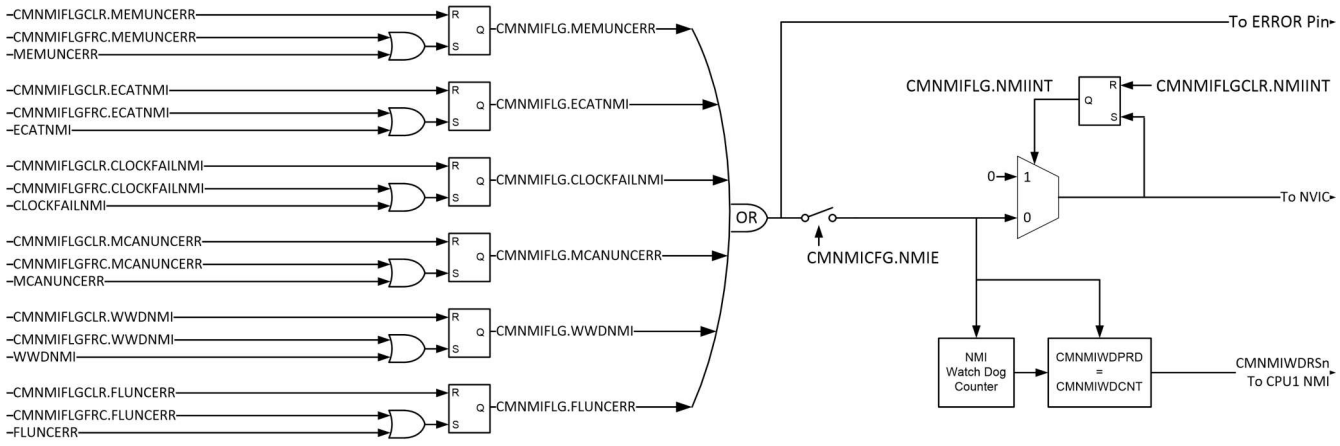


Figure 41-2. CM Subsystem NMI Sources and NMIWD

All the NMI sources shown in Figure 41-2 are enabled by default on reset. CMNMICFG.NMIE is disabled on reset and needs to be enabled by setting the bit to 1.

Whenever an NMI signal is generated, the respective bit in the CMNMIFLG register is set. To aid in debug, development, and testing, a CMNMIFLGFRC register is provided. Setting these bits in this register forces the NMI to the CPU core as shown in Figure 41-2. Refer to the CMNMIFLGFRC register for more details. When an NMI is triggered to the CM CPU, a CMNMIWD counter is triggered and begins counting and resets the device when the CMNMIWD counter reaches the programmed CMNMIWD period value. The CMNMIWD counter stops counting and resets back to zero once all the set CMNMIFLG bits and the CMNMIINT flag bit in the CMNMIFLG register are cleared.

41.6.3.1 CM Subsystem NMI Sources

This section explains all the possible NMI sources on the CM subsystem.

41.6.3.1.1 RAM/ROM Uncorrectable Error

This NMI is triggered if an error occurred on a RAM/ROM access (including peripheral RAMs) by any master.

41.6.3.1.2 Reset Request from EtherCAT

This NMI is triggered if a reset request was sent from the EtherCAT module.

41.6.3.1.3 Clock Fail Condition

A main oscillator verification circuit is provided that generates an error condition if the oscillator is running too fast or too slow or goes missing. This logic is referred to as Missing Clock Detection. When a missing clock error is generated, the CLOCKFAIL bit (bit 1) of the CMNMIFLG register is set, the clock source is switched to the 10 MHz internal oscillator, and the PLL is bypassed.

The CLOCKFAIL NMI is triggered to both the CM and C28 subsystems. Since this NMI source is enabled by default on power up, it makes it necessary for boot ROM to handle this NMI. Refer to the *Boot ROM* chapter of this document for more details on how boot ROM handles this NMI.

41.6.3.1.4 MCAN Uncorrectable Error

This NMI is triggered if an error occurred on a MCAN message RAM access.

41.6.3.1.5 CM Windowed Watchdog Timed Out

This NMI is triggered if the CM's windowed watchdog times out.

41.6.3.1.6 Flash Uncorrectable Error

This NMI is triggered if an error occurred on a CM Flash access.

41.6.3.2 CM Subsystem NMIWD Module

As explained previously, the CM subsystem is equipped with an NMI Watchdog module whose function is to make sure that a triggered non-maskable interrupt is handled by user software. This can be achieved by clearing the error conditions and clearing the respective flags in the CMNMIFLG register or by acknowledging the NMI and gracefully shutting down the system. If none of the actions mentioned are taken, then the CMNMIWD counter keeps counting until the counter value reaches the CMNMIWD period register value. An CMNMIWD reset will then be generated, which will reset the entire device.

As shown in Figure 41-2, any enabled NMI source can set the CMNMIINT respective bit in CMNMIFLG register, which will trigger an NMI to the CPU and start the CMNMIWD counter. The CMNMIWD counter will keep counting as long as the CMNMIINT bit of the CMNMIFLG register is not cleared or a reset is generated.

The CMNMIWD counter is clocked by the M4 system clock. The CMNMIWDPRD register, which is the CMNMI Watchdog Period register, can be programmed with a period limit as per user requirements which sets the clock cycle limit required for software to handle or acknowledge the NMI. A timeout condition that generates the NMI watchdog reset means that the counter value of the CMNMIWDCNT register reached the value programmed in the period register, CMNMIWDPRD.

41.6.3.2.1 Emulation Considerations

When the Cortex®-M4 CPU is suspended (in debug halt), the NMI watchdog counter will be suspended.

41.6.3.3 Handling of CMNMI

User software must clear all the flag bits which are set in the CMNMIFLG register before clearing CMNMIINT, bit 0 of the CMNMIFLG register. If the user clears the CMNMIINT bit in the CMNMIFLG register before clearing all the individual flag bits, as soon as the CMNMIINT bit is cleared it will be set back to "1" again. This will generate another back-to-back NMI to the CM subsytem's CPU, and the CMNMIWD counter will start counting again.

41.6.4 CM Interrupts/NMI to CPU1/CPU2

Table 41-3 and Table 41-4 show the various Interrupts and NMI generated to C28x CPU1 and CPU2, respectively.

Table 41-3. Interrupts and NMI From CM to CPU1

Source	NMI/Interrupt	Note
CMTOCPU1IPCINT0	Interrupt	Refer to the PIE section in C28 System Control
CMTOCPU1IPCINT1	Interrupt	Refer to the PIE section in C28 System Control
CMTOCPU1IPCINT2	Interrupt	Refer to the PIE section in C28 System Control
CMTOCPU1IPCINT3	Interrupt	Refer to the PIE section in C28 System Control
CMTOCPU1IPCINT4	Interrupt	Refer to the PIE section in C28 System Control
CMTOCPU1IPCINT5	Interrupt	Refer to the PIE section in C28 System Control
CMTOCPU1IPCINT6	Interrupt	Refer to the PIE section in C28 System Control
CMTOCPU1IPCINT7	Interrupt	Refer to the PIE section in C28 System Control
CM.SYSRESETREQ	Interrupt	A SYSRESETREQ of CM can generate an Interrupt to CPU1 based on configuration of CMTOCPU1INTCTL register.
CM.VECTRESET	Interrupt	A VECTRESET of CM can generate an Interrupt to CPU1 based on configuration of CMTOCPU1INTCTL register.
CM.NMIWDRST	NMI/Interrupt	On CMNMIWD timing out and resetting CM, an NMI/Interrupt can be generated to CPU1 based on configuration of CMTOCPU1NMICTL and CMTOCPU1INTCTL registers.

Table 41-4. Interrupts and NMI From CM to CPU2

Source	NMI/Interrupt	Note
CMTOCPU2IPCINT0	Interrupt	Refer to the PIE section in C28 System Control
CMTOCPU2IPCINT1	Interrupt	Refer to the PIE section in C28 System Control
CMTOCPU2IPCINT2	Interrupt	Refer to the PIE section in C28 System Control
CMTOCPU2IPCINT3	Interrupt	Refer to the PIE section in C28 System Control
CMTOCPU2IPCINT4	Interrupt	Refer to the PIE section in C28 System Control
CMTOCPU2IPCINT5	Interrupt	Refer to the PIE section in C28 System Control
CMTOCPU2IPCINT6	Interrupt	Refer to the PIE section in C28 System Control
CMTOCPU2IPCINT7	Interrupt	Refer to the PIE section in C28 System Control

41.7 Nested Vectored Interrupt Controller (NVIC)

The NVIC multiplexes interrupts from various peripherals into the CM interrupt lines. In essence, the NVIC is the Peripheral Interrupt Expansion (PIE) equivalent for the CM. The features supported by the NVIC are as follows:

- 80 interrupts
- A programmable priority level of 0-7 for each interrupt. A higher level corresponds to a lower priority, so level 0 is the highest interrupt priority.
- Low-latency exception and interrupt handling
- Level and pulse detection of interrupt signals
- Dynamic reprioritization of interrupts
- Grouping of priority values into group priority and subpriority fields
- Interrupt tail-chaining
- An external non-maskable interrupt

The CM automatically stacks its state on exception entry and unstacks this state on exception exit, with no instruction overhead, providing low latency exception handling.

Table 41-5. NVIC Interrupt Mapping

Exception	Exception Number (Vector Table Offset)	Priority (Higher Number, Lower Priority)	Allocation	Interrupt Type
-(Stack Top)	0	-	NA	NA
Reset	1	-3	NA	NA
NMI	2	-2	NA	NA
HardFault	3	-1	NA	NA
MemManage	4	Configurable	NA	NA
BusFault	5	Configurable	NA	NA
UsageFault	6	Configurable	NA	NA
RESERVED	7	Configurable	NA	NA
RESERVED	8	Configurable	NA	NA
RESERVED	9	Configurable	NA	NA
RESERVED	10	Configurable	NA	NA
SVCall	11	Configurable	NA	NA
Debug Monitor	12	Configurable	NA	NA
RESERVED	13	Configurable	NA	NA
PendSV	14	Configurable	NA	NA
SysTick	15	Configurable	NA	NA
IRQ0	16	Configurable	MCANSS_INT[0]	Active High Level Interrupt
IRQ1	17	Configurable	MCANSS_INT[1]	Active High Level Interrupt
IRQ2	18	Configurable	MCANSS_WAKE_AND_TS_PLS_INT	Active High Pulse Interrupt
IRQ3	19	Configurable	MCANSS_ECC_CORR_PLS_INT	Active High Pulse Interrupt
IRQ4	20	Configurable	RESERVED	
IRQ5	21	Configurable	ECATINT	Active High Level Interrupt
IRQ6	22	Configurable	ECATSYNC0INT	Active High Pulse Interrupt
IRQ7	23	Configurable	ECATSYNC1INT	Active High Pulse Interrupt
IRQ8	24	Configurable	ECATRSTINT	Active High Pulse Interrupt
IRQ9	25	Configurable	DCAN0INT0	Active High Level Interrupt
IRQ10	26	Configurable	DCAN0INT1	Active High Level Interrupt
IRQ11	27	Configurable	DCAN1INT0	Active High Level Interrupt
IRQ12	28	Configurable	DCAN1INT1	Active High Level Interrupt
IRQ13	29	Configurable	EMAC_INT	Active High Level Interrupt
IRQ14	30	Configurable	EMAC_TX_INT[0]	Active High Level Interrupt
IRQ15	31	Configurable	EMAC_TX_INT[1]	Active High Level Interrupt
IRQ16	32	Configurable	EMAC_RX_INT[0]	Active High Level Interrupt
IRQ17	33	Configurable	EMAC_RX_INT[1]	Active High Level Interrupt
IRQ18	34	Configurable	UART0INT	Active High Level Interrupt
IRQ19	35	Configurable	RESERVED	
IRQ20	36	Configurable	SSI0INT	Active High Level Interrupt
IRQ21	37	Configurable	RESERVED	
IRQ22	38	Configurable	I2C0INT	Active High Level Interrupt
IRQ23	39	Configurable	RESERVED	
IRQ24	40	Configurable	USBINT	Active High Level Interrupt
IRQ25	41	Configurable	UDMASWINT	Active High Level Interrupt
IRQ26	42	Configurable	UDMAERRINT	Active High Pulse Interrupt

Table 41-5. NVIC Interrupt Mapping (continued)

Exception	Exception Number (Vector Table Offset)	Priority (Higher Number, Lower Priority)	Allocation	Interrupt Type
IRQ27	43	Configurable	RESERVED	
IRQ28	44	Configurable	RESERVED	
IRQ29	45	Configurable	CPU1TOCMIPCINT0	Active High Pulse Interrupt
IRQ30	46	Configurable	CPU1TOCMIPCINT1	Active High Pulse Interrupt
IRQ31	47	Configurable	CPU1TOCMIPCINT2	Active High Pulse Interrupt
IRQ32	48	Configurable	CPU1TOCMIPCINT3	Active High Pulse Interrupt
IRQ33	49	Configurable	CPU1TOCMIPCINT4	Active High Pulse Interrupt
IRQ34	50	Configurable	CPU1TOCMIPCINT5	Active High Pulse Interrupt
IRQ35	51	Configurable	CPU1TOCMIPCINT6	Active High Pulse Interrupt
IRQ36	52	Configurable	CPU1TOCMIPCINT7	Active High Pulse Interrupt
IRQ37	53	Configurable	CPU2TOCMIPCINT0	Active High Pulse Interrupt
IRQ38	54	Configurable	CPU2TOCMIPCINT1	Active High Pulse Interrupt
IRQ39	55	Configurable	CPU2TOCMIPCINT2	Active High Pulse Interrupt
IRQ40	56	Configurable	CPU2TOCMIPCINT3	Active High Pulse Interrupt
IRQ41	57	Configurable	CPU2TOCMIPCINT4	Active High Pulse Interrupt
IRQ42	58	Configurable	CPU2TOCMIPCINT5	Active High Pulse Interrupt
IRQ43	59	Configurable	CPU2TOCMIPCINT6	Active High Pulse Interrupt
IRQ44	60	Configurable	CPU2TOCMIPCINT7	Active High Pulse Interrupt
IRQ45	61	Configurable	FMC_FSMDONE_INT	Active High Pulse Interrupt
IRQ46	62	Configurable	FMC_CORR_INT	Active High Pulse Interrupt
IRQ47	63	Configurable	AESINT	Active High Level Interrupt
IRQ48	64	Configurable	TINT1	Active High Pulse Interrupt
IRQ49	65	Configurable	TINT2	Active High Pulse Interrupt
IRQ50	66	Configurable	TINT3	Active High Pulse Interrupt
IRQ51	67	Configurable	CM_RAM_TESTERROR_LOG	Active High Pulse Interrupt
IRQ52	68	Configurable	RESERVED	
IRQ53	69	Configurable	RESERVED	
IRQ54	70	Configurable	RESERVED	
IRQ55	71	Configurable	RESERVED	
IRQ56	72	Configurable	RESERVED	
IRQ57	73	Configurable	RESERVED	
IRQ58	74	Configurable	RESERVED	
IRQ59	75	Configurable	RESERVED	
IRQ60	76	Configurable	RESERVED	
IRQ61	77	Configurable	RESERVED	
IRQ62	78	Configurable	RESERVED	
IRQ63	79	Configurable	RESERVED	

41.7.1 Level-Sensitive and Pulse Interrupts

The CM supports both level-sensitive and pulse interrupts. Pulse interrupts are also described as edge-triggered interrupts.

A level-sensitive interrupt is held asserted until the peripheral deasserts the interrupt signal. Typically this happens because the ISR accesses the peripheral, causing it to clear the interrupt request. A pulse interrupt is an interrupt signal sampled synchronously on the rising edge of the CM clock. To ensure the NVIC detects the interrupt, the peripheral must assert the interrupt signal for at least one clock cycle, during which the NVIC detects the pulse and latches the interrupt.

When the CM enters the ISR, it automatically removes the pending state from the interrupt (see Section 41.7.2 for more information). For a level-sensitive interrupt, if the signal is not deasserted before the CM returns from the ISR, the interrupt becomes pending again, and the CM must execute its ISR again. As a result, the peripheral can hold the interrupt signal asserted until it no longer needs servicing.

41.7.2 Hardware and Software Control of Interrupts

The CM latches all interrupts. A peripheral interrupt becomes pending for one of the following reasons:

- The NVIC detects that the interrupt signal is High and the interrupt is not active.
- The NVIC detects a rising edge on the interrupt signal.
- Software writes to the corresponding interrupt set-pending register bit, or to the Software Trigger Interrupt (STIR) register to make a software-generated Interrupt pending. See the NVIC_ISPRx register or STIR register.

A pending interrupt remains pending until one of the following:

- The CM enters the ISR for the interrupt, changing the state of the interrupt from pending to active. Then:
  - For a level-sensitive interrupt, when the CM returns from the ISR, the NVIC samples the interrupt signal. If the signal is asserted, the state of the interrupt changes to pending, which might cause the CM to immediately re-enter the ISR. Otherwise, the state of the interrupt changes to inactive.
  - For a pulse interrupt, the NVIC continues to monitor the interrupt signal, and if this is pulsed the state of the interrupt changes to pending and active. In this case, when the CM returns from the ISR the state of the interrupt changes to pending, which might cause the CM to immediately re-enter the ISR. If the interrupt signal is not pulsed while the CM is in the ISR, when the CM returns from the ISR, the state of the interrupt changes to inactive.
- Software writes to the corresponding interrupt clear-pending register bit
  - For a level-sensitive interrupt, if the interrupt signal is still asserted, the state of the interrupt does not change. Otherwise, the state of the interrupt changes to inactive.
  - For a pulse interrupt, the state of the interrupt changes to inactive, if the state was pending or to active, if the state was active and pending.

41.7.3 NVIC Registers Access

The NVIC registers can only be fully accessed from privileged mode, but interrupts can be pended while in unprivileged mode by enabling the Configuration and Control (CCR) register. Any other unprivileged mode access causes a bus fault. Before accessing the registers:

- Ensure software uses correctly aligned register accesses. The CM does not support unaligned accesses to NVIC registers.
- Be aware that an interrupt can enter the pending state even if it is disabled.
- Before programming the VTOR register to relocate the vector table, ensure the vector table entries of the new vector table are set up for fault handlers, NMI, and all enabled exceptions such as interrupts

41.8 32-Bit CM CPU Timers 0/1/2

The Connectivity Manager has three CPU timers, each operating on the CMCLK and generating an interrupt at certain periodic interval which is determined by the TDDR and TPRD register settings. Each timer operates at a clock period which equals:
TIMER_CLOCK_PERIOD = CMCLK_PERIOD x (TDDR+1)

Each timer generates an interrupt when it reaches 0, whose period equals:
TIMER_INTERRUPT_PERIOD = TIMER_CLOCK_PERIOD x (PRD+1)

Upon the timer reaching zero, the period value is reloaded and the sequence repeats. The timer can be stopped/started by writing a 1/0 to the TCR.TSS bit. The behavior of the timer on a CPU halt (debug halt) is determined by the "FREE, SOFT" bits of the TCR register.

Figure 41-3 and Figure 41-4 show the timer clock periods and the timer interrupt periods.

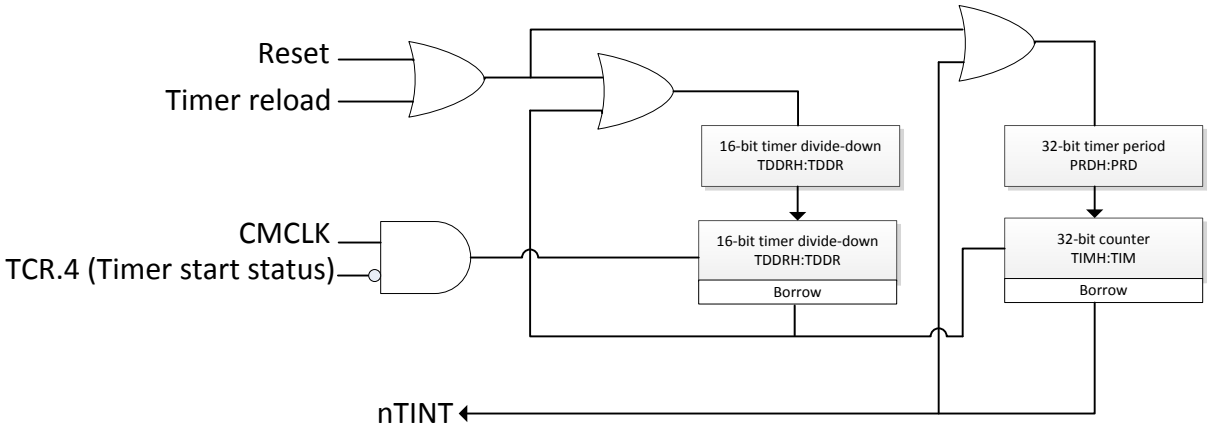


Figure 41-3. CM CPU-Timers

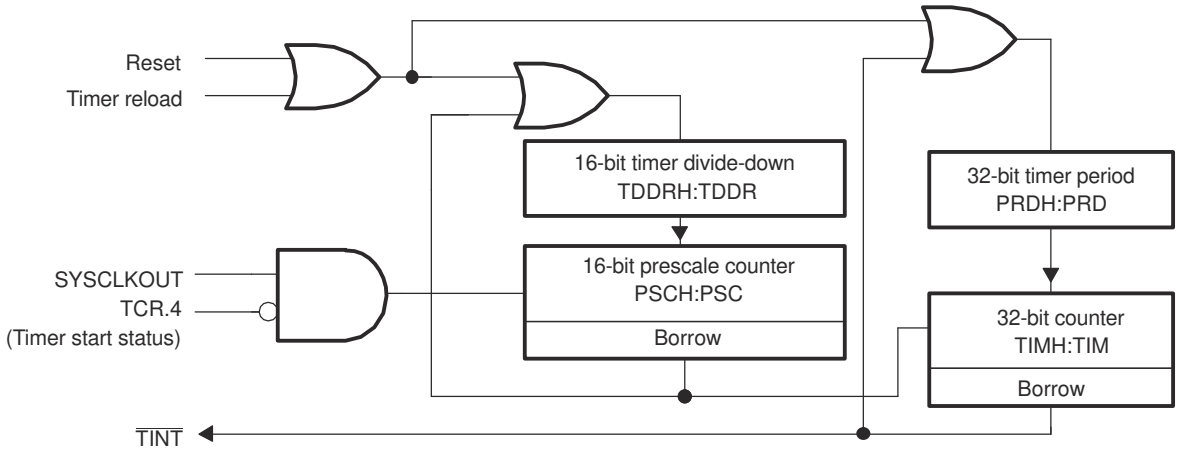


Figure 41-4. CM CPU-Timers Interrupt Signals

41.9 Memory Controller Module

On the CM subsystem, the RAMs have different characteristics. Some are:

- Dedicated to Cortex®-M4, accessible from ICODE and DCODE bus only (C0/C1 RAM)
- Shared between Cortex®-M4, µDMA and other masters in the CM subsystem (Sx RAMs, E0 RAM)
- Used to send and receive messages between the CM and CPU1/CPU2 subsystems (MSGRAM)

All these RAMs are highly configurable to achieve control for write access and fetch access from different masters. On the CM subsystemm, only E0 RAMs are enabled with the ECC feature (both data and address). All other RAMs (including MSGRAMs) are enabled with the PARITY (both data and address) feature. C0 and C1 RAMs can be configured as secure RAM. Each RAM has its own controller which takes care of the access protection/security related checks and ECC/Parity features for that RAM. Figure 41-5 shows the configuration of these RAMs.

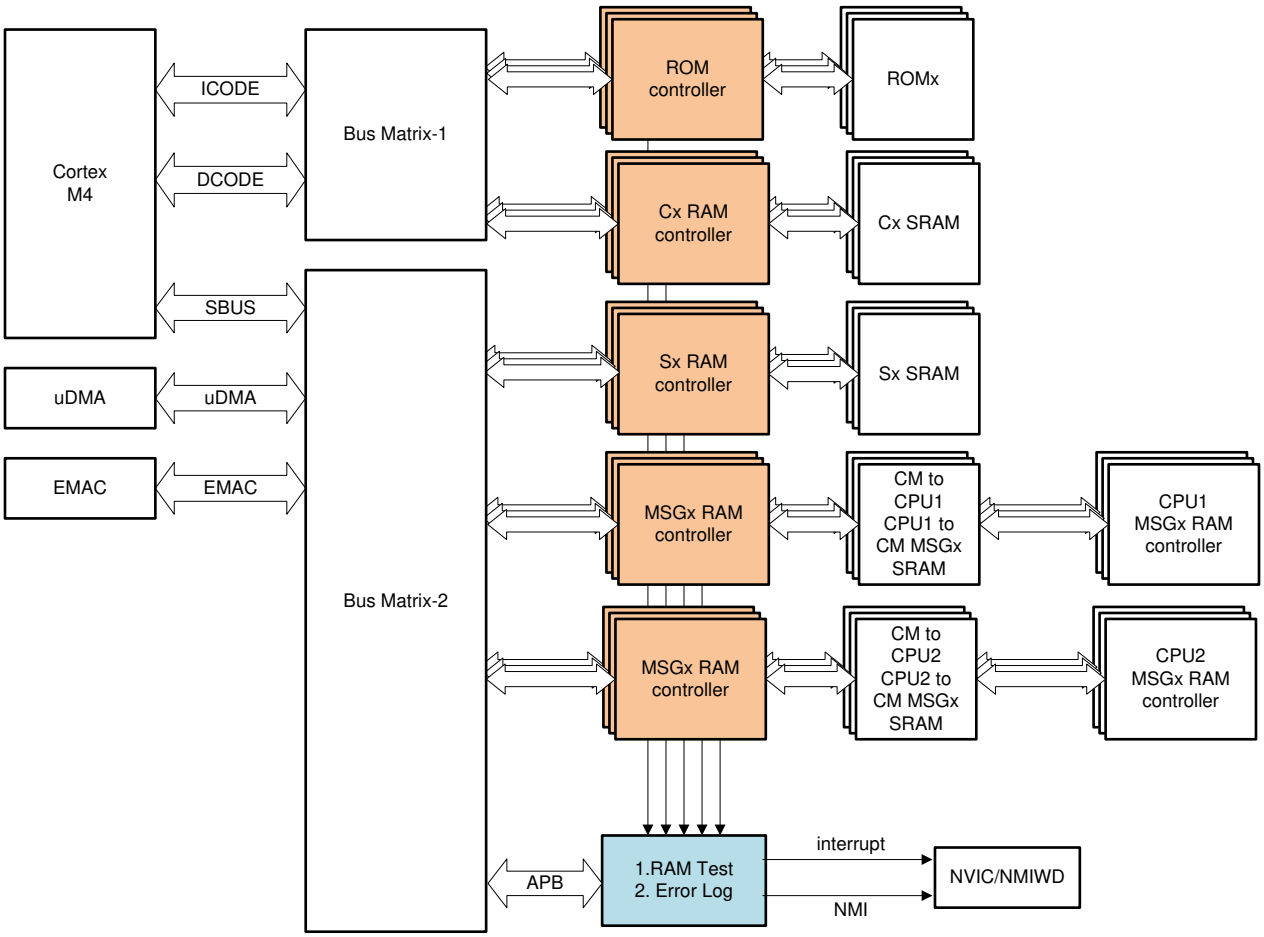


Figure 41-5. CM Memory Block Diagram

41.9.1 Functional Description

This section further defines and discusses different types of memories on the CM subsystem and other features of the SRAM controller.

41.9.1.1 Dedicated RAM

C0 and C1 are dedicated RAMs on the CM subsystem. These RAMs are connected to the ICODE and DOCDE bus of Cortex®-M4; hence, only Cortex®-M4 has access to these RAMs. These RAMs can be used for timing critical code and are also parity protected. If needed in the application, the user can configure these RAMs as secure RAM. Refer to the *Dual Code Security Module (DCSM)* chapter for more information about security.

41.9.1.2 Shared RAM

Shared RAMs are connected to the system bus and are accessible from all the masters (Cortex®-M4, µDMA and EMAC) on the CM subsystem. On this device, the CM subsystem has five blocks of shared RAMs. Four of these are parity-protected (Sx) and one block is ECC protected (E0). The user can use the E0 RAM block for safety critical code or data-like stack or interrupt handlers.

41.9.1.3 MSG RAM

MSG RAMs are connected to the system bus and are accessible from all the masters (Cortex®-M4, µDMA and EMAC) on the CM subsystem. These RAMs are also accessible from the CPU1/CPU2 subsystem (different MSG RAMs for CPU1 and CPU2) and therefore is used for message/data exchange between the CM and CPU1/CPU2 subsystems. MSG RAMs are parity protected. MSG RAMs are also referred to as IPC (inter processor communication) RAMs because these are used for communication between different subsystems. MSG RAMs do not have fetch access and cannot be used for code. On this device, the CM subsystem is asynchronous to the CPU1/CPU2 subsystem and both have access to MSG RAMs. Specific logic (mem allocate logic) is implemented to arbitrate the access from different subsystems. At any given time only one master access is connected to the MSG RAM and Mem allocate logic manages switching of MSGx RAM between C28 and CM RAM controllers.

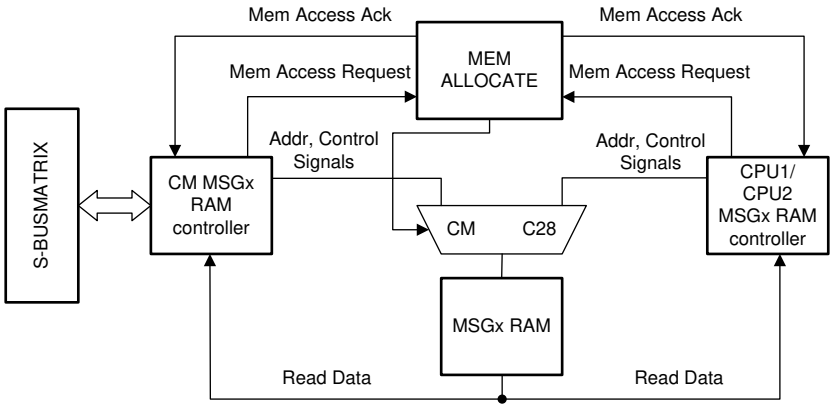


Figure 41-6. Mem allocate logic

Upon detecting valid access, the RAM controller generates a memory access request to mem allocate logic, and then waits for memory allocate logic to acknowledge. Mem Allocate logic acknowledges only after Memory is switched to the requested RAM controller. Upon detecting acknowledge, the RAM controller initiates the memory access. Mem allocate logic arbitrates accesses using round robin priority. Simultaneous access from both masters will result additional latency due to round robin prioritization, as accesses are serviced alternately and latency is introduced due to synchronization. The user application should use the IPC mechanism to avoid simultaneous accesses.

Two sets of message RAMs are defined to overcome latency issues in case of multiple threads running on CM. One is Cortex®-M4 writing/reading a message and another master such as µDMA or EMAC DMA writing or reading the message at the same time.

The message RAMs are:

- CMTOCPU1MSGRAM0 (no access from CPU2)
- CMTOCPU1MSGRAM1 (no access from CPU2)
- CPU1TOCMMSGRAM0 (no access from CPU2)

- CPU1TOCMMSGRAM1 (no access from CPU2)
- CMTOCPU2MSGRAM0 (no access from CPU1)
- CMTOCPU2MSGRAM1 (no access from CPU1)
- CPU2TOCMMSGRAM0 (no access from CPU1)
- CPU2TOCMMSGRAM1 (no access from CPU1)

Access permissions to MSG RAMs are hard-coded in functional mode.

Table 41-6 lists the allowed accesses to Message RAMs in functional mode.

Table 41-6. CM Message RAM Accesses

Message RAM	Cortex-M4	µDMA	EMAC	CPUx	CPUx.DMA
CMTOCPUxMSGRAM0	RD/WR	RD/WR	RD/WR	RD	RD
CMTOCPUxMSGRAM1	RD/WR	RD/WR	RD/WR	RD	RD
CPUxTOCMMSGRAM0	RD	RD	RD	WR	WR
CPUxTOCMMSGRAM1	RD	RD	RD	WR	WR

41.9.1.4 ROM

On this device ROMs are parity protected, and the SRAM controller is used for ROM accesses as well. Like dedicated RAMs, ROMs are also connected to the ICODE and DCODE bus and are programmed with TI code.

41.9.1.5 Interleaving

On the CM subsystem all the RAM and ROM use interleaving techniques to minimize the latencies, especially under scenarios where multiple bus masters are simultaneously trying to access data from a memory block.

These scenarios are common on CM subsystem. Examples are as follows:

- Sx memories: Cortex®-M4 loads the data into its registers for processing while EMAC dumps the received data to memory
- Cx memories: Simultaneous data and instruction fetches

Note that interleaving does not always improve the throughput compared to single bank implementation.

To implement interleaving, a single block of memory is divided into two separate equal physical blocks of half the size and only alternate 32-bit words are stored in each bank. Figure 41-7 shows how data is arranged with interleaving.

In dual-bank implementation, even and odd 32-bit word addresses are decoded separately and routed to the appropriate bank. This allows simultaneous accesses from two bus masters to these banks if accesses are not to the same bank; that is, two accesses are serviced in one cycle.

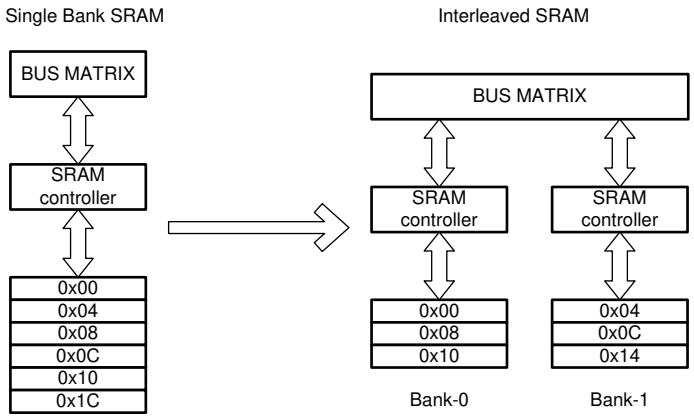


Figure 41-7. Interleaving

41.9.1.6 Access Arbitration

On the CM subsystem, accesses to all shared RAMs (except MSG RAMs) are arbitrated with fixed priority. The Cortex®-M4 system bus has high priority over any other access to ensure no wastage of MIPs. Arbitration is handled by the Arm® Bus matrix component and not part of the SRAM controller.

41.9.1.7 Access Protection

On the CM subsystem, access protection to all the RAMs are done via master specific memory protection unit (MPU) and not in the SRAM controller. Each master on the CM subsystem has a dedicated MPU to filter the accesses. For MSG RAMs, in addition to the MPU, there is logic in-built in the controller to prevent accesses such as fetch or write (if not allowed).

41.9.1.8 Memory Error Detection, Correction and Error Handling

As mentioned in earlier sections, some RAMs on CM subsystem are ECC-protected and some are parity. ROMs are parity-protected. The ECC scheme used is Single Error Correction Double Error Detection (SECDED). The parity scheme used is even parity. ECC/Parity will cover the data bits stored in memory as well as address. ECC/Parity calculation is done inside the memory controller module and then calculated. ECC/Parity is written into the memory along with the data. ECC/Parity is computed for 16-bit data; hence, for each 32-bit data, there will be three 7-bit ECC codes (or 3-bit parity), two of which are for data and a third one for the address.

Figure 41-8 and Figure 41-9 show how a word is stored in memory.

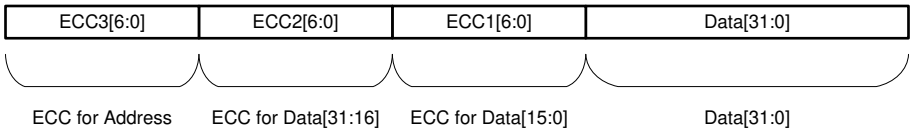


Figure 41-8. Content of Each Memory Location for ECC Memories

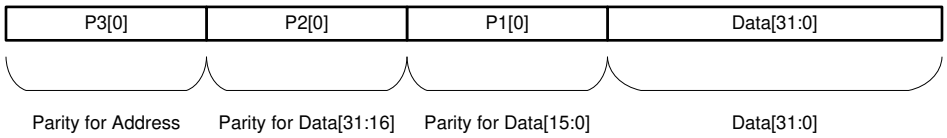


Figure 41-9. Content of Each Memory Location for Parity Memories

41.9.1.8.1 Error Detection and Correction

Error detection is done while reading the data from memory. The error detection is performed for data as well as address. For parity memory, only a single-bit error gets detected, whereas in the case of ECC memory, along with a single-bit error, a double-bit error also gets detected. These errors are called correctable errors and uncorrectable errors.

The following are characteristics of these errors:

- Parity errors are always uncorrectable errors
- Single-bit ECC errors are correctable errors
- Double-bit ECC errors are uncorrectable errors
- Address ECC errors are always uncorrectable errors

Correctable errors get corrected by the memory controller module and then correct data is given back as read data to the master. It is also written back into the memory to prevent double-bit error due to another single-bit error at the same memory address.

Size of write access initiated by bus masters on CM subsystem can be byte or half-word or full word. Since ECC/Parity is calculated for 16-it word, in case of byte write access memory controller performs Read-Modify-Write

operation (read 16-bit from RAM block, modify the specific byte with new data, calculate ECC/parity for new data and write the data and ECC/parity back into RAM block).

41.9.1.8.2 Error Handling

Two types of errors get generated by the memory controller, a correctable error and an uncorrectable error.

- On a correctable error, the following actions are performed by the memory controller:
  - Correct the data and return corrected data to the master
  - The address, for which the error occurred, gets latched into the address status register
  - Bus master info is captured
  - Correctable error counter is incremented
  - Interrupt is generated if correctable error count exceeded user programmed threshold

  The user needs to configure the correctable error threshold register based on the system requirements.
- On an uncorrectable error, the following actions are performed by the memory controller – incorrect data is returned to the Master and an NMI is generated to the Cortex®-M4. The address for which the error occurred gets latched into the address status register and a flag gets set.
  - NMI is generated to Cortex®-M4 and incorrect data is returned to the master
  - The address, for which the error occurred, gets latched into the address status register
  - Bus master info is captured

Table 41-7. Error Handling of Memories

Access Type	Error	Action
Write (16bit/32bit access)	NA	NA
Write (byte access) (Controller performs Read-Modify-Write)	Uncorrectable error on read data	Write is aborted, NMI gets generated
	Correctable error on data (only ECC RAMs)	• RMW operation is performed on corrected data. • Correctable error counter is incremented. Interrupt is generated when correctable error counter exceeds user programmed threshold value. • Corrected Data is written back to the memory.
Read	Correctable error	• Correctable error counter is incremented. Interrupt is generated when correctable error counter exceeds user programmed threshold value. • Corrected data is written back to the memory.
	Uncorrectable error on read data	NMI is generated, data (could be wrong) is returned to bus master.

Note

ECC/Parity errors are ignored for debug access (no NMI/interrupt for error during debug access). Even though an error is not generated for debug access, the memory controller always returns the corrected data for correctable errors.

41.9.1.8.3 Application Test Hooks for Error Detection and Correction

Since error detection and correction logic are part of safety critical logic, safety applications may need to ensure that the logic is always working fine (during run time also). To enable this, a test mode is provided. Using test mode, the user can inject the ECC/parity error by modifying data bits (controller does not update the ECC/parity bits) or the ECC/parity bits directly. Since the memory map for ECC/parity bits and data bits are the same, a different test mode is provided for accessing data and ECC/parity bits. The user programs different test modes based on the usage.

Table 41-8 and Table 41-9 show the bit mapping for the ECC/Parity bits when they are read in RAMTEST mode, using their respective addresses.

Table 41-8. Mapping of ECC Bits in Read Data From ECC/Parity Address Map

Data Bits Location in Read Data	Content (ECC Memory)
6:0	ECC Code for lower 16 bits of data
7	Not Used
14:8	ECC Code for upper 16 bits of data
15	Not Used
22:16	ECC Code for address
31:23	Not Used

Table 41-9. Mapping of Parity Bits in Read Data From ECC/Parity Address Map

Data Bits Location in Read Data	Content (Parity Memory)
0	Parity for lower 16 bits of data
7:1	Not Used
8	Parity for upper 16 bits of data
15:9	Not Used
16	Parity for address
31:17	Not Used

41.9.1.8.4 ROM Test

ROMs are read only memory; unlike RAMs, data or parity bits cannot be modified to introduce errors for diagnostic coverage of parity checking logic. The following method is used to check health of parity checking logic in ROMs.

- Add duplicate parity check logic and feed the same data into duplicate parity checker
- Generate uncorrectable error if the parity check status of these two separate parity checkers do not match

The probability of both circuits having fault is unlikely ; therefore, parity errors will be certainly detected.

To generate the error, a test bit FORCE_ERROR is added. When the FORCE_ERROR bit is set, the parity bit going to one of party checker is inverted, thereby introducing an uncorrectable error. An uncorrectable error is generated only if there is an error on all parity checker, that is, address, data [15:0] and data [31:16]. This will ensure that all three parity checkers are working as expected.

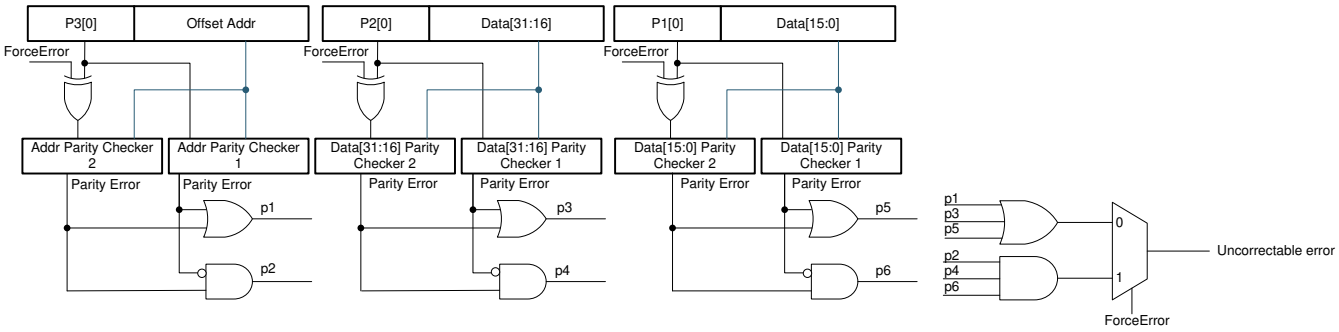


Figure 41-10. ROM Parity Checking Logic

41.9.1.9 RAM Initialization

After power-up, RAM comes up with random data and reading memories may result in an error. To ensure that read/fetch from uninitialized RAM locations do not cause ECC or parity errors, the RAM_INIT feature is provided for each memory block. Using this feature, any RAM block can be initialized with 0x0 data and respective ECC/parity bits, accordingly. This can be initiated by setting the INIT bit to '1' for the specific RAM block in INIT registers. To check the status of RAM initialization, software must poll for the INITDONE bit to be set for that RAM block in the INITDONE. Unless this bit gets set, no access should be made to that RAM memory block.

Note

None of the masters should access the memory while initialization is taking place. If memory is accessed before RAMINITDONE is set, the memory read/write as well as initialization will not happen correctly.

41.10 Memory Protection Unit (MPU)

The CM subsystem has multiple masters accessing the memory blocks and peripherals. Following is the list of masters on the CM subsystem:

- Cortex®-M4
- µDMA
- EtherNET DMA

In a multi-master system, it is important to have a protection mechanism to prevent unauthorized access to critical code, data, or peripherals from different masters or threads. This protection mechanism will:

- Prevent a process or a task from accessing memory that is not allocated to it
- Protect Cortex®-M4 code from unintended corruption by other bus masters on the CM subsystem
- Protect stack corruption by other bus masters on CM systems

The Cortex®-M4 has an Arm® native MPU (Cortex®-M4 MPU) which provides such protection (see the Memory Protection Unit chapter of the *Arm® Cortex®-M4 Processor Technical Reference Manual*). For other masters (µDMA and Ethernet DMA), a generic memory protection unit (CM-MPU) has been provided which users can configure based on the use case, to enable the protection. Basically, one MPU for each master is provided to protect the accesses from that master. See Section 41.9 for more details.

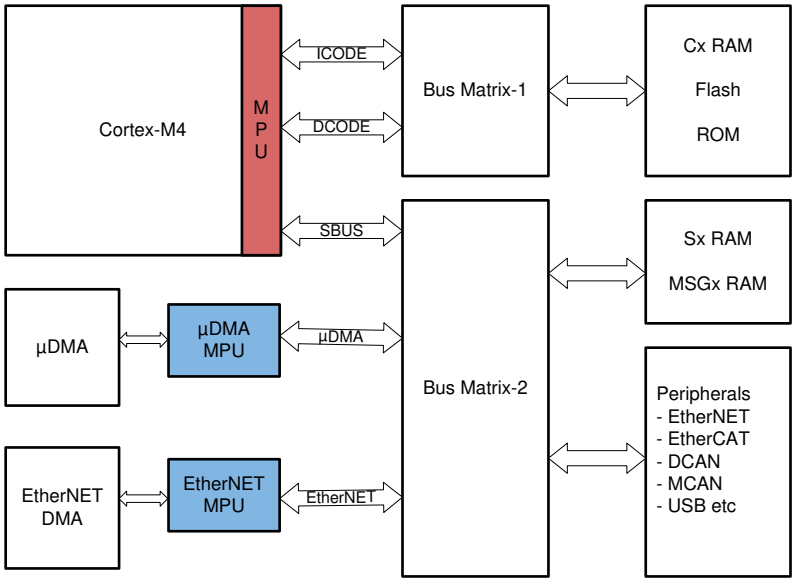


Figure 41-11. CM Block Diagram

41.10.1 Functional Description

The CM-MPU is used for access protection on µDMA and EtherNET master bus. The MPU divides the memory map into a number of regions. Each region has programmable start address, size, and access permissions.

The following are the access protections supported:

- Read-only access
- Full access - both Read and Write accesses are allowed
- No access - Read and Write accesses are not allowed

An access protection violation will result in a bus fault to the master and access info is captured in the MPU register for debug purposes.

Each MPU has a maximum of eight regions. The start address of a region must be aligned to its size. For example, a 32KB region must be aligned to a multiple of 32KB address at 0x0000_0000 or 0x0000_8000. If the start address of the region is not boundary-aligned to region size, that is, the start address is not a divisible size of the region, then hardware automatically aligns the region start address by truncating the number of LSBs, depending on region size. Figure 41-12 is an example of an unaligned start address with a region size of 2KB.

The minimum size of the region is limited to 1KB. Each region can be enabled or disabled in the application based on the use case.

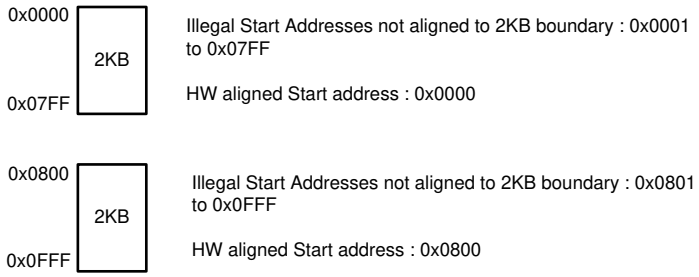


Figure 41-12. Unaligned Start Address

41.10.2 Overlapping Regions

When defining the start address and size for different regions, the user can overlap the regions; in that case, a fixed priority scheme is used to resolve the access protection. Region 7 is the highest priority and region 0 is the lowest priority. This feature is useful to define different sets of access permissions for a part of an already defined bigger region.

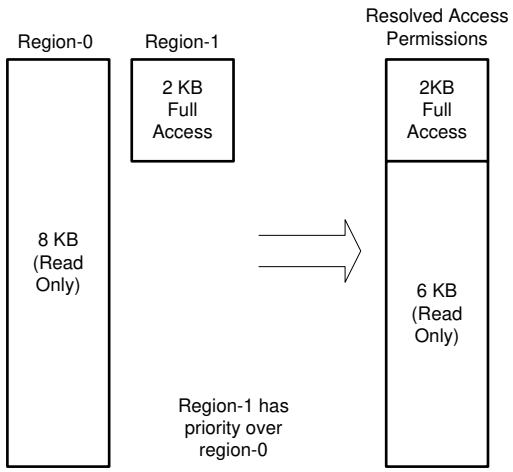


Figure 41-13. Overlapping Regions

41.10.3 Sub-Regions

Each region can be further divided into eight equal sub regions and each sub-region can be independently enabled or disabled in application code. When a sub-region is disabled, access permissions do not apply to that region. Disabled sub-regions are similar to undefined regions. This feature is useful when a part of the region is to be excluded while running a specific section of application.

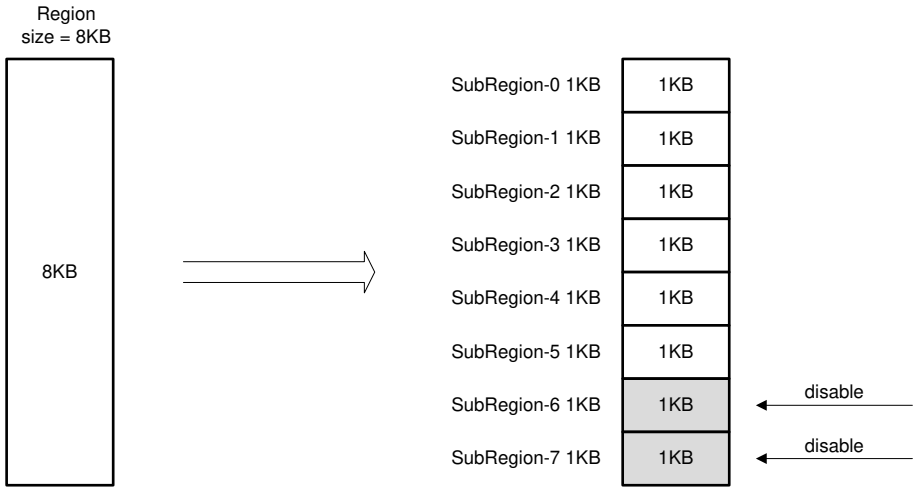


Figure 41-14. Sub-Regions

41.10.4 Programmers Model

The primary purpose of a µDMA MPU is to protect code, stack and peripherals from unintended µDMA accesses. In the following example memory map, the expectation is to not have any µDMA access to code and stack section; only read access to one section of peripheral region and full access to other sections of peripheral.

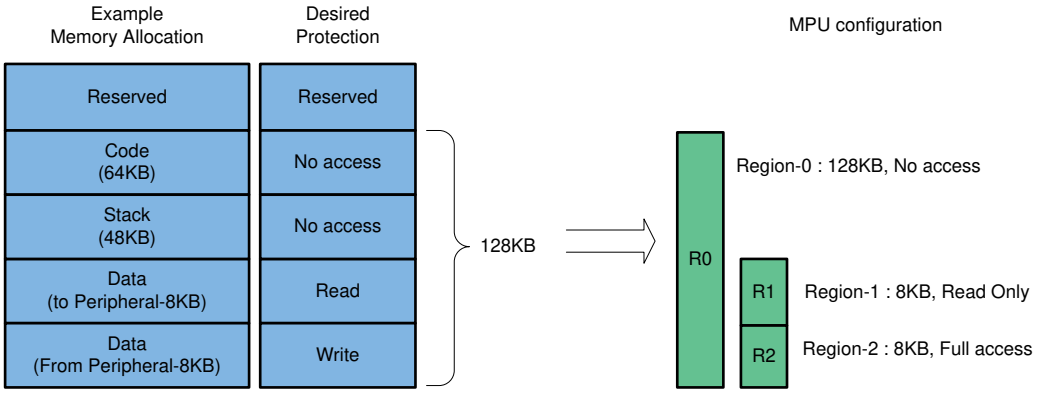


Figure 41-15. Programmers Model Memory Map

There are two ways to define the MPU region.

1. Define three different (independent) regions for no access, read access and full access memory map section.
2. Define no access for full memory map section and then define other region which overlaps to the ist region and have different protections attribute. For this Region-0 is defined as a no-access region with full memory map and then Region-1 is defined as read-only and Region-2 is defined as full-access regions. Since Region 1 and Region 2 have higher priority over Region-0, their protection attribute will override Region-0 protection attribute.

41.11 Debug and Trace

The CM subsystem includes a Cortex®-M4 that has its own debug interface via the Debug Access Port (DAP). Users can connect to the CM subsystem in parallel to the CPU1 and CPU2 core and perform a debug. Users must also release the reset for the CM subsystem before connecting to the debugger. The CCS gel file on CPU1 will take care of this if CPU1 is connected to CCS.

41.11.1 Trace Port Interface Unit

Trace capability from Cortex®-M4 will be supported on the CM subsystem. There are two trace interfaces supported on Cortex®-M4:

- Single wire trace, which follows a UART protocol and is asynchronous.
- Five-pin (four data pins and one clock pin) and parallel trace.

Both the options are supported on this device. Figure 41-16 illustrates the high-level clock and signal hookup to and from Trace Port Interface Unit.

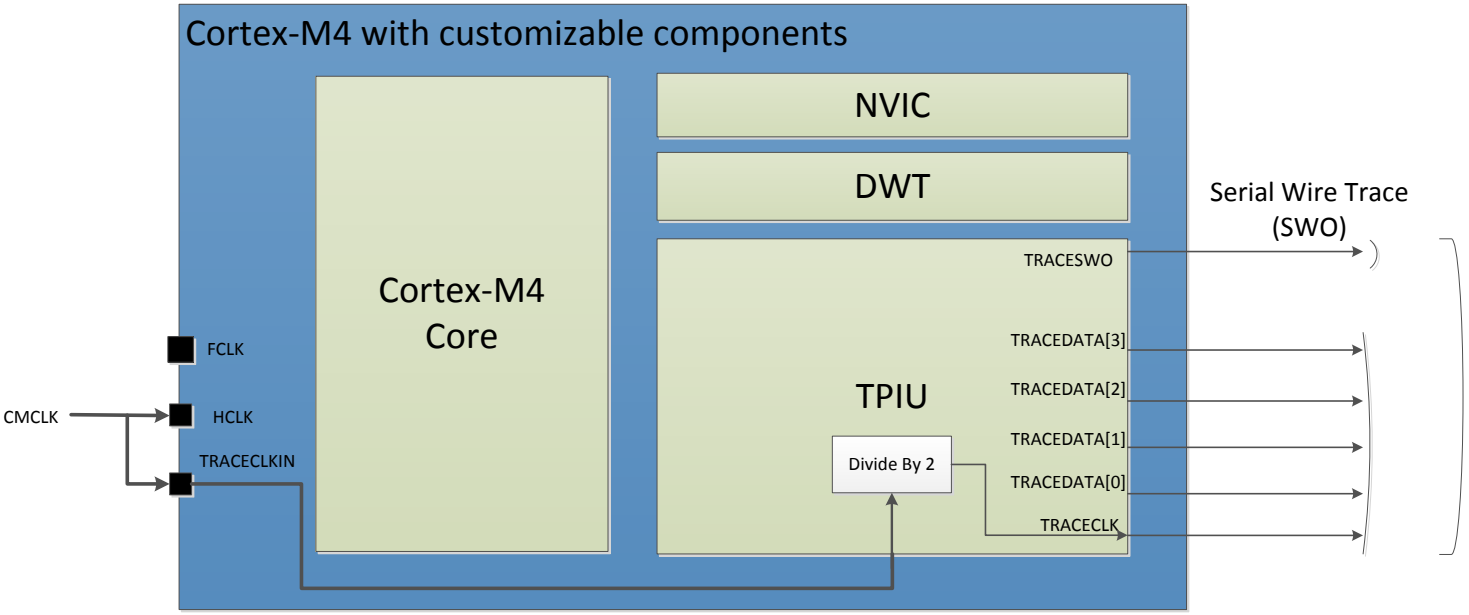


Figure 41-16. Debug Trace

Table 41-10 lists the key attributes of the two trace data export mechanisms. Refer to the *Arm® Architecture Reference Manual* for further details about TPIU and trace mechanisms.

Table 41-10. Key Attributes of Trace Data Export

Attribute Parallel Trace	Serial Wire Trace	Parallel Trace
Protocol	UART Protocol/Manchester encoded data stream	Trace Data changes on both edges of TRACECLK.
Data throughput rate	Frequency(CMHCLK)/(TPIU_ACPR + 1)	Frequency(CMHCLK)/2

You must configure the GPIO mux to select a trace function on the GPIO pin to use it.

41.12 CM-SysCtrl Registers

This section describes the Connectivity Manager System Control Registers.

41.12.1 CM System Control Base Addresses

Note

NVIC_BASE applies to NVIC, SCB, CFSR, SYSTICK, and MPU Registers.

Table 41-11. CM SYSCTRL Base Address Table (CM)

DriverLib Name	Base Address	uDMA Access	Ethernet DMA Access
CMMEMCFG_BASE	0x400F_E000	-	-
CMMEMORYDIAGERROR_BASE	0x400F_E800	-	-
CMMEMORYERROR_BASE	0x400F_E400	-	-
CMSYSCTL_BASE	0x400F_C000	-	-
CPUTIMER0_BASE	0x4008_4000	-	-
CPUTIMER1_BASE	0x4008_4010	-	-
CPUTIMER2_BASE	0x4008_4020	-	-
DMPU_BASE	0x400C_C000	-	-
EMPU_BASE	0x400C_D000	-	-
NMI_BASE	0x4008_1000	-	-
NVIC_BASE	0xE000_E000	-	-
WD_BASE	0x4008_0000	-	-

41.12.2 CM_MEMCFG_REGS Registers

Table 41-12 lists the memory-mapped registers for the CM_MEMCFG_REGS registers. All register offset addresses not listed in Table 41-12 should be considered as reserved locations and the register contents should not be modified.

Table 41-12. CM_MEMCFG_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CxLOCK	C RAM Config Lock Register		Go
4h	CxTEST	C RAM TEST Register	Lock Protection	Go
8h	CxINIT	C RAM Init Register	Lock Protection	Go
Ch	CxINITDONE	C RAM Initialization Status Register		Go
20h	CMMSGxLOCK	CM Messae RAM Config Lock Register		Go
24h	CMMSGxTEST	CM Messae RAM TEST Register	Lock Protection	Go
28h	CMMSGxINIT	CM Messae RAM Init Register	Lock Protection	Go
2Ch	CMMSGxINITDONE	CM Messae RAM Initialization Status Register		Go
40h	SxGROUP1_LOCK	Group1 S and E RAM Config Lock Register		Go
44h	SxGROUP1_TEST	Group1 S and E RAM TEST Register	Lock Protection	Go
48h	SxGROUP1_INIT	Group1 S and E RAM Init Register	Lock Protection	Go
4Ch	SxGROUP1_INITDONE	Group1 S and E RAM Initialization Status Register		Go
80h	ROM_LOCK	ROM Config Lock Register		Go
84h	ROM_TEST	ROM TEST Register	Lock Protection	Go
88h	ROM_FORCE_ERROR	ROM Force Error register	Lock Protection	Go
A0h	PERI_MEM_TEST_LOCK	Peripheral Memory Test Lock Register		Go
A4h	PERI_MEM_TEST_CONTROL	Peripheral Memory Test control Register	Lock Protection	Go

Complex bit access types are encoded to fit into small table cells. Table 41-13 shows the codes that are used for access types in this section.

Table 41-13. CM_MEMCFG_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 41-13. CM_MEMCFG_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

41.12.2.1 CxLOCK Register (Offset = 0h) [Reset = 0h]

CxLOCK is shown in Figure 41-17 and described in Table 41-14.

Return to the Summary Table.

C RAM Config Lock Register

Figure 41-17. CxLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						LOCK_C1	LOCK_C0
R-0h						R/W-0h	R/W-0h

Table 41-14. CxLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-2	RESERVED	R	0h	Reserved
1	LOCK_C1	R/W	0h	Locks write access to initialization and test control fields of C1 RAM 0: Write access allowed 1: Write access blocked Reset type: CM.RESETn
0	LOCK_C0	R/W	0h	Locks write access to initialization and test control fields of C0 RAM 0: Write access allowed 1: Write access blocked Reset type: CM.RESETn

41.12.2.2 CxTEST Register (Offset = 4h) [Reset = 0h]

CxTEST is shown in Figure 41-18 and described in Table 41-15.

Return to the Summary Table.

C RAM TEST Register

Figure 41-18. CxTEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				TEST_C1		TEST_C0	
R-0h				R/W-0h		R/W-0h	

Table 41-15. CxTEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3-2	TEST_C1	R/W	0h	Selects the different modes for C1 RAM: 00: Functional Mode. 01: Test mode to introduce Error in Data. Write updates only Data, ECC/Parity bits are not updated. 10: Test mode to introduce errors in ECC or Parity bits. ECC or Parity bits are visible on same memory map instead of Data bits. 11: Same as "00" but NMI is not generated on errors, used for diagnostics. Reset type: CM.RESETn
1-0	TEST_C0	R/W	0h	Selects the different modes for C0 RAM: 00: Functional Mode. 01: Test mode to introduce Error in Data. Write updates only Data, ECC/Parity bits are not updated. 10: Test mode to introduce errors in ECC or Parity bits. ECC or Parity bits are visible on same memory map instead of Data bits. 11: Same as "00" but NMI is not generated on errors, used for diagnostics. Reset type: CM.RESETn

41.12.2.3 CxINIT Register (Offset = 8h) [Reset = 0h]

CxINIT is shown in Figure 41-19 and described in Table 41-16.

Return to the Summary Table.

C RAM Init Register

Figure 41-19. CxINIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						INIT_C1	INIT_C0
R-0h						R-0/W1S-0h	R-0/W1S-0h

Table 41-16. CxINIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-2	RESERVED	R	0h	Reserved
1	INIT_C1	R-0/W1S	0h	RAM Initialization control for C1 RAM: 0: None. 1: Start RAM Initialization. Reset type: CM.RESETn
0	INIT_C0	R-0/W1S	0h	RAM Initialization control for C0 RAM: 0: None. 1: Start RAM Initialization. Reset type: CM.RESETn

41.12.2.4 CxINITDONE Register (Offset = Ch) [Reset = 0h]

CxINITDONE is shown in Figure 41-20 and described in Table 41-17.

Return to the Summary Table.

C RAM Initialization Status Register

Figure 41-20. CxINITDONE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						INITDONE_C1	INITDONE_C0
R-0h						R-0h	R-0h

Table 41-17. CxINITDONE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-2	RESERVED	R	0h	Reserved
1	INITDONE_C1	R	0h	RAM Initialization status for C1 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: CM.RESETn
0	INITDONE_C0	R	0h	RAM Initialization status for C0 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: CM.RESETn

41.12.2.5 CMMSGxLOCK Register (Offset = 20h) [Reset = 0h]

CMMSGxLOCK is shown in Figure 41-21 and described in Table 41-18.

Return to the Summary Table.

CM Messae RAM Config Lock Register

Figure 41-21. CMMSGxLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				LOCK_CMTOCPU2MSGRAM1	LOCK_CMTOCPU2MSGRAM0	LOCK_CMTOCPU1MSGRAM1	LOCK_CMTOCPU1MSGRAM0
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 41-18. CMMSGxLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	LOCK_CMTOCPU2MSGRAM1	R/W	0h	Locks write access to initialization and test control fields of Message RAM CMTOCPU2MSGRAM1 0: Write access allowed 1: Write access blocked Reset type: CM.RESETn
2	LOCK_CMTOCPU2MSGRAM0	R/W	0h	Locks write access to initialization and test control fields of Message RAM CMTOCPU2MSGRAM0 0: Write access allowed 1: Write access blocked Reset type: CM.RESETn
1	LOCK_CMTOCPU1MSGRAM1	R/W	0h	Locks write access to initialization and test control fields of Message RAM CMTOCPU1MSGRAM1 0: Write access allowed 1: Write access blocked Reset type: CM.RESETn
0	LOCK_CMTOCPU1MSGRAM0	R/W	0h	Locks write access to initialization and test control fields of Message RAM CMTOCPU1MSGRAM0 0: Write access allowed 1: Write access blocked Reset type: CM.RESETn

41.12.2.6 CMMSGxTEST Register (Offset = 24h) [Reset = 0h]

CMMSGxTEST is shown in Figure 41-22 and described in Table 41-19.

Return to the Summary Table.

CM Messae RAM TEST Register

Figure 41-22. CMMSGxTEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
TEST_CMTOCPU2MSGRAM1		TEST_CMTOCPU2MSGRAM0		TEST_CMTOCPU1MSGRAM1		TEST_CMTOCPU1MSGRAM0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 41-19. CMMSGxTEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7-6	TEST_CMTOCPU2MSGRAM1	R/W	0h	Selects the different modes for Message RAM CMTOCPU2MSGRAM1 00: Functional Mode. 01: Test mode to introduce Error in Data. Write updates only Data, ECC/Parity bits are not updated. 10: Test mode to introduce errors in ECC or Parity bits. ECC or Parity bits are visible on same memory map instead of Data bits. 11: Same as "00" but NMI is not generated on errors, used for diagnostics. Reset type: CM.RESETn
5-4	TEST_CMTOCPU2MSGRAM0	R/W	0h	Selects the different modes for Message RAM CMTOCPU2MSGRAM0 00: Functional Mode. 01: Test mode to introduce Error in Data. Write updates only Data, ECC/Parity bits are not updated. 10: Test mode to introduce errors in ECC or Parity bits. ECC or Parity bits are visible on same memory map instead of Data bits. 11: Same as "00" but NMI is not generated on errors, used for diagnostics. Reset type: CM.RESETn
3-2	TEST_CMTOCPU1MSGRAM1	R/W	0h	Selects the different modes for Message RAM CMTOCPU1MSGRAM1 00: Functional Mode. 01: Test mode to introduce Error in Data. Write updates only Data, ECC/Parity bits are not updated. 10: Test mode to introduce errors in ECC or Parity bits. ECC or Parity bits are visible on same memory map instead of Data bits. 11: Same as "00" but NMI is not generated on errors, used for diagnostics. Reset type: CM.RESETn

Table 41-19. CMMSGxTEST Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	TEST_CMTOCPU1MSGRAM0	R/W	0h	Selects the different modes for Message RAM CMTOCPU1MSGRAM0 00: Functional Mode. 01: Test mode to introduce Error in Data. Write updates only Data, ECC/Parity bits are not updated. 10: Test mode to introduce errors in ECC or Parity bits. ECC or Parity bits are visible on same memory map instead of Data bits. 11: Same as "00" but NMI is not generated on errors, used for diagnostics. Reset type: CM.RESETn

41.12.2.7 CMMSGxINIT Register (Offset = 28h) [Reset = 0h]

CMMSGxINIT is shown in Figure 41-23 and described in Table 41-20.

Return to the Summary Table.

CM Messae RAM Init Register

Figure 41-23. CMMSGxINIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				INIT_CMTOCPU2MSGRAM1	INIT_CMTOCPU2MSGRAM0	INIT_CMTOCPU1MSGRAM1	INIT_CMTOCPU1MSGRAM0
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 41-20. CMMSGxINIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	INIT_CMTOCPU2MSGRAM1	R-0/W1S	0h	RAM Initialization control for Message RAM CMTOCPU2MSGRAM1 0: None. 1: Start RAM Initialization. Reset type: CM.RESETn
2	INIT_CMTOCPU2MSGRAM0	R-0/W1S	0h	RAM Initialization control for Message RAM CMTOCPU2MSGRAM0 0: None. 1: Start RAM Initialization. Reset type: CM.RESETn
1	INIT_CMTOCPU1MSGRAM1	R-0/W1S	0h	RAM Initialization control for Message RAM CMTOCPU1MSGRAM1 0: None. 1: Start RAM Initialization. Reset type: CM.RESETn
0	INIT_CMTOCPU1MSGRAM0	R-0/W1S	0h	RAM Initialization control for Message RAM CMTOCPU1MSGRAM0 0: None. 1: Start RAM Initialization. Reset type: CM.RESETn

41.12.2.8 CMMSGxINITDONE Register (Offset = 2Ch) [Reset = 0h]

CMMSGxINITDONE is shown in Figure 41-24 and described in Table 41-21.

Return to the Summary Table.

CM Messae RAM Initialization Status Register

Figure 41-24. CMMSGxINITDONE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				INITDONE_CMTOCPU2MSGRAM1	INITDONE_CMTOCPU2MSGRAM0	INITDONE_CMTOCPU1MSGRAM1	INITDONE_CMTOCPU1MSGRAM0
R-0h				R-0h	R-0h	R-0h	R-0h

Table 41-21. CMMSGxINITDONE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	INITDONE_CMTOCPU2MSGRAM1	R	0h	RAM Initialization status for Message RAM CMTOCPU2MSGRAM1 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: CM.RESETn
2	INITDONE_CMTOCPU2MSGRAM0	R	0h	RAM Initialization status for Message RAM CMTOCPU2MSGRAM0 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: CM.RESETn
1	INITDONE_CMTOCPU1MSGRAM1	R	0h	RAM Initialization status for Message RAM CMTOCPU1MSGRAM1 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: CM.RESETn
0	INITDONE_CMTOCPU1MSGRAM0	R	0h	RAM Initialization status for Message RAM CMTOCPU1MSGRAM0 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: CM.RESETn

41.12.2.9 SxGROUP1_LOCK Register (Offset = 40h) [Reset = 0h]

SxGROUP1_LOCK is shown in Figure 41-25 and described in Table 41-22.

Return to the Summary Table.

Group1 S and E RAM Config Lock Register

Figure 41-25. SxGROUP1_LOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			LOCK_E0	LOCK_S3	LOCK_S2	LOCK_S1	LOCK_S0
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 41-22. SxGROUP1_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-5	RESERVED	R	0h	Reserved
4	LOCK_E0	R/W	0h	Locks write access to initialization and test control fields of E0 RAM 0: Write access allowed 1: Write access blocked Reset type: CM.RESETn
3	LOCK_S3	R/W	0h	Locks write access to initialization and test control fields of S3 RAM 0: Write access allowed 1: Write access blocked Reset type: CM.RESETn
2	LOCK_S2	R/W	0h	Locks write access to initialization and test control fields of S2 RAM 0: Write access allowed 1: Write access blocked Reset type: CM.RESETn
1	LOCK_S1	R/W	0h	Locks write access to initialization and test control fields of S1 RAM 0: Write access allowed 1: Write access blocked Reset type: CM.RESETn
0	LOCK_S0	R/W	0h	Locks write access to initialization and test control fields of S0 RAM 0: Write access allowed 1: Write access blocked Reset type: CM.RESETn

41.12.2.10 SxGROUP1_TEST Register (Offset = 44h) [Reset = 0h]

SxGROUP1_TEST is shown in Figure 41-26 and described in Table 41-23.

Return to the Summary Table.

Group1 S and E RAM TEST Register

Figure 41-26. SxGROUP1_TEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						TEST_E0	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
TEST_S3		TEST_S2		TEST_S1		TEST_S0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 41-23. SxGROUP1_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9-8	TEST_E0	R/W	0h	Selects the different modes for E0 RAM: 00: Functional Mode. 01: Test mode to introduce Error in Data. Write updates only Data, ECC/Parity bits are not updated. 10: Test mode to introduce errors in ECC or Parity bits. ECC or Parity bits are visible on same memory map instead of Data bits. 11: Same as "00" but NMI is not generated on errors, used for diagnostics. Reset type: CM.RESETn
7-6	TEST_S3	R/W	0h	Selects the different modes for S3 RAM: 00: Functional Mode. 01: Test mode to introduce Error in Data. Write updates only Data, ECC/Parity bits are not updated. 10: Test mode to introduce errors in ECC or Parity bits. ECC or Parity bits are visible on same memory map instead of Data bits. 11: Same as "00" but NMI is not generated on errors, used for diagnostics. Reset type: CM.RESETn
5-4	TEST_S2	R/W	0h	Selects the different modes for S2 RAM: 00: Functional Mode. 01: Test mode to introduce Error in Data. Write updates only Data, ECC/Parity bits are not updated. 10: Test mode to introduce errors in ECC or Parity bits. ECC or Parity bits are visible on same memory map instead of Data bits. 11: Same as "00" but NMI is not generated on errors, used for diagnostics. Reset type: CM.RESETn

Table 41-23. SxGROUP1_TEST Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	TEST_S1	R/W	0h	Selects the different modes for S1 RAM: 00: Functional Mode. 01: Test mode to introduce Error in Data. Write updates only Data, ECC/Parity bits are not updated. 10: Test mode to introduce errors in ECC or Parity bits. ECC or Parity bits are visible on same memory map instead of Data bits. 11: Same as "00" but NMI is not generated on errors, used for diagnostics. Reset type: CM.RESETn
1-0	TEST_S0	R/W	0h	Selects the different modes for S0 RAM: 00: Functional Mode. 01: Test mode to introduce Error in Data. Write updates only Data, ECC/Parity bits are not updated. 10: Test mode to introduce errors in ECC or Parity bits. ECC or Parity bits are visible on same memory map instead of Data bits. 11: Same as "00" but NMI is not generated on errors, used for diagnostics. Reset type: CM.RESETn

41.12.2.11 SxGROUP1_INIT Register (Offset = 48h) [Reset = 0h]

SxGROUP1_INIT is shown in Figure 41-27 and described in Table 41-24.

Return to the Summary Table.

Group1 S and E RAM Init Register

Figure 41-27. SxGROUP1_INIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			INIT_E0	INIT_S3	INIT_S2	INIT_S1	INIT_S0
R-0h			R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 41-24. SxGROUP1_INIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-5	RESERVED	R	0h	Reserved
4	INIT_E0	R-0/W1S	0h	RAM Initialization control for E0 RAM: 0: None. 1: Start RAM Initialization. Reset type: CM.RESETn
3	INIT_S3	R-0/W1S	0h	RAM Initialization control for S3 RAM: 0: None. 1: Start RAM Initialization. Reset type: CM.RESETn
2	INIT_S2	R-0/W1S	0h	RAM Initialization control for S2 RAM: 0: None. 1: Start RAM Initialization. Reset type: CM.RESETn
1	INIT_S1	R-0/W1S	0h	RAM Initialization control for S1 RAM: 0: None. 1: Start RAM Initialization. Reset type: CM.RESETn
0	INIT_S0	R-0/W1S	0h	RAM Initialization control for S0 RAM: 0: None. 1: Start RAM Initialization. Reset type: CM.RESETn

41.12.2.12 SxGROUP1_INITDONE Register (Offset = 4Ch) [Reset = 0h]

SxGROUP1_INITDONE is shown in Figure 41-28 and described in Table 41-25.

Return to the Summary Table.

Group1 S and E RAM Initialization Status Register

Figure 41-28. SxGROUP1_INITDONE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			INITDONE_E0	INITDONE_S3	INITDONE_S2	INITDONE_S1	INITDONE_S0
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h

Table 41-25. SxGROUP1_INITDONE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-5	RESERVED	R	0h	Reserved
4	INITDONE_E0	R	0h	RAM Initialization status for E0 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: CM.RESETn
3	INITDONE_S3	R	0h	RAM Initialization status for S3 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: CM.RESETn
2	INITDONE_S2	R	0h	RAM Initialization status for S2 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: CM.RESETn
1	INITDONE_S1	R	0h	RAM Initialization status for S1 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: CM.RESETn
0	INITDONE_S0	R	0h	RAM Initialization status for S0 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: CM.RESETn

41.12.2.13 ROM_LOCK Register (Offset = 80h) [Reset = 0h]

ROM_LOCK is shown in Figure 41-29 and described in Table 41-26.

Return to the Summary Table.

ROM Config Lock Register

Figure 41-29. ROM_LOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK_BOOTROM
R-0h							R/W-0h

Table 41-26. ROM_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	LOCK_BOOTROM	R/W	0h	Locks write access to test control fields (TEST and FORCE_ERROR) of BOOTROM 0: Write access allowed 1: Write access blocked Reset type: CM.RESETn

41.12.2.14 ROM_TEST Register (Offset = 84h) [Reset = 0h]

ROM_TEST is shown in Figure 41-30 and described in Table 41-27.

Return to the Summary Table.

ROM TEST Register

Figure 41-30. ROM_TEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						TEST_BOOTROM	
R-0h						R/W-0h	

Table 41-27. ROM_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	TEST_BOOTROM	R/W	0h	Selects the different modes for BOOTROM: 00: Functional Mode. 01: same as "00" but Parity check on data read is disabled (for debug) 10: Parity Bits are visible on memory map (for debug) 11: Same as "00" but NMI is not generated on errors, used for diagnostics. (for diagnostics) Reset type: CM.RESETn

41.12.2.15 ROM_FORCE_ERROR Register (Offset = 88h) [Reset = 0h]

ROM_FORCE_ERROR is shown in Figure 41-31 and described in Table 41-28.

Return to the Summary Table.

ROM Force Error register

Figure 41-31. ROM_FORCE_ERROR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							FORCE_BOOTROM_ERROR
R-0h							R/W-0h

Table 41-28. ROM_FORCE_ERROR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	FORCE_BOOTROM_ERROR	R/W	0h	Force parity error by feeding inverted Parity bit to Parity checking logic. Reset type: CM.RESETn

41.12.2.16 PERI_MEM_TEST_LOCK Register (Offset = A0h) [Reset = 0h]

PERI_MEM_TEST_LOCK is shown in Figure 41-32 and described in Table 41-29.

Return to the Summary Table.

Peripheral Memory Test Lock Register

Figure 41-32. PERI_MEM_TEST_LOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK_PERI_MEM_TEST_CONTROL
R-0h							R/W-0h

Table 41-29. PERI_MEM_TEST_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	LOCK_PERI_MEM_TEST_CONTROL	R/W	0h	Locks write access to register PERI_MEM_TEST_CONTROL 0: Write access allowed 1: Write access blocked Reset type: CM.RESETn

41.12.2.17 PERI_MEM_TEST_CONTROL Register (Offset = A4h) [Reset = 0h]

PERI_MEM_TEST_CONTROL is shown in Figure 41-33 and described in Table 41-30.

Return to the Summary Table.

Peripheral Memory Test control Register

Figure 41-33. PERI_MEM_TEST_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		EtherCAT_MEM_FORCE_ERROR	EtherCAT_TEST_ENABLE	RESERVED	RESERVED	EMAC_MEM_FORCE_ERROR	EMAC_TEST_ENABLE
R-0h		R/W-0h	R/W-0h	R-0h	R-0h	R/W-0h	R/W-0h

Table 41-30. PERI_MEM_TEST_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	EtherCAT_MEM_FORCE_ERROR	R/W	0h	Force error bit 0 : No effect 1 : Parity bit going to Parity checker module of EtherCAT is inverted to introduce parity Error Reset type: CM.RESETn
4	EtherCAT_TEST_ENABLE	R/W	0h	Selects EtherCAT test mode 0 : EtherCAT test mode disabled, Error on EtherCAT memory read access will generate NMI 1 : EtherCAT test mode enabled, Error on EtherCAT memory read access will NOT generate NMI, used for diagnostics Reset type: CM.RESETn
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	EMAC_MEM_FORCE_ERROR	R/W	0h	Force error bit 0 : No effect 1 : Parity bit going to Parity checker module of EMAC is inverted to introduce parity Error Reset type: CM.RESETn
0	EMAC_TEST_ENABLE	R/W	0h	Selects EMAC test mode 0 : EMAC test mode disabled, Error on EMAC memory read access will generate NMI 1 : EMAC test mode enabled, Error on EMAC memory read access will NOT generate NMI, used for diagnostics Reset type: CM.RESETn

41.12.3 CM_MEMORYDIAGERROR_REGS Registers

Table 41-31 lists the memory-mapped registers for the CM_MEMORYDIAGERROR_REGS registers. All register offset addresses not listed in Table 41-31 should be considered as reserved locations and the register contents should not be modified.

Table 41-31. CM_MEMORYDIAGERROR_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	DIAGERRFLG	Error Flag Register		Go
8h	DIAGERRCLR	Error Flag Clear Register		Go
Ch	DIAGERRADDR	Read Error Address		Go

Complex bit access types are encoded to fit into small table cells. Table 41-32 shows the codes that are used for access types in this section.

Table 41-32. CM_MEMORYDIAGERROR_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

41.12.3.1 DIAGERRFLG Register (Offset = 0h) [Reset = 0h]

DIAGERRFLG is shown in Figure 41-34 and described in Table 41-33.

Return to the Summary Table.

Error Flag Register

Figure 41-34. DIAGERRFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CWRERROR	CRDERROR	UCWRERROR	UCRDERROR
R-0h				R-0h	R-0h	R-0h	R-0h

Table 41-33. DIAGERRFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	CWRERROR	R	0h	Correctable Write Error Flag for diagnostics 0: No Error. 1:Correctable error occurred on write to memory under test Notes: 1. Only M4 access to System RAM or ROM under test will set this bit 2. Any access to EtherCAT and Ethernet IP memory under test will set this bit. Reset type: CM.RESETn
2	CRDERROR	R	0h	Correctable Read Error Flag for diagnostics 0: No Error. 1: Correctable error occurred on read to memory under test Notes: 1. Only M4 access to System RAM or ROM under test will set this bit 2. Any access to EtherCAT and Ethernet IP memory under test will set this bit. Reset type: CM.RESETn
1	UCWRERROR	R	0h	Uncorrectable Write Error Flag for diagnostics 0: No Error. 1: Uncorrectable error occurred on write to memory under test Notes: 1. Only M4 access to System RAM or ROM under test will set this bit 2. Any access to EtherCAT and Ethernet IP memory under test will set this bit. Reset type: CM.RESETn

Table 41-33. DIAGERRFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	UCRDERROR	R	0h	Uncorrectable Read Error Flag for diagnostics 0: No Error. 1: Uncorrectable error occurred on read to memory under test Notes: 1. Only M4 access to System RAM or ROM under test will set this bit 2. Any access to EtherCAT and Ethernet IP memory under test will set this bit. Reset type: CM.RESETn

41.12.3.2 DIAGERRCLR Register (Offset = 8h) [Reset = 0h]

DIAGERRCLR is shown in Figure 41-35 and described in Table 41-34.

Return to the Summary Table.

Error Flag Clear Register

Figure 41-35. DIAGERRCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CWRERROR	CRDERROR	UCWRERROR	UCRDERROR
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 41-34. DIAGERRCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	CWRERROR	R-0/W1S	0h	0: No action. 1: CWRERROR flag will be cleared. Reset type: CM.RESETn
2	CRDERROR	R-0/W1S	0h	0: No action. 1: CRDERROR flag will be cleared. Reset type: CM.RESETn
1	UCWRERROR	R-0/W1S	0h	0: No action. 1: UCWRERROR flag will be cleared. Reset type: CM.RESETn
0	UCRDERROR	R-0/W1S	0h	0: No action. 1: UCRDERROR flag will be cleared. Reset type: CM.RESETn

41.12.3.3 DIAGERRADDR Register (Offset = Ch) [Reset = 0h]

DIAGERRADDR is shown in Figure 41-36 and described in Table 41-35.

Return to the Summary Table.

Read Error Address

Figure 41-36. DIAGERRADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EADDR																															
R-0h																															

Table 41-35. DIAGERRADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	EADDR	R	0h	RAM, ROM : This register captures the address location at which read or write access resulted in ECC/Parity error when in test mode = "11". EMAC RAM : This register captures the address location at which read or write access resulted in Parity error when PERI_MEM_TEST_CONTROL.EMAC_TEST_ENABLE is set EtherCAT RAM : This register captures the address location at which read or write access resulted in Parity error when PERI_MEM_TEST_CONTROL.EtherCAT_TEST_ENABLE is set Reset type: CM.RESETn

41.12.4 CM_MEMORYERROR_REGS Registers

Table 41-36 lists the memory-mapped registers for the CM_MEMORYERROR_REGS registers. All register offset addresses not listed in Table 41-36 should be considered as reserved locations and the register contents should not be modified.

Table 41-36. CM_MEMORYERROR_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	UCERRFLG	Uncorrectable Error Flag Register		Go
4h	UCERRSET	Uncorrectable Error Flag Set Register		Go
8h	UCERRCLR	Uncorrectable Error Flag Clear Register		Go
Ch	UCM4EADDR	Uncorrectable M4 Error Address		Go
10h	UCEMACEADDR	Uncorrectable EMAC Error Address		Go
14h	UCuDMAEADDR	Uncorrectable uDMA Error Address		Go
18h	UCEtherCATMEMREADDR	Uncorrectable EtherCAT IP RAM Read Error Address		Go
1Ch	UCEMACMEMREADDR	Uncorrectable EMAC IP RAM Read Error Address		Go
50h	BUSFAULTFLG	BusFault Flag register		Go
54h	BUSFAULTCLR	BusFault Flag clear register		Go
58h	M4BUSFAULTADDR	M4 busfault address		Go
5Ch	uDMABUSFAULTADDR	uDMA busfault address		Go
60h	EMACBUSFAULTADDR	EMAC busfault address		Go
80h	CERRFLG	Correctable Error Flag Register		Go
84h	CERRSET	Correctable Error Flag Set Register		Go
88h	CERRCLR	Correctable Error Flag Clear Register		Go
8Ch	CM4EADDR	Correctable M4 Error Address		Go
90h	CEMACEADDR	Correctable EMAC Error Address		Go
94h	CuDMAEADDR	Correctable uDMA Error Address		Go
C0h	CERRCNT	Correctable Error Count Register		Go
C4h	CERRTHRES	Correctable Error Threshold Value Register		Go
C8h	CEINTFLG	Correctable Error Interrupt Flag Status Register		Go
CCh	CEINTSET	Correctable Error Interrupt Flag Set Register		Go
D0h	CEINTCLR	Correctable Error Interrupt Flag Clear Register		Go
D4h	CEINTEN	Correctable Error Interrupt Enable Register		Go

Complex bit access types are encoded to fit into small table cells. Table 41-37 shows the codes that are used for access types in this section.

Table 41-37. CM_MEMORYERROR_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		

Table 41-37. CM_MEMORYERROR_REGS Access Type Codes (continued)

Access Type	Code	Description
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

41.12.4.1 UCERRFLG Register (Offset = 0h) [Reset = 0h]

UCERRFLG is shown in Figure 41-37 and described in Table 41-38.

Return to the Summary Table.

Uncorrectable Error Flag Register

Figure 41-37. UCERRFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
EMACMEMRDERR	EtherCATMEMRDERR	uDMAWRERR	uDMARDERR	RESERVED	EMACRDERR	M4WRERR	M4RDERR
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 41-38. UCERRFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	EMACMEMRDERR	R	0h	EMAC IP RAM Uncorrectable Read Error Flag 0: No Error. 1: Uncorrectable error occurred during EMAC IP memory read. NMI interrupt to M4 is generated when this flag is set Reset type: CM.RESETn
6	EtherCATMEMRDERR	R	0h	EtherCAT IP RAM Uncorrectable Read Error Flag 0: No Error. 1: Uncorrectable error occurred during EtherCAT IP memory read. NMI interrupt to M4 is generated when this flag is set Reset type: CM.RESETn
5	uDMAWRERR	R	0h	uDMA Uncorrectable Write Error Flag 0: No Error. 1: Uncorrectable error occurred during uDMA Write NMI interrupt to M4 is generated when this flag is set Reset type: CM.RESETn
4	uDMARDERR	R	0h	uDMA Uncorrectable Read Error Flag 0: No Error. 1: Uncorrectable error occurred during uDMA read. NMI interrupt to M4 is generated when this flag is set Reset type: CM.RESETn
3	RESERVED	R	0h	Reserved
2	EMACRDERR	R	0h	EMAC Uncorrectable Read Error Flag 0: No Error. 1: Uncorrectable error occurred during EMAC read. NMI interrupt to M4 is generated when this flag is set Reset type: CM.RESETn

Table 41-38. UCERRFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	M4WRERR	R	0h	M4 Uncorrectable Write Error Flag 0: No Error. 1: Uncorrectable error occurred during M4 Write NMI interrupt to M4 is generated when this flag is set Reset type: CM.RESETn
0	M4RDERR	R	0h	M4 Uncorrectable Read Error Flag 0: No Error. 1: Uncorrectable error occurred during M4 read. NMI interrupt to M4 is generated when this flag is set Reset type: CM.RESETn

41.12.4.2 UCERRSET Register (Offset = 4h) [Reset = 0h]

UCERRSET is shown in Figure 41-38 and described in Table 41-39.

Return to the Summary Table.

Uncorrectable Error Flag Set Register

Figure 41-38. UCERRSET Register

31	30	29	28	27	26	25	24
KEY							
R-0/W1S-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W1S-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
EMACMEMRDERR	EtherCATMEMRDERR	uDMAWRERR	uDMARDERR	RESERVED	EMACRDERR	M4WRERR	M4RDERR
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 41-39. UCERRSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W1S	0h	Write '1' to SET bits will take affect only if [31:16] is written with Data 0xA5A5 Reset type: CM.RESETn
15-8	RESERVED	R	0h	Reserved
7	EMACMEMRDERR	R-0/W1S	0h	0: No action. 1: EMAC IP RAM Read Error Flag in UCERRFLG register will be set. Reset type: CM.RESETn
6	EtherCATMEMRDERR	R-0/W1S	0h	0: No action. 1: EtherCAT IP RAM Read Error Flag in UCERRFLG register will be set. Reset type: CM.RESETn
5	uDMAWRERR	R-0/W1S	0h	0: No action. 1: uDMA Write Error Flag in UCERRFLG register will be set. Reset type: CM.RESETn
4	uDMARDERR	R-0/W1S	0h	0: No action. 1: uDMA Read Error Flag in UCERRFLG register will be set. Reset type: CM.RESETn
3	RESERVED	R-0/W1S	0h	Reserved
2	EMACRDERR	R-0/W1S	0h	0: No action. 1: EMAC Read Error Flag in UCERRFLG register will be set. Reset type: CM.RESETn
1	M4WRERR	R-0/W1S	0h	0: No action. 1: M4 Write Error Flag in UCERRFLG register will be set. Reset type: CM.RESETn
0	M4RDERR	R-0/W1S	0h	0: No action. 1: M4 Read Error Flag in UCERRFLG register will be set. Reset type: CM.RESETn

41.12.4.3 UCERRCLR Register (Offset = 8h) [Reset = 0h]

UCERRCLR is shown in Figure 41-39 and described in Table 41-40.

Return to the Summary Table.

Uncorrectable Error Flag Clear Register

Figure 41-39. UCERRCLR Register

31	30	29	28	27	26	25	24
KEY							
R-0/W1S-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W1S-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
EMACMEMRDERR	EtherCATMEMRDERR	uDMAWRERR	uDMARDERR	RESERVED	EMACRDERR	M4WRERR	M4RDERR
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 41-40. UCERRCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W1S	0h	Write '1' to CLEAR bits will take affect only if [31:16] is written with Data 0xA5A5 Reset type: CM.RESETn
15-8	RESERVED	R	0h	Reserved
7	EMACMEMRDERR	R-0/W1S	0h	0: No action. 1: EMAC IP RAM Read Error Flag in UCERRFLG register will be cleared. Reset type: CM.RESETn
6	EtherCATMEMRDERR	R-0/W1S	0h	0: No action. 1: EtherCAT IP RAM Read Error Flag in UCERRFLG register will be cleared. Reset type: CM.RESETn
5	uDMAWRERR	R-0/W1S	0h	0: No action. 1: uDMA Write Error Flag in UCERRFLG register will be cleared. Reset type: CM.RESETn
4	uDMARDERR	R-0/W1S	0h	0: No action. 1: uDMA Read Error Flag in UCERRFLG register will be cleared. Reset type: CM.RESETn
3	RESERVED	R-0/W1S	0h	Reserved
2	EMACRDERR	R-0/W1S	0h	0: No action. 1: EMAC Read Error Flag in UCERRFLG register will be cleared . Reset type: CM.RESETn
1	M4WRERR	R-0/W1S	0h	0: No action. 1: M4 Write Error Flag in UCERRFLG register will be cleared. Reset type: CM.RESETn
0	M4RDERR	R-0/W1S	0h	0: No action. 1: M4 Read Error Flag in UCERRFLG register will be cleared. Reset type: CM.RESETn

41.12.4.4 UCM4EADDR Register (Offset = Ch) [Reset = 0h]

UCM4EADDR is shown in Figure 41-40 and described in Table 41-41.

Return to the Summary Table.

Uncorrectable M4 Error Address

Figure 41-40. UCM4EADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCM4EADDR																															
R-0h																															

Table 41-41. UCM4EADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UCM4EADDR	R	0h	This register captures the address location at which M4 read or write access resulted in uncorrectable error. Reset type: CM.RESETn

41.12.4.5 UCEMACEADDR Register (Offset = 10h) [Reset = 0h]

UCEMACEADDR is shown in Figure 41-41 and described in Table 41-42.

Return to the Summary Table.

Uncorrectable EMAC Error Address

Figure 41-41. UCEMACEADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCEMACEADDR																															
R-0h																															

Table 41-42. UCEMACEADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UCEMACEADDR	R	0h	This register captures the address location at which EMAC read or write access resulted in uncorrectable error. Reset type: CM.RESETn

41.12.4.6 UCuDMAEADDR Register (Offset = 14h) [Reset = 0h]

UCuDMAEADDR is shown in Figure 41-42 and described in Table 41-43.

Return to the Summary Table.

Uncorrectable uDMA Error Address

Figure 41-42. UCuDMAEADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCuDMAEADDR																															
R-0h																															

Table 41-43. UCuDMAEADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UCuDMAEADDR	R	0h	This register captures the address location at which uDMA read or write access resulted in uncorrectable error. Reset type: CM.RESETn

41.12.4.7 UCEtherCATMEMREADDR Register (Offset = 18h) [Reset = 0h]

UCEtherCATMEMREADDR is shown in Figure 41-43 and described in Table 41-44.

Return to the Summary Table.

Uncorrectable EtherCAT IP RAM Read Error Address

Figure 41-43. UCEtherCATMEMREADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCEtherCATMEMREADDR																															
R-0h																															

Table 41-44. UCEtherCATMEMREADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UCEtherCATMEMREADDR	R	0h	This register captures the address location at which EtherCAT IP RAM access resulted in uncorrectable ECC/Parity error. Reset type: CM.RESETn

41.12.4.8 UCEMACMEMREADDR Register (Offset = 1Ch) [Reset = 0h]

UCEMACMEMREADDR is shown in Figure 41-44 and described in Table 41-45.

Return to the Summary Table.

Uncorrectable EMAC IP RAM Read Error Address

Figure 41-44. UCEMACMEMREADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCEMACMEMREADDR																															
R-0h																															

Table 41-45. UCEMACMEMREADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UCEMACMEMREADDR	R	0h	This register captures the address location at which EMAC IP RAM access resulted in uncorrectable ECC/Parity error. Reset type: CM.RESETn

41.12.4.9 BUSFAULTFLG Register (Offset = 50h) [Reset = 0h]

BUSFAULTFLG is shown in Figure 41-45 and described in Table 41-46.

Return to the Summary Table.

BusFault Flag register

Figure 41-45. BUSFAULTFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					EMACBUSFAULT	UDMABUSFAULT	M4BUSFAULT
R-0h					R-0h	R-0h	R-0h

Table 41-46. BUSFAULTFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-3	RESERVED	R	0h	Reserved
2	EMACBUSFAULT	R	0h	EMAC busfault Flag 0: No Error. 1: EMAC access encountered busfault Reset type: CM.RESETn
1	UDMABUSFAULT	R	0h	UDMA busfault Flag 0: No Error. 1: UDMA access encountered busfault Reset type: CM.RESETn
0	M4BUSFAULT	R	0h	M4 busfault Flag 0: No Error. 1: M4 access encountered busfault Reset type: CM.RESETn

41.12.4.10 BUSFAULTCLR Register (Offset = 54h) [Reset = 0h]

BUSFAULTCLR is shown in Figure 41-46 and described in Table 41-47.

Return to the Summary Table.

BusFault Flag clear register

Figure 41-46. BUSFAULTCLR Register

31	30	29	28	27	26	25	24
KEY							
R-0/W1S-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W1S-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					EMACBUSFAULT	UDMABUSFAULT	M4BUSFAULT
R-0h					R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 41-47. BUSFAULTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W1S	0h	Write '1' to CLEAR bits will take affect only if [31:16] is written with Data 0xA5A5 Reset type: CM.RESETn
15-3	RESERVED	R	0h	Reserved
2	EMACBUSFAULT	R-0/W1S	0h	0: No action. 1: EMAC busfault flag will be cleared Reset type: CM.RESETn
1	UDMABUSFAULT	R-0/W1S	0h	0: No action. 1: UDMA busfault flag will be cleared Reset type: CM.RESETn
0	M4BUSFAULT	R-0/W1S	0h	0: No action. 1: M4 busfault flag will be cleared Reset type: CM.RESETn

41.12.4.11 M4BUSFAULTADDR Register (Offset = 58h) [Reset = 0h]

M4BUSFAULTADDR is shown in Figure 41-47 and described in Table 41-48.

Return to the Summary Table.

M4 busfault address

Figure 41-47. M4BUSFAULTADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M4BUSFAULTADDRESS																															
R-0h																															

Table 41-48. M4BUSFAULTADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	M4BUSFAULTADDRESS	R	0h	This register captures the address location at M4 access encountered busfault. Value of this register is valid when BUSFAULTFLG.M4BUSFAULT flag is set. Capture first busfault address, capture can be reenabled by clearing BUSFAULTFLG.M4BUSFAULT . Reset type: CM.RESETn

41.12.4.12 uDMABUSFAULTADDR Register (Offset = 5Ch) [Reset = 0h]

uDMABUSFAULTADDR is shown in Figure 41-48 and described in Table 41-49.

Return to the Summary Table.

uDMA busfault address

Figure 41-48. uDMABUSFAULTADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UDMABUSFAULTADDRESS																															
R-0h																															

Table 41-49. uDMABUSFAULTADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UDMABUSFAULTADDRESS	R	0h	This register captures the address location at M4 access encountered busfault. Value of this register is valid when BUSFAULTFLG.UDMABUSFAULT flag is set. Capture first busfault address, capture can be reenabled by clearing BUSFAULTFLG.UDMABUSFAULT . Reset type: CM.RESETn

41.12.4.13 EMACBUSFAULTADDR Register (Offset = 60h) [Reset = 0h]

EMACBUSFAULTADDR is shown in Figure 41-49 and described in Table 41-50.

Return to the Summary Table.

EMAC busfault address

Figure 41-49. EMACBUSFAULTADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EMACBUSFAULTADDRESS																															
R-0h																															

Table 41-50. EMACBUSFAULTADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	EMACBUSFAULTADDRESS	R	0h	This register captures the address location at M4 access encountered busfault. Value of this register is valid when BUSFAULTFLG.EMACBUSFAULT flag is set. Capture first busfault address, capture can be reenabled by clearing BUSFAULTFLG.EMACBUSFAULT . Reset type: CM.RESETn

41.12.4.14 CERRFLG Register (Offset = 80h) [Reset = 0h]

CERRFLG is shown in Figure 41-50 and described in Table 41-51.

Return to the Summary Table.

Correctable Error Flag Register

Figure 41-50. CERRFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	uDMAWRERR	uDMARDERR	RESERVED	EMACRDERR	M4WRERR	M4RDERR
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 41-51. CERRFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	RESERVED	R	0h	Reserved
6	RESERVED	R	0h	Reserved
5	uDMAWRERR	R	0h	uDMA Correctable Write Error Flag 0: No Error. 1: Correctable error occurred during uDMA write Reset type: CM.RESETn
4	uDMARDERR	R	0h	uDMA Correctable Read Error Flag 0: No Error. 1: Correctable error occurred during uDMA read Reset type: CM.RESETn
3	RESERVED	R	0h	Reserved
2	EMACRDERR	R	0h	EMAC Correctable Read Error Flag 0: No Error. 1: Correctable error occurred during EMAC read Reset type: CM.RESETn
1	M4WRERR	R	0h	M4 Correctable Write Error Flag 0: No Error. 1: Correctable error occurred during M4 write Reset type: CM.RESETn
0	M4RDERR	R	0h	M4 Correctable Read Error Flag 0: No Error. 1: Correctable error occurred during M4 read Reset type: CM.RESETn

41.12.4.15 CERRSET Register (Offset = 84h) [Reset = 0h]

CERRSET is shown in Figure 41-51 and described in Table 41-52.

Return to the Summary Table.

Correctable Error Flag Set Register

Figure 41-51. CERRSET Register

31	30	29	28	27	26	25	24
KEY							
R-0/W1S-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W1S-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	uDMAWRERR	uDMARDERR	RESERVED	EMACRDERR	M4WRERR	M4RDERR
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 41-52. CERRSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W1S	0h	Write '1' to SET bits will take affect only if [31:16] is written with Data 0xA5A5 Reset type: CM.RESETn
15-8	RESERVED	R	0h	Reserved
7	RESERVED	R-0/W1S	0h	Reserved
6	RESERVED	R-0/W1S	0h	Reserved
5	uDMAWRERR	R-0/W1S	0h	0: No action. 1: uDMA Write Error Flag in CERRFLG register will be set Reset type: CM.RESETn
4	uDMARDERR	R-0/W1S	0h	0: No action. 1: uDMA Read Error Flag in CERRFLG register will be set Reset type: CM.RESETn
3	RESERVED	R-0/W1S	0h	Reserved
2	EMACRDERR	R-0/W1S	0h	0: No action. 1: EMAC Read Error Flag in CERRFLG register will be set Reset type: CM.RESETn
1	M4WRERR	R-0/W1S	0h	0: No action. 1: M4 Write Error Flag in CERRFLG register will be set Reset type: CM.RESETn
0	M4RDERR	R-0/W1S	0h	0: No action. 1: M4 Read Error Flag in CERRFLG register will be set Reset type: CM.RESETn

41.12.4.16 CERRCLR Register (Offset = 88h) [Reset = 0h]

CERRCLR is shown in Figure 41-52 and described in Table 41-53.

Return to the Summary Table.

Correctable Error Flag Clear Register

Figure 41-52. CERRCLR Register

31	30	29	28	27	26	25	24
KEY							
R-0/W1S-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W1S-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	uDMAWRERR	uDMARDERR	RESERVED	EMACRDERR	M4WRERR	M4RDERR
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 41-53. CERRCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W1S	0h	Write '1' to CLEAR bits will take affect only if [31:16] is written with Data 0xA5A5 Reset type: CM.RESETn
15-8	RESERVED	R	0h	Reserved
7	RESERVED	R-0/W1S	0h	Reserved
6	RESERVED	R-0/W1S	0h	Reserved
5	uDMAWRERR	R-0/W1S	0h	0: No action. 1: uDMA Write Error Flag in CERRFLG register will be cleared. Reset type: CM.RESETn
4	uDMARDERR	R-0/W1S	0h	0: No action. 1: uDMA Read Error Flag in CERRFLG register will be cleared. Reset type: CM.RESETn
3	RESERVED	R-0/W1S	0h	Reserved
2	EMACRDERR	R-0/W1S	0h	0: No action. 1: EMAC Read Error Flag in CERRFLG register will be cleared . Reset type: CM.RESETn
1	M4WRERR	R-0/W1S	0h	0: No action. 1: M4 Write Error Flag in CERRFLG register will be cleared. Reset type: CM.RESETn
0	M4RDERR	R-0/W1S	0h	0: No action. 1: M4 Read Error Flag in CERRFLG register will be cleared. Reset type: CM.RESETn

41.12.4.17 CM4EADDR Register (Offset = 8Ch) [Reset = 0h]

CM4EADDR is shown in Figure 41-53 and described in Table 41-54.

Return to the Summary Table.

Correctable M4 Error Address

Figure 41-53. CM4EADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CM4EADDR																															
R-0h																															

Table 41-54. CM4EADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CM4EADDR	R	0h	This register captures the address location at which M4 read or write access resulted in correctable ECC error. Reset type: CM.RESETn

41.12.4.18 CEMACEADDR Register (Offset = 90h) [Reset = 0h]

CEMACEADDR is shown in Figure 41-54 and described in Table 41-55.

Return to the Summary Table.

Correctable EMAC Error Address

Figure 41-54. CEMACEADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CEMACEADDR																															
R-0h																															

Table 41-55. CEMACEADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CEMACEADDR	R	0h	This register captures the address location at which EMAC read or write access resulted in correctable ECC error. Reset type: CM.RESETn

41.12.4.19 CuDMAEADDR Register (Offset = 94h) [Reset = 0h]

CuDMAEADDR is shown in Figure 41-55 and described in Table 41-56.

Return to the Summary Table.

Correctable uDMA Error Address

Figure 41-55. CuDMAEADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CuDMAEADDR																															
R-0h																															

Table 41-56. CuDMAEADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CuDMAEADDR	R	0h	This register captures the address location at which uDMA read or write access resulted in correctable ECC error. Reset type: CM.RESETn

41.12.4.20 CERRCNT Register (Offset = C0h) [Reset = 0h]

CERRCNT is shown in Figure 41-56 and described in Table 41-57.

Return to the Summary Table.

Correctable Error Count Register

Figure 41-56. CERRCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CERRCNT																															
R/W-0h																															

Table 41-57. CERRCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CERRCNT	R/W	0h	This register holds the count of how many times correctable error occurred. It will stop counting once threshold value is reached Counter is reset to 0x0 automatically upon clearing correctable interrupt flag i.e. by writing '1' to CEINTCLR[CEINTCLR] Reset type: CM.RESETn

41.12.4.21 CERRTHRES Register (Offset = C4h) [Reset = 0h]

CERRTHRES is shown in Figure 41-57 and described in Table 41-58.

Return to the Summary Table.

Correctable Error Threshold Value Register

Figure 41-57. CERRTHRES Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CERRTHRES																															
R/W-0h																															

Table 41-58. CERRTHRES Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CERRTHRES	R/W	0h	When value in CERRCNT register is greater than or equal to the value configured in this register, correctable interrupt gets generated if enabled. Reset type: CM.RESETn

41.12.4.22 CEINTFLG Register (Offset = C8h) [Reset = 0h]

CEINTFLG is shown in Figure 41-58 and described in Table 41-59.

Return to the Summary Table.

Correctable Error Interrupt Flag Status Register

Figure 41-58. CEINTFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CEINTFLAG
R-0h							R-0h

Table 41-59. CEINTFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	CEINTFLAG	R	0h	Total corrected error count exceeded threshold Flag 0: Total correctable errors < Threshold value configured in CERRTHRES register. 1: Total correctable errors = Threshold value configured in CERRTHRES register. Reset type: CM.RESETn

41.12.4.23 CEINTSET Register (Offset = CCh) [Reset = 0h]

CEINTSET is shown in Figure 41-59 and described in Table 41-60.

Return to the Summary Table.

Correctable Error Interrupt Flag Set Register

Figure 41-59. CEINTSET Register

31	30	29	28	27	26	25	24
KEY							
R-0/W1S-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W1S-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CEINTSET
R-0h							R-0/W1S-0h

Table 41-60. CEINTSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W1S	0h	Write '1' to SET bits will take affect only if [31:16] is written with Data 0xA5A5 Reset type: CM.RESETn
15-1	RESERVED	R	0h	Reserved
0	CEINTSET	R-0/W1S	0h	0: No action. 1: Total corrected error count exceeded flag in CEINTFLG register will be set and interrupt will be generated if enabled. Reset type: CM.RESETn

41.12.4.24 CEINTCLR Register (Offset = D0h) [Reset = 0h]

CEINTCLR is shown in Figure 41-60 and described in Table 41-61.

Return to the Summary Table.

Correctable Error Interrupt Flag Clear Register

Figure 41-60. CEINTCLR Register

31	30	29	28	27	26	25	24
KEY							
R-0/W1S-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W1S-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CEINTCLR
R-0h							R-0/W1S-0h

Table 41-61. CEINTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W1S	0h	Write '1' to CLEAR bits will take affect only if [31:16] is written with Data 0xA5A5 Reset type: CM.RESETn
15-1	RESERVED	R	0h	Reserved
0	CEINTCLR	R-0/W1S	0h	0: No action. 1: Total corrected error count exceeded flag in CEINTFLG register will be cleared. Reset type: CM.RESETn

41.12.4.25 CEINTEN Register (Offset = D4h) [Reset = 0h]

CEINTEN is shown in Figure 41-61 and described in Table 41-62.

Return to the Summary Table.

Correctable Error Interrupt Enable Register

Figure 41-61. CEINTEN Register

31	30	29	28	27	26	25	24
KEY							
R-0/W1S-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W1S-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CEINTEN
R-0h							R/W-0h

Table 41-62. CEINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W1S	0h	Write to CEINTEN is allowed only if [31:16] is written with Data 0xA5A5 Reset type: CM.RESETn
15-1	RESERVED	R	0h	Reserved
0	CEINTEN	R/W	0h	0: Correctable Error Interrupt is disabled. 1: Correctable Error Interrupt is enabled. Reset type: CM.RESETn

41.12.5 CMSYSCTL_REGS Registers

Table 41-63 lists the memory-mapped registers for the CMSYSCTL_REGS registers. All register offset addresses not listed in Table 41-63 should be considered as reserved locations and the register contents should not be modified.

Table 41-63. CMSYSCTL_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CMPCLKCR0	CM Peripheral clock gating register 0.		Go
2h	CMPCLKCR1	CM Peripheral clock gating register 1.		Go
4h	CMPCLKCR2	CM Peripheral clock gating register 2.		Go
10h	CMSOFTPRESET0	CM Software Peripheral Reset register 0		Go
12h	CMSOFTPRESET1	CM Software Peripheral Reset register 1		Go
14h	CMSOFTPRESET2	CM Software Peripheral Reset register 2		Go
20h	CMCLKSTOPREQ0	Peripheral Clock Stop Request Register 0		Go
22h	CMCLKSTOPREQ1	Peripheral Clock Stop Request Register 1		Go
24h	CMCLKSTOPREQ2	Peripheral Clock Stop Request Register 2		Go
30h	CMCLKSTOPACK0	Peripheral Clock Stop Ackonwledge Register 0		Go
32h	CMCLKSTOPACK1	Peripheral Clock Stop Ackonwledge Register 1		Go
34h	CMCLKSTOPACK2	Peripheral Clock Stop Ackonwledge Register 2		Go
70h	MCANWAKESTATUS	MCAN Wake Status Register		Go
72h	MCANWAKESTATUSCLR	MCAN Wake Status Clear Register		Go
FAh	PALLOCATESTS	Status of PALLOCATE register.		Go
FCh	CMRESCCLR	CM Reset Cause Status Clear Register		Go
FEh	CMRESC	CM Reset Cause Status Register		Go
100h	CMSYSCTLLOCK	Locks the configuration registers of CM System control		Go

Complex bit access types are encoded to fit into small table cells. Table 41-64 shows the codes that are used for access types in this section.

Table 41-64. CMSYSCTL_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 41-64. CMSYSCTL_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

41.12.5.1 CMPCLKCR0 Register (Offset = 0h) [Reset = 0h]

CMPCLKCR0 is shown in Figure 41-62 and described in Table 41-65.

Return to the Summary Table.

CM Peripheral clock gating register 0.

Figure 41-62. CMPCLKCR0 Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED			USB	RESERVED			I2C0
R-0h			R/W-0h	R-0h			R/W-0h
7	6	5	4	3	2	1	0
RESERVED			SSI0	RESERVED			UART0
R-0h			R/W-0h	R-0h			R/W-0h

Table 41-65. CMPCLKCR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to any of the bits in this register will succeed only if a value of 0x5634 is written to the KEY field. Reset type: CM.RESETn
15-13	RESERVED	R	0h	Reserved
12	USB	R/W	0h	USB Clock gating Bit 0: Clock to USB is turned off 1: Clock to USB is turned on Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
11-9	RESERVED	R	0h	Reserved
8	I2C0	R/W	0h	I2C0 Clock gating Bit 0: Clock to I2C0 is turned off 1: Clock to I2C0 is turned on Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
7-5	RESERVED	R	0h	Reserved
4	SSI0	R/W	0h	SSI0 Clock gating Bit 0: Clock to SSI0 is turned off 1: Clock to SSI0 is turned on Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
3-1	RESERVED	R	0h	Reserved
0	UART0	R/W	0h	UART0 Clock gating Bit 0: Clock to UART0 is turned off 1: Clock to UART0 is turned on Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn

41.12.5.2 CMPCLKCR1 Register (Offset = 2h) [Reset = 0h]

CMPCLKCR1 is shown in Figure 41-63 and described in Table 41-66.

Return to the Summary Table.

CM Peripheral clock gating register 1.

Figure 41-63. CMPCLKCR1 Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED			RESERVED				MCAN_A
R-0h			R-0h				R/W-0h
7	6	5	4	3	2	1	0
RESERVED		CAN_B	CAN_A	RESERVED	ETHERCAT	RESERVED	ETHERNET
R-0h		R/W-0h	R/W-0h	R-0h	R/W-0h	R-0h	R/W-0h

Table 41-66. CMPCLKCR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to any of the bits in this register will succeed only if a value of 0x5634 is written to the KEY field. Reset type: CM.RESETn
15-13	RESERVED	R	0h	Reserved
12-9	RESERVED	R	0h	Reserved
8	MCAN_A	R/W	0h	MCAN_A Clock gating Bit 0: Clock to MCAN_A is turned off 1: Clock to MCAN_A is turned on Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
7-6	RESERVED	R	0h	Reserved
5	CAN_B	R/W	0h	CAN_B Clock gating Bit 0: Clock to CAN_B is turned off 1: Clock to CAN_B is turned on Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
4	CAN_A	R/W	0h	CAN_A Clock gating Bit 0: Clock to CAN_A is turned off 1: Clock to CAN_A is turned on Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
3	RESERVED	R	0h	Reserved
2	ETHERCAT	R/W	0h	ETHERCAT Clock gating Bit 0: Clock to ETHERCAT is turned off 1: Clock to ETHERCAT is turned on Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
1	RESERVED	R	0h	Reserved

Table 41-66. CMPCLKCR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	ETHERNET	R/W	0h	ETHERNET Clock gating Bit 0: Clock to ETHERNET is turned off 1: Clock to ETHERNET is turned on Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn

41.12.5.3 CMPCLKCR2 Register (Offset = 4h) [Reset = 0h]

CMPCLKCR2 is shown in Figure 41-64 and described in Table 41-67.

Return to the Summary Table.

CM Peripheral clock gating register 2.

Figure 41-64. CMPCLKCR2 Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED			RESERVED				GCRC
R-0h			R-0h				R/W-0h
7	6	5	4	3	2	1	0
RESERVED	AESIP	RESERVED	UDMA	RESERVED	CPUTIMER2	CPUTIMER1	CPUTIMER0
R-0h	R/W-0h	R-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h

Table 41-67. CMPCLKCR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to any of the bits in this register will succeed only if a value of 0x5634 is written to the KEY field. Reset type: CM.RESETn
15-13	RESERVED	R	0h	Reserved
12-9	RESERVED	R	0h	Reserved
8	GCRC	R/W	0h	GCRC Clock gating Bit 0: Clock to GCRC is turned off 1: Clock to GCRC is turned on Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
7	RESERVED	R	0h	Reserved
6	AESIP	R/W	0h	AESIP Clock gating Bit 0: Clock to AESIP is turned off 1: Clock to AESIP is turned on Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
5	RESERVED	R	0h	Reserved
4	UDMA	R/W	0h	UDMA Clock gating Bit 0: Clock to UDMA is turned off 1: Clock to UDMA is turned on Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
3	RESERVED	R	0h	Reserved
2	CPUTIMER2	R/W	0h	CPUTIMER2 Clock gating Bit 0: Clock to CPUTIMER2 is turned off 1: Clock to CPUTIMER2 is turned on Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn

Table 41-67. CMPCLKCR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	CPUTIMER1	R/W	0h	CPUTIMER1 Clock gating Bit 0: Clock to CPUTIMER1 is turned off 1: Clock to CPUTIMER1 is turned on Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
0	CPUTIMER0	R/W	0h	CPUTIMER0 Clock gating Bit 0: Clock to CPUTIMER0 is turned off 1: Clock to CPUTIMER0 is turned on Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn

41.12.5.4 CMSOFTPRESET0 Register (Offset = 10h) [Reset = 0h]

CMSOFTPRESET0 is shown in Figure 41-65 and described in Table 41-68.

Return to the Summary Table.

CM Software Peripheral Reset register 0

Figure 41-65. CMSOFTPRESET0 Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED			USB	RESERVED			I2C0
R-0h			R/W-0h	R-0h			R/W-0h
7	6	5	4	3	2	1	0
RESERVED			SSI0	RESERVED			UART0
R-0h			R/W-0h	R-0h			R/W-0h

Table 41-68. CMSOFTPRESET0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to any of the bits in this register will succeed only if a value of 0x5634 is written to the KEY field. Reset type: CM.RESETn
15-13	RESERVED	R	0h	Reserved
12	USB	R/W	0h	USB Soft Reset Bit 1: Reset to USB is asserted 0: Reset to USB is released Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
11-9	RESERVED	R	0h	Reserved
8	I2C0	R/W	0h	I2C0 Soft Reset Bit 1: Reset to I2C0 is asserted 0: Reset to I2C0 is released Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
7-5	RESERVED	R	0h	Reserved
4	SSI0	R/W	0h	SSI0 Soft Reset Bit 1: Reset to SSI0 is asserted 0: Reset to SSI0 is released Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
3-1	RESERVED	R	0h	Reserved
0	UART0	R/W	0h	UART0 Soft Reset Bit 1: Reset to UART0 is asserted 0: Reset to UART0 is released Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn

41.12.5.5 CMSOFTPRESET1 Register (Offset = 12h) [Reset = 5h]

CMSOFTPRESET1 is shown in Figure 41-66 and described in Table 41-69.

Return to the Summary Table.

CM Software Peripheral Reset register 1

Figure 41-66. CMSOFTPRESET1 Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED				RESERVED			MCAN_A
R-0h				R-0h			R/W-0h
7	6	5	4	3	2	1	0
RESERVED		CAN_B	CAN_A	RESERVED	ETHERCAT	RESERVED	ETHERNET
R-0h		R/W-0h	R/W-0h	R-0h	R/W-1h	R-0h	R/W-1h

Table 41-69. CMSOFTPRESET1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to any of the bits in this register will succeed only if a value of 0x5634 is written to the KEY field. Reset type: CM.RESETn
15-12	RESERVED	R	0h	Reserved
11-9	RESERVED	R	0h	Reserved
8	MCAN_A	R/W	0h	MCAN_A Soft Reset Bit 1: Reset to MCAN_A is asserted 0: Reset to MCAN_A is released Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
7-6	RESERVED	R	0h	Reserved
5	CAN_B	R/W	0h	CAN_B Soft Reset Bit 1: Reset to CAN_B is asserted 0: Reset to CAN_B is released Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
4	CAN_A	R/W	0h	CAN_A Soft Reset Bit 1: Reset to CAN_A is asserted 0: Reset to CAN_A is released Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
3	RESERVED	R	0h	Reserved
2	ETHERCAT	R/W	1h	ETHERCAT Soft Reset Bit 1: Reset to ETHERCAT is asserted 0: Reset to ETHERCAT is released Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
1	RESERVED	R	0h	Reserved

Table 41-69. CMSOFTPRESET1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	ETHERNET	R/W	1h	ETHERNET Soft Reset Bit 1: Reset to ETHERNET is asserted 0: Reset to ETHERNET is released Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn

41.12.5.6 CMSOFTPRESET2 Register (Offset = 14h) [Reset = 0h]

CMSOFTPRESET2 is shown in Figure 41-67 and described in Table 41-70.

Return to the Summary Table.

CM Software Peripheral Reset register 2

Figure 41-67. CMSOFTPRESET2 Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED			RESERVED				GCRC
R-0h			R-0h				R/W-0h
7	6	5	4	3	2	1	0
RESERVED	AESIP	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0h	R/W-0h	R-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h

Table 41-70. CMSOFTPRESET2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to any of the bits in this register will succeed only if a value of 0x5634 is written to the KEY field. Reset type: CM.RESETn
15-13	RESERVED	R	0h	Reserved
12-9	RESERVED	R	0h	Reserved
8	GCRC	R/W	0h	GCRC Soft Reset Bit 1: Reset to GCRC is asserted 0: Reset to GCRC is released Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
7	RESERVED	R	0h	Reserved
6	AESIP	R/W	0h	AESIP Soft Reset Bit 1: Reset to AESIP is asserted 0: Reset to AESIP is released Note: Write to this bit will succeed only if a matching key value is written to the KEY field of this register Reset type: CM.RESETn
5	RESERVED	R	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	RESERVED	R/W	0h	Reserved

41.12.5.7 CMCLKSTOPREQ0 Register (Offset = 20h) [Reset = 0h]

CMCLKSTOPREQ0 is shown in Figure 41-68 and described in Table 41-71.

Return to the Summary Table.

Peripheral Clock Stop Request Register 0

Figure 41-68. CMCLKSTOPREQ0 Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED			RESERVED	RESERVED			RESERVED
R-0h			R/W-0h	R-0h			R/W-0h
7	6	5	4	3	2	1	0
RESERVED			RESERVED	RESERVED			RESERVED
R-0h			R/W-0h	R-0h			R/W-0h

Table 41-71. CMCLKSTOPREQ0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to any of the bits in this register will succeed only if a value of 0x5634 is written to the KEY field. Reset type: CM.RESETn
15-13	RESERVED	R	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11-9	RESERVED	R	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7-5	RESERVED	R	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3-1	RESERVED	R	0h	Reserved
0	RESERVED	R/W	0h	Reserved

41.12.5.8 CMCLKSTOPREQ1 Register (Offset = 22h) [Reset = 0h]

CMCLKSTOPREQ1 is shown in Figure 41-69 and described in Table 41-72.

Return to the Summary Table.

Peripheral Clock Stop Request Register 1

Figure 41-69. CMCLKSTOPREQ1 Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED				RESERVED			MCAN_A
R-0h				R-0h			R/W-0h
7	6	5	4	3	2	1	0
RESERVED		RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0h		R/W-0h	R/W-0h	R-0h	R/W-0h	R-0h	R/W-0h

Table 41-72. CMCLKSTOPREQ1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to any of the bits in this register will succeed only if a value of 0x5634 is written to the KEY field. Reset type: CM.RESETn
15-12	RESERVED	R	0h	Reserved
11-9	RESERVED	R	0h	Reserved
8	MCAN_A	R/W	0h	MCAN_A Clock Stop Request Bit 0: If clock to MCAN_A is turned off, it will be turned on, else no effect. 1: Clock stop request to MCAN_A Note: Once set, this bit is cleared when clock to MCAN_A is turned on as a result of a wakeup event in hardware Reset type: CM.RESETn
7-6	RESERVED	R	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	RESERVED	R	0h	Reserved
0	RESERVED	R/W	0h	Reserved

41.12.5.9 CMCLKSTOPREQ2 Register (Offset = 24h) [Reset = 0h]

CMCLKSTOPREQ2 is shown in Figure 41-70 and described in Table 41-73.

Return to the Summary Table.

Peripheral Clock Stop Request Register 2

Figure 41-70. CMCLKSTOPREQ2 Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED			RESERVED				RESERVED
R-0h			R-0h				R/W-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0h	R/W-0h	R-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h

Table 41-73. CMCLKSTOPREQ2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to any of the bits in this register will succeed only if a value of 0x5634 is written to the KEY field. Reset type: CM.RESETn
15-13	RESERVED	R	0h	Reserved
12-9	RESERVED	R	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7	RESERVED	R	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	RESERVED	R/W	0h	Reserved

41.12.5.10 CMCLKSTOPACK0 Register (Offset = 30h) [Reset = 0h]

CMCLKSTOPACK0 is shown in Figure 41-71 and described in Table 41-74.

Return to the Summary Table.

Peripheral Clock Stop Ackonwledge Register 0

Figure 41-71. CMCLKSTOPACK0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			RESERVED	RESERVED			RESERVED
R-0h			R-0h	R-0h			R-0h
7	6	5	4	3	2	1	0
RESERVED			RESERVED	RESERVED			RESERVED
R-0h			R-0h	R-0h			R-0h

Table 41-74. CMCLKSTOPACK0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-13	RESERVED	R	0h	Reserved
12	RESERVED	R	0h	Reserved
11-9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7-5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3-1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

41.12.5.11 CMCLKSTOPACK1 Register (Offset = 32h) [Reset = 0h]

CMCLKSTOPACK1 is shown in Figure 41-72 and described in Table 41-75.

Return to the Summary Table.

Peripheral Clock Stop Ackonwledge Register 1

Figure 41-72. CMCLKSTOPACK1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							MCAN_A
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED		RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 41-75. CMCLKSTOPACK1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	MCAN_A	R	0h	MCAN_A Clock Stop Acknowledge Bit 0: Clock stop request not acknowledged 1: Clock stop acknowledged Reset type: CM.RESETn
7-6	RESERVED	R	0h	Reserved
5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

41.12.5.12 CMCLKSTOPACK2 Register (Offset = 34h) [Reset = 0h]

CMCLKSTOPACK2 is shown in Figure 41-73 and described in Table 41-76.

Return to the Summary Table.

Peripheral Clock Stop Ackonwledge Register 2

Figure 41-73. CMCLKSTOPACK2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			RESERVED				RESERVED
R-0h			R-0h				R-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 41-76. CMCLKSTOPACK2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-13	RESERVED	R	0h	Reserved
12-9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	RESERVED	R	0h	Reserved
6	RESERVED	R	0h	Reserved
5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

41.12.5.13 MCANWAKESTATUS Register (Offset = 70h) [Reset = 0h]

MCANWAKESTATUS is shown in Figure 41-74 and described in Table 41-77.

Return to the Summary Table.

MCAN Wake Status Register

Figure 41-74. MCANWAKESTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							WAKE
R-0h							R-0h

Table 41-77. MCANWAKESTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	WAKE	R	0h	0 : wakeup event has not occured. 1 : wakeup event has occured. Reset type: CM.RESETn

41.12.5.14 MCANWAKESTATUSCLR Register (Offset = 72h) [Reset = 0h]

MCANWAKESTATUSCLR is shown in Figure 41-75 and described in Table 41-78.

Return to the Summary Table.

MCAN Wake Status Clear Register

Figure 41-75. MCANWAKESTATUSCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							WAKE
R-0h							R-0/W1S-0h

Table 41-78. MCANWAKESTATUSCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	WAKE	R-0/W1S	0h	0 : No effect. 1 : Clears WAKE bit of MCANWAKESTATUS register Reset type: CM.RESETn

41.12.5.15 PALLOCATESTS Register (Offset = FAh) [Reset = X]

PALLOCATESTS is shown in Figure 41-76 and described in Table 41-79.

Return to the Summary Table.

Status of PALLOCATE register.

Figure 41-76. PALLOCATESTS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			MCAN_A	CAN_B	CAN_A	ETHERCAT	USB
R-0h			R-X	R-X	R-X	R-X	R-X

Table 41-79. PALLOCATESTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4	MCAN_A	R	X	Status of PALLOCATE.MCAN_A bit Reset type: CM.RESETn
3	CAN_B	R	X	Status of PALLOCATE.CAN_B bit Reset type: CM.RESETn
2	CAN_A	R	X	Status of PALLOCATE.CAN_A bit Reset type: CM.RESETn
1	ETHERCAT	R	X	Status of PALLOCATE.ETHERCAT bit Reset type: CM.RESETn
0	USB	R	X	Status of PALLOCATE.USB bit Reset type: CM.RESETn

41.12.5.16 CMRESCCLR Register (Offset = FCh) [Reset = 0h]

CMRESCCLR is shown in Figure 41-77 and described in Table 41-80.

Return to the Summary Table.

CM Reset Cause Status Clear Register

Figure 41-77. CMRESCCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				CMEOLRESETn	CMNMIWDRSTn	CMSYSRESETREQ	CMVECTRESETn
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
RESERVED						CPU1_SIMRESET_XRSn	CMRSTCTLRESETREQ
R-0h						R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
CPU1SIMRESET_CPURSn	ECAT_RESET_OUT	CPU1SCCRESETn	CPU1SYSRSN	CPU1NMIWDRSn	CPU1WDRSn	XRSn	PORESETn
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 41-80. CMRESCCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19	CMEOLRESETn	R-0/W1S	0h	0: No effect. 1:Clear CMEOLRESETn flag in CMRESC register. Reset type: CM.RESETn
18	CMNMIWDRSTn	R-0/W1S	0h	0: No effect. 1:Clear CMNMIWDRSTn flag in CMRESC register. Reset type: CM.RESETn
17	CMSYSRESETREQ	R-0/W1S	0h	0: No effect. 1:Clear CMSYSRESETREQ flag in CMRESC register. Reset type: CM.RESETn
16	CMVECTRESETn	R-0/W1S	0h	0: No effect. 1:Clear CMVECTRESETn flag in CMRESC register. Reset type: CM.RESETn
15-10	RESERVED	R	0h	Reserved
9	CPU1_SIMRESET_XRSn	R-0/W1S	0h	0: No effect. 1:Clear CPU1_SIMRESET_XRSn flag in CMRESC register. Reset type: CM.RESETn
8	CMRSTCTLRESETREQ	R-0/W1S	0h	0: No effect. 1:Clear CMRSTCTLRESETREQ flag in CMRESC register. Reset type: CM.RESETn
7	CPU1SIMRESET_CPURSn	R-0/W1S	0h	0: No effect. 1:Clear CPU1SIMRESET_CPURSn flag in CMRESC register. Reset type: CM.RESETn
6	ECAT_RESET_OUT	R-0/W1S	0h	0: No effect. 1:Clear ECAT_RESET_OUT flag in CMRESC register. Reset type: CM.RESETn
5	CPU1SCCRESETn	R-0/W1S	0h	0: No effect. 1:Clear CPU1SCCRESETn flag in CMRESC register. Reset type: CM.RESETn

Table 41-80. CMRESCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	CPU1SYSRSN	R-0/W1S	0h	0: No effect. 1:Clear CPU1SYSRSN flag in CMRESC register. Reset type: CM.RESETn
3	CPU1NMIWDRSn	R-0/W1S	0h	0: No effect. 1:Clear CPU1NMIWDRSn flag in CMRESC register. Reset type: CM.RESETn
2	CPU1WDRSn	R-0/W1S	0h	0: No effect. 1:Clear CPU1WDRSn flag in CMRESC register. Reset type: CM.RESETn
1	XRSn	R-0/W1S	0h	0: No effect. 1:Clear XRSn flag in CMRESC register. Reset type: CM.RESETn
0	PORESETn	R-0/W1S	0h	0: No effect. 1:Clear PORESETn flag in CMRESC register. Reset type: CM.RESETn

41.12.5.17 CMRESC Register (Offset = FEh) [Reset = 3h]

CMRESC is shown in Figure 41-78 and described in Table 41-81.

Return to the Summary Table.

CM Reset Cause Status Register

Figure 41-78. CMRESC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				CMEOLRESETn	CMNMIWDRSTn	CMSYSRESETREQ	CMVECTRESETn
R-0h				R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED						CPU1_SIMRESET_XRSn	CMRSTCTL_RESETREQ
R-0h						R-0h	R-0h
7	6	5	4	3	2	1	0
CPU1_SIMRESET_CPURSn	ECAT_RESET_OUT	CPU1_SCCRESETn	CPU1_SYSRSN	CPU1_NMIWDRSn	CPU1_WDRSn	XRSn	PORESETn
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-1h	R-1h

Table 41-81. CMRESC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19	CMEOLRESETn	R	0h	0: Reset of CM4 not due to CMEOLRESETn. 1:Reset of CM4 due to CMEOLRESETn. Reset type: PORESETn
18	CMNMIWDRSTn	R	0h	0: Reset of CM4 not due to CMNMIWDRSTn. 1:Reset of CM4 due to CMNMIWDRSTn. Reset type: PORESETn
17	CMSYSRESETREQ	R	0h	0: Reset of CM4 not due to CMSYSRESETREQ. 1:Reset of CM4 due to CMSYSRESETREQ. Reset type: PORESETn
16	CMVECTRESETn	R	0h	0: Reset of CM4 not due to CMVECTRESETn. 1:Reset of CM4 due to CMVECTRESETn. Reset type: PORESETn
15-10	RESERVED	R	0h	Reserved
9	CPU1_SIMRESET_XRSn	R	0h	0: Reset of CM4 not due to CPU1_SIMRESET_XRSn. 1:Reset of CM4 due to CPU1_SIMRESET_XRSn. Reset type: PORESETn
8	CMRSTCTL_RESETREQ	R	0h	0: Reset of CM4 not due to CMRSTCTL_RESETREQ. 1:Reset of CM4 due to CMRSTCTL_RESETREQ. Reset type: PORESETn
7	CPU1_SIMRESET_CPURSn	R	0h	0: Reset of CM4 not due to CPU1_SIMRESET_CPURSn. 1:Reset of CM4 due to CPU1_SIMRESET_CPURSn. Reset type: PORESETn
6	ECAT_RESET_OUT	R	0h	0: Reset of CM4 not due to ECAT_RESET_OUT. 1:Reset of CM4 due to ECAT_RESET_OUT. Reset type: PORESETn
5	CPU1_SCCRESETn	R	0h	0: Reset of CM4 not due to CPU1_SCCRESETn. 1:Reset of CM4 due to CPU1_SCCRESETn. Reset type: PORESETn

Table 41-81. CMRESC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	CPU1_SYSRSN	R	0h	0: Reset of CM4 not due to CPU1_SYSRSN. 1:Reset of CM4 due to CPU1_SYSRSN. Reset type: PORESETn
3	CPU1_NMIWDRSn	R	0h	0: Reset of CM4 not due to CPU1_NMIWDRSn. 1:Reset of CM4 due to CPU1_NMIWDRSn. Reset type: PORESETn
2	CPU1_WDRSn	R	0h	0: Reset of CM4 not due to CPU1_WDRSn. 1:Reset of CM4 due to CPU1_WDRSn. Reset type: PORESETn
1	XRSn	R	1h	0: Reset of CM4 not due to XRSn. 1:Reset of CM4 due to XRSn. Reset type: PORESETn
0	PORESETn	R	1h	0: Reset of CM4 not due to PORESETn. 1:Reset of CM4 due to PORESETn. Reset type: PORESETn

41.12.5.18 CMSYSCTLLOCK Register (Offset = 100h) [Reset = 0h]

CMSYSCTLLOCK is shown in Figure 41-79 and described in Table 41-82.

Return to the Summary Table.

Locks the configuration registers of CM System control

Figure 41-79. CMSYSCTLLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK
R-0h							R/WSonce-0h

Table 41-82. CMSYSCTLLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	LOCK	R/WSonce	0h	0: Writes to CMECATCTL are not blocked 1: Writes to CMECATCTL are blocked Reset type: CM.RESETn

41.12.6 CM_CPUTIMER_REGS Registers

Table 41-83 lists the memory-mapped registers for the CM_CPUTIMER_REGS registers. All register offset addresses not listed in Table 41-83 should be considered as reserved locations and the register contents should not be modified.

Table 41-83. CM_CPUTIMER_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	TIM	Timer counter register	EALLOW	Go
4h	PRD	Timer period register	EALLOW	Go
8h	TCR	Timer control register	EALLOW	Go
Ch	TPR	Timer prescaler register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 41-84 shows the codes that are used for access types in this section.

Table 41-84. CM_CPUTIMER_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

41.12.6.1 TIM Register (Offset = 0h) [Reset = FFFFh]

TIM is shown in Figure 41-80 and described in Table 41-85.

Return to the Summary Table.

System Control & Status Register

Figure 41-80. TIM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT																															
R/W-FFFFh																															

Table 41-85. TIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNT	R/W	FFFFh	Timer Counter Registers (TIM): The TIM register holds the 32-bit count of the timer. The TIM register decrements by one every (TDDRH:TDDRL+1) clock cycles, where TDDRH:TDDRL is the timer pre-scale divide-down value. When the TIM decrements to zero, the TIM register is re-loaded with the period value contained in the PRD register, the timer the timer interrupt (TINTn) signal is pulsed. Reset type: CM.RESETn

41.12.6.2 PRD Register (Offset = 4h) [Reset = FFFFh]

PRD is shown in Figure 41-81 and described in Table 41-86.

Return to the Summary Table.

Timer period register

Figure 41-81. PRD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRD																															
R/W-FFFFh																															

Table 41-86. PRD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PRD	R/W	FFFFh	Timer Period Register (PRD): The PRD register holds the 32-bit period value. When the TIM decrements to zero, the TIM register is re-loaded with the period value contained in the PRD register, at the start of the next timer input clock cycle. The PRD contents are also loaded into the TIM when you set the timer reload bit (TRB) in the Timer Control Register (TCR). Reset type: CM.RESETn

41.12.6.3 TCR Register (Offset = 8h) [Reset = X]

TCR is shown in Figure 41-82 and described in Table 41-87.

Return to the Summary Table.

Timer control register

Figure 41-82. TCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
TIF	TIE	RESERVED		FREE	SOFT	RESERVED	
R/W1S-0h	R/W-0h	R-0h		R/W-0h	R/W-0h	R-0h	
7	6	5	4	3	2	1	0
RESERVED		TRB	TSS	RESERVED			
R-0h		R-0/W1S-0h	R/W-0h	R-0h			

Table 41-87. TCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	TIF	R/W1S	0h	Timer Interrupt Flag. This flag gets set when the timer decrements to zero. This bit can be cleared by software writing a 1, but it can only be set by the timer reaching zero. Writing a 1 to this bit will clear it, writing a zero has no effect. Reset type: CM.RESETn
14	TIE	R/W	0h	Timer Interrupt Enable. If the timer decrements to zero, and this bit is set, the timer will assert it's interrupt request. Reset type: CM.RESETn
13-12	RESERVED	R	0h	Reserved
11	FREE	R/W	0h	If this bit is set then on a debug halt of CM4, timer would continute to decrement and generate periodic interrupts. If this bit is 0, counter behavior under debug halt condition is determined by TCR.SOFT bit. Reset type: CM.RESETn
10	SOFT	R/W	0h	If the TCR.FREE bit is 0 and SOFT bit is: 0 : On a debug halt of CM4, timer will stop counting immediately (Hard Stop) 1 : On a debug halt of CM4, timer will stop counting upon reaching 0 and generating the interrupt (Soft Stop) If the TCR.FREE bit is 1 then SOFT bit has no impact in timer behavior. Reset type: CM.RESETn
9-6	RESERVED	R	0h	Reserved
5	TRB	R-0/W1S	0h	Timer Reload bit. When a 1 is written to TRB, the TIM is loaded with the value in the PRD, and the pre-scaler counter (PSC) is loaded with the value in the timer divide-down register (TDDR). The TRB bit is always read as zero Reset type: CM.RESETn

Table 41-87. TCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	TSS	R/W	0h	Timer stop status bit. TSS is a 1-bit flag that stops or starts the timer. To stop the timer, set TSS to 1. To start or restart the timer, set TSS to 0. At reset, TSS is cleared to 0 and the timer immediately starts. Reset type: CM.RESETn
3-0	RESERVED	R	0h	Reserved

41.12.6.4 TPR Register (Offset = Ch) [Reset = 0h]

TPR is shown in Figure 41-83 and described in Table 41-88.

Return to the Summary Table.

Timer prescaler register

Figure 41-83. TPR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSCH								TDDRH								PSCL								TDDRL							
R-0h								R/W-0h								R-0h								R/W-0h							

Table 41-88. TPR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	PSCH	R	0h	See description of PSCL above Reset type: CM.RESETn
23-16	TDDRH	R/W	0h	See description of TDDRL above Reset type: CM.RESETn
15-8	PSCL	R	0h	Timer Pre-Scale Counter. These bits hold the current pre-scale count for the timer. For every timer clock source cycle that the PSCH:PSCL value is greater than 0, the PSCH:PSC decrements by one. One timer clock (output of the timer pre-scaler) cycle after the PSCH:PSC reaches 0, the PSCH:PSCL is loaded with the contents of the TDDRH:TDDR, and the timer counter register (TIMH:TIM) decrements by one. The PSCH:PSC is also reloaded whenever the timer reload bit (TRB) is set by software. The PSCH:PSCL can be checked by reading the register, but it cannot be set directly. It must get its value from the timer divide-down register (TDDRH:TDDR). At reset, the PSCH:PSCL is set to 0. Reset type: CM.RESETn
7-0	TDDRL	R/W	0h	Timer Divide-Down. Every (TDDRH:TDDRL + 1) timer clock source cycles, the timer counter register (TIM) decrements by one. At reset, the TDDRH:TDDRL bits are cleared to 0. If you want to increase the overall timer count by an integer factor, write this factor minus one to the TDDRH:TDDRL bits. When the prescaler counter (PSCH:PSCL) value is 0, one timer clock source cycle later, the contents of the TDDRH:TDDRL reload the PSCH:PSC, and the TIMH:TIM decrements by one. TDDRH:TDDRL also reloads the PSCH:PSC whenever the timer reload bit (TRB) is set by software. Reset type: CM.RESETn

41.12.7 MPU_REGS Registers

Table 41-89 lists the memory-mapped registers for the MPU_REGS registers. All register offset addresses not listed in Table 41-89 should be considered as reserved locations and the register contents should not be modified.

Table 41-89. MPU_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	MPU_CONTROL_REG	MPU control regster	Privilege mode	Go
20h	ACC_VIO_INTEN	Access violation interrupt enable	Key based protection	Go
24h	ACC_VIO_FLAGS	Access violation flag register		Go
28h	ACC_VIO_FLAGS_SET	Acesss violation set register	Key based protection	Go
2Ch	ACC_VIO_FLAGS_CLR	Access violation clear register	Key based protection	Go
30h	ACC_VIO_ADDR_REG	Access violation address register		Go
40h	REGION0_STARTADDRESSS	Region 0 start address register	Privilege mode	Go
44h	REGION0_CONFIG	Region 0 configuration register	Privilege mode	Go
48h	REGION1_STARTADDRESSS	Region 1 start address register	Privilege mode	Go
4Ch	REGION1_CONFIG	Region 1 configuration register	Privilege mode	Go
50h	REGION2_STARTADDRESSS	Region 2 start address register	Privilege mode	Go
54h	REGION2_CONFIG	Region 2 configuration register	Privilege mode	Go
58h	REGION3_STARTADDRESSS	Region 3 start address register	Privilege mode	Go
5Ch	REGION3_CONFIG	Region 3 configuration register	Privilege mode	Go
60h	REGION4_STARTADDRESSS	Region 4 start address register	Privilege mode	Go
64h	REGION4_CONFIG	Region 4 configuration register	Privilege mode	Go
68h	REGION5_STARTADDRESSS	Region 5 start address register	Privilege mode	Go
6Ch	REGION5_CONFIG	Region 5 configuration register	Privilege mode	Go
70h	REGION6_STARTADDRESSS	Region 6 start address register	Privilege mode	Go
74h	REGION6_CONFIG	Region 6 configuration register	Privilege mode	Go
78h	REGION7_STARTADDRESSS	Region 7 start address register	Privilege mode	Go
7Ch	REGION7_CONFIG	Region 7 configuration register	Privilege mode	Go

Complex bit access types are encoded to fit into small table cells. Table 41-90 shows the codes that are used for access types in this section.

Table 41-90. MPU_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 41-90. MPU_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

41.12.7.1 MPU_CONTROL_REG Register (Offset = 0h) [Reset = 0h]

MPU_CONTROL_REG is shown in Figure 41-84 and described in Table 41-91.

Return to the Summary Table.

MPU control regster

Figure 41-84. MPU_CONTROL_REG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ENABLE
R-0h							R/W-0h

Table 41-91. MPU_CONTROL_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	ENABLE	R/W	0h	Memory Protection unit enable 0: MPU disabled, Entire memory region is accessible by Bus Master 1: MPU enabled, Selected regions are accessible by Bus Master Reset type: CM.RESETn

41.12.7.2 ACC_VIO_INTEN Register (Offset = 20h) [Reset = 0h]

ACC_VIO_INTEN is shown in Figure 41-85 and described in Table 41-92.

Return to the Summary Table.

Access violation interrupt enable

Figure 41-85. ACC_VIO_INTEN Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INTEN
R-0h							R/W-0h

Table 41-92. ACC_VIO_INTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to INTEN is allowed only if [31:16] is written with Data 0xA5A5 Reset type: CM.RESETn
15-1	RESERVED	R	0h	Reserved
0	INTEN	R/W	0h	Access violation Interrupt enable 0: Interrupt disabled 1: Interrupt enabled Reset type: CM.RESETn

41.12.7.3 ACC_VIO_FLAGS Register (Offset = 24h) [Reset = 0h]

ACC_VIO_FLAGS is shown in Figure 41-86 and described in Table 41-93.

Return to the Summary Table.

Access violation flag register

Figure 41-86. ACC_VIO_FLAGS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														WR	RD
R-0h														R-0h	R-0h

Table 41-93. ACC_VIO_FLAGS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-2	RESERVED	R	0h	Reserved
1	WR	R	0h	Gets set when write access violation is detected Reset type: CM.RESETn
0	RD	R	0h	Gets set when read access violation is detected Reset type: CM.RESETn

41.12.7.4 ACC_VIO_FLAGS_SET Register (Offset = 28h) [Reset = 0h]

ACC_VIO_FLAGS_SET is shown in Figure 41-87 and described in Table 41-94.

Return to the Summary Table.

Acesss violation set register

Figure 41-87. ACC_VIO_FLAGS_SET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
R-0/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														WR	RD
R-0h														R-0/W1S-0h	R-0/W1S-0h

Table 41-94. ACC_VIO_FLAGS_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to SET bits will take effect only if [31:16] is written with Data 0xA5A5 Reset type: CM.RESETn
15-2	RESERVED	R	0h	Reserved
1	WR	R-0/W1S	0h	0: None. 1: sets ACC_VIO_FLAGS[WR] Flag Reset type: CM.RESETn
0	RD	R-0/W1S	0h	0: None. 1: sets ACC_VIO_FLAGS[RD] Flag Reset type: CM.RESETn

41.12.7.5 ACC_VIO_FLAGS_CLR Register (Offset = 2Ch) [Reset = 0h]

ACC_VIO_FLAGS_CLR is shown in Figure 41-88 and described in Table 41-95.

Return to the Summary Table.

Access violation clear register

Figure 41-88. ACC_VIO_FLAGS_CLR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
R-0/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														WR	RD
R-0h														R-0/W1S-0h	R-0/W1S-0h

Table 41-95. ACC_VIO_FLAGS_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to CLR bits will take effect only if [31:16] is written with Data 0xA5A5 Reset type: CM.RESETn
15-2	RESERVED	R	0h	Reserved
1	WR	R-0/W1S	0h	0: None. 1: Clears ACC_VIO_FLAGS[WR] Flag Reset type: CM.RESETn
0	RD	R-0/W1S	0h	0: None. 1: Clears ACC_VIO_FLAGS[RD] Flag Reset type: CM.RESETn

41.12.7.6 ACC_VIO_ADDR_REG Register (Offset = 30h) [Reset = 0h]

ACC_VIO_ADDR_REG is shown in Figure 41-89 and described in Table 41-96.

Return to the Summary Table.

Access violation address register

Figure 41-89. ACC_VIO_ADDR_REG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VIOLATION_ADDRESS																															
R-0h																															

Table 41-96. ACC_VIO_ADDR_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VIOLATION_ADDRESS	R	0h	Capture the First access violation address. Address of subsequent violations is not captured until set flag in ACC_VIO_FLAGS is cleared. Read or write violation can be determined by reading ACC_VIO_FLAGS. ACC_VIO_FLAGS is one hot, only 1 flag can be set at a time. Reset type: CM.RESETn

41.12.7.7 REGION0_STARTADDRESSS Register (Offset = 40h) [Reset = 0h]

REGION0_STARTADDRESSS is shown in Figure 41-90 and described in Table 41-97.

Return to the Summary Table.

Region 0 start address register

Figure 41-90. REGION0_STARTADDRESSS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
START_ADDR																															
R/W-0h																															

Table 41-97. REGION0_STARTADDRESSS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	START_ADDR	R/W	0h	Start address of region-0. Address written to this register must be aligned to multiples of region size. Address must be >=1KB boundary Reset type: CM.RESETn

41.12.7.8 REGION0_CONFIG Register (Offset = 44h) [Reset = 0h]

REGION0_CONFIG is shown in Figure 41-91 and described in Table 41-98.

Return to the Summary Table.

Region 0 configuration register

Figure 41-91. REGION0_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
SUBREGION7_DISABLE	SUBREGION6_DISABLE	SUBREGION5_DISABLE	SUBREGION4_DISABLE	SUBREGION3_DISABLE	SUBREGION2_DISABLE	SUBREGION1_DISABLE	SUBREGION0_DISABLE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED			SIZE				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
RESERVED		PROT_TYPE		RESERVED			ENABLE
R-0h		R/W-0h		R-0h			R/W-0h

Table 41-98. REGION0_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23	SUBREGION7_DISABLE	R/W	0h	Disable Sub Region-7 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
22	SUBREGION6_DISABLE	R/W	0h	Disable Sub Region-6 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
21	SUBREGION5_DISABLE	R/W	0h	Disable Sub Region-5 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
20	SUBREGION4_DISABLE	R/W	0h	Disable Sub Region-4 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
19	SUBREGION3_DISABLE	R/W	0h	Disable Sub Region-3 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
18	SUBREGION2_DISABLE	R/W	0h	Disable Sub Region-2 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
17	SUBREGION1_DISABLE	R/W	0h	Disable Sub Region-1 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn

Table 41-98. REGION0_CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	SUBREGION0_DISABLE	R/W	0h	Disable Sub Region-0 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
15-13	RESERVED	R	0h	Reserved
12-8	SIZE	R/W	0h	Size of the region Region size = 2^{SIZE}*1KB 0x0 : 1KB 0x1 : 2KB .... 0x16: Covers full 32 bit address range (4GB) 0x17 to 0x1F : Reserved (same as 0x0) Reset type: CM.RESETn
7-6	RESERVED	R	0h	Reserved
5-4	PROT_TYPE	R/W	0h	Defines access permission for region-0 00: Read Only 10: No Access, Read and Write are not allowed 01: Full Access, both Read and Write are allowed 11: Reserved (Read only) Reset type: CM.RESETn
3-1	RESERVED	R	0h	Reserved
0	ENABLE	R/W	0h	Enable Region-0 0: Region Disabled 1: Region Enabled Reset type: CM.RESETn

41.12.7.9 REGION1_STARTADDRESSS Register (Offset = 48h) [Reset = 0h]

REGION1_STARTADDRESSS is shown in Figure 41-92 and described in Table 41-99.

Return to the Summary Table.

Region 1 start address register

Figure 41-92. REGION1_STARTADDRESSS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
START_ADDR																															
R/W-0h																															

Table 41-99. REGION1_STARTADDRESSS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	START_ADDR	R/W	0h	Start address of region-1. Address written to this register must be aligned to multiples of region size. Address must be >=1KB boundary Reset type: CM.RESETn

41.12.7.10 REGION1_CONFIG Register (Offset = 4Ch) [Reset = 0h]

REGION1_CONFIG is shown in Figure 41-93 and described in Table 41-100.

Return to the Summary Table.

Region 1 configuration register

Figure 41-93. REGION1_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
SUBREGION7_DISABLE	SUBREGION6_DISABLE	SUBREGION5_DISABLE	SUBREGION4_DISABLE	SUBREGION3_DISABLE	SUBREGION2_DISABLE	SUBREGION1_DISABLE	SUBREGION0_DISABLE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED			SIZE				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
RESERVED		PROT_TYPE		RESERVED			ENABLE
R-0h		R/W-0h		R-0h			R/W-0h

Table 41-100. REGION1_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23	SUBREGION7_DISABLE	R/W	0h	Disable Sub Region-7 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
22	SUBREGION6_DISABLE	R/W	0h	Disable Sub Region-6 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
21	SUBREGION5_DISABLE	R/W	0h	Disable Sub Region-5 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
20	SUBREGION4_DISABLE	R/W	0h	Disable Sub Region-4 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
19	SUBREGION3_DISABLE	R/W	0h	Disable Sub Region-3 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
18	SUBREGION2_DISABLE	R/W	0h	Disable Sub Region-2 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
17	SUBREGION1_DISABLE	R/W	0h	Disable Sub Region-1 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn

Table 41-100. REGION1_CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	SUBREGION0_DISABLE	R/W	0h	Disable Sub Region-0 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
15-13	RESERVED	R	0h	Reserved
12-8	SIZE	R/W	0h	Size of the region Region size = 2^{SIZE}*1KB 0x0 : 1KB 0x1 : 2KB .... 0x16: Covers full 32 bit address range (4GB) 0x17 to 0x1F : Reserved (same as 0x0) Reset type: CM.RESETn
7-6	RESERVED	R	0h	Reserved
5-4	PROT_TYPE	R/W	0h	Defines access permission for region-1 00: Read Only 10: No Access, Read and Write are not allowed 01: Full Access, both Read and Write are allowed 11: Reserved (Read only) Reset type: CM.RESETn
3-1	RESERVED	R	0h	Reserved
0	ENABLE	R/W	0h	Enable Region-1 0: Region Disabled 1: Region Enabled Reset type: CM.RESETn

41.12.7.11 REGION2_STARTADDRESSS Register (Offset = 50h) [Reset = 0h]

REGION2_STARTADDRESSS is shown in Figure 41-94 and described in Table 41-101.

Return to the Summary Table.

Region 2 start address register

Figure 41-94. REGION2_STARTADDRESSS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
START_ADDR																															
R/W-0h																															

Table 41-101. REGION2_STARTADDRESSS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	START_ADDR	R/W	0h	Start address of region-2. Address written to this register must be aligned to multiples of region size. Address must be >=1KB boundary Reset type: CM.RESETn

41.12.7.12 REGION2_CONFIG Register (Offset = 54h) [Reset = 0h]

REGION2_CONFIG is shown in Figure 41-95 and described in Table 41-102.

Return to the Summary Table.

Region 2 configuration register

Figure 41-95. REGION2_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
SUBREGION7_DISABLE	SUBREGION6_DISABLE	SUBREGION5_DISABLE	SUBREGION4_DISABLE	SUBREGION3_DISABLE	SUBREGION2_DISABLE	SUBREGION1_DISABLE	SUBREGION0_DISABLE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED			SIZE				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
RESERVED		PROT_TYPE		RESERVED			ENABLE
R-0h		R/W-0h		R-0h			R/W-0h

Table 41-102. REGION2_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23	SUBREGION7_DISABLE	R/W	0h	Disable Sub Region-7 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
22	SUBREGION6_DISABLE	R/W	0h	Disable Sub Region-6 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
21	SUBREGION5_DISABLE	R/W	0h	Disable Sub Region-5 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
20	SUBREGION4_DISABLE	R/W	0h	Disable Sub Region-4 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
19	SUBREGION3_DISABLE	R/W	0h	Disable Sub Region-3 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
18	SUBREGION2_DISABLE	R/W	0h	Disable Sub Region-2 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
17	SUBREGION1_DISABLE	R/W	0h	Disable Sub Region-1 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn

Table 41-102. REGION2_CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	SUBREGION0_DISABLE	R/W	0h	Disable Sub Region-0 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
15-13	RESERVED	R	0h	Reserved
12-8	SIZE	R/W	0h	Size of the region Region size = 2^{SIZE}*1KB 0x0 : 1KB 0x1 : 2KB .... 0x16: Covers full 32 bit address range (4GB) 0x17 to 0x1F : Reserved (same as 0x0) Reset type: CM.RESETn
7-6	RESERVED	R	0h	Reserved
5-4	PROT_TYPE	R/W	0h	Defines access permission for region-2 00: Read Only 10: No Access, Read and Write are not allowed 01: Full Access, both Read and Write are allowed 11: Reserved (Read only) Reset type: CM.RESETn
3-1	RESERVED	R	0h	Reserved
0	ENABLE	R/W	0h	Enable Region-2 0: Region Disabled 1: Region Enabled Reset type: CM.RESETn

41.12.7.13 REGION3_STARTADDRESSS Register (Offset = 58h) [Reset = 0h]

REGION3_STARTADDRESSS is shown in Figure 41-96 and described in Table 41-103.

Return to the Summary Table.

Region 3 start address register

Figure 41-96. REGION3_STARTADDRESSS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
START_ADDR																															
R/W-0h																															

Table 41-103. REGION3_STARTADDRESSS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	START_ADDR	R/W	0h	Start address of region-3. Address written to this register must be aligned to multiples of region size. Address must be >=1KB boundary Reset type: CM.RESETn

41.12.7.14 REGION3_CONFIG Register (Offset = 5Ch) [Reset = 0h]

REGION3_CONFIG is shown in Figure 41-97 and described in Table 41-104.

Return to the Summary Table.

Region 3 configuration register

Figure 41-97. REGION3_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
SUBREGION7_DISABLE	SUBREGION6_DISABLE	SUBREGION5_DISABLE	SUBREGION4_DISABLE	SUBREGION3_DISABLE	SUBREGION2_DISABLE	SUBREGION1_DISABLE	SUBREGION0_DISABLE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED			SIZE				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
RESERVED		PROT_TYPE		RESERVED			ENABLE
R-0h		R/W-0h		R-0h			R/W-0h

Table 41-104. REGION3_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23	SUBREGION7_DISABLE	R/W	0h	Disable Sub Region-7 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
22	SUBREGION6_DISABLE	R/W	0h	Disable Sub Region-6 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
21	SUBREGION5_DISABLE	R/W	0h	Disable Sub Region-5 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
20	SUBREGION4_DISABLE	R/W	0h	Disable Sub Region-4 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
19	SUBREGION3_DISABLE	R/W	0h	Disable Sub Region-3 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
18	SUBREGION2_DISABLE	R/W	0h	Disable Sub Region-2 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
17	SUBREGION1_DISABLE	R/W	0h	Disable Sub Region-1 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn

Table 41-104. REGION3_CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	SUBREGION0_DISABLE	R/W	0h	Disable Sub Region-0 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
15-13	RESERVED	R	0h	Reserved
12-8	SIZE	R/W	0h	Size of the region Region size = 2^{SIZE}*1KB 0x0 : 1KB 0x1 : 2KB .... 0x16: Covers full 32 bit address range (4GB) 0x17 to 0x1F : Reserved (same as 0x0) Reset type: CM.RESETn
7-6	RESERVED	R	0h	Reserved
5-4	PROT_TYPE	R/W	0h	Defines access permission for region-3 00: Read Only 10: No Access, Read and Write are not allowed 01: Full Access, both Read and Write are allowed 11: Reserved (Read only) Reset type: CM.RESETn
3-1	RESERVED	R	0h	Reserved
0	ENABLE	R/W	0h	Enable Region-3 0: Region Disabled 1: Region Enabled Reset type: CM.RESETn

41.12.7.15 REGION4_STARTADDRESSS Register (Offset = 60h) [Reset = 0h]

REGION4_STARTADDRESSS is shown in Figure 41-98 and described in Table 41-105.

Return to the Summary Table.

Region 4 start address register

Figure 41-98. REGION4_STARTADDRESSS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
START_ADDR																															
R/W-0h																															

Table 41-105. REGION4_STARTADDRESSS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	START_ADDR	R/W	0h	Start address of region-4. Address written to this register must be aligned to multiples of region size. Address must be >=1KB boundary Reset type: CM.RESETn

41.12.7.16 REGION4_CONFIG Register (Offset = 64h) [Reset = 0h]

REGION4_CONFIG is shown in Figure 41-99 and described in Table 41-106.

Return to the Summary Table.

Region 4 configuration register

Figure 41-99. REGION4_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
SUBREGION7_DISABLE	SUBREGION6_DISABLE	SUBREGION5_DISABLE	SUBREGION4_DISABLE	SUBREGION3_DISABLE	SUBREGION2_DISABLE	SUBREGION1_DISABLE	SUBREGION0_DISABLE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED			SIZE				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
RESERVED		PROT_TYPE		RESERVED			ENABLE
R-0h		R/W-0h		R-0h			R/W-0h

Table 41-106. REGION4_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23	SUBREGION7_DISABLE	R/W	0h	Disable Sub Region-7 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
22	SUBREGION6_DISABLE	R/W	0h	Disable Sub Region-6 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
21	SUBREGION5_DISABLE	R/W	0h	Disable Sub Region-5 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
20	SUBREGION4_DISABLE	R/W	0h	Disable Sub Region-4 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
19	SUBREGION3_DISABLE	R/W	0h	Disable Sub Region-3 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
18	SUBREGION2_DISABLE	R/W	0h	Disable Sub Region-2 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
17	SUBREGION1_DISABLE	R/W	0h	Disable Sub Region-1 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn

Table 41-106. REGION4_CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	SUBREGION0_DISABLE	R/W	0h	Disable Sub Region-0 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
15-13	RESERVED	R	0h	Reserved
12-8	SIZE	R/W	0h	Size of the region Region size = 2^{SIZE}*1KB 0x0 : 1KB 0x1 : 2KB .... 0x16: Covers full 32 bit address range (4GB) 0x17 to 0x1F : Reserved (same as 0x0) Reset type: CM.RESETn
7-6	RESERVED	R	0h	Reserved
5-4	PROT_TYPE	R/W	0h	Defines access permission for region-4 00: Read Only 10: No Access, Read and Write are not allowed 01: Full Access, both Read and Write are allowed 11: Reserved (Read only) Reset type: CM.RESETn
3-1	RESERVED	R	0h	Reserved
0	ENABLE	R/W	0h	Enable Region-4 0: Region Disabled 1: Region Enabled Reset type: CM.RESETn

41.12.7.17 REGION5_STARTADDRESSS Register (Offset = 68h) [Reset = 0h]

REGION5_STARTADDRESSS is shown in Figure 41-100 and described in Table 41-107.

Return to the Summary Table.

Region 5 start address register

Figure 41-100. REGION5_STARTADDRESSS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
START_ADDR																															
R/W-0h																															

Table 41-107. REGION5_STARTADDRESSS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	START_ADDR	R/W	0h	Start address of region-5. Address written to this register must be aligned to multiples of region size. Address must be >=1KB boundary Reset type: CM.RESETn

41.12.7.18 REGION5_CONFIG Register (Offset = 6Ch) [Reset = 0h]

REGION5_CONFIG is shown in Figure 41-101 and described in Table 41-108.

Return to the Summary Table.

Region 5 configuration register

Figure 41-101. REGION5_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
SUBREGION7_DISABLE	SUBREGION6_DISABLE	SUBREGION5_DISABLE	SUBREGION4_DISABLE	SUBREGION3_DISABLE	SUBREGION2_DISABLE	SUBREGION1_DISABLE	SUBREGION0_DISABLE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED			SIZE				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
RESERVED		PROT_TYPE		RESERVED			ENABLE
R-0h		R/W-0h		R-0h			R/W-0h

Table 41-108. REGION5_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23	SUBREGION7_DISABLE	R/W	0h	Disable Sub Region-7 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
22	SUBREGION6_DISABLE	R/W	0h	Disable Sub Region-6 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
21	SUBREGION5_DISABLE	R/W	0h	Disable Sub Region-5 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
20	SUBREGION4_DISABLE	R/W	0h	Disable Sub Region-4 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
19	SUBREGION3_DISABLE	R/W	0h	Disable Sub Region-3 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
18	SUBREGION2_DISABLE	R/W	0h	Disable Sub Region-2 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
17	SUBREGION1_DISABLE	R/W	0h	Disable Sub Region-1 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn

Table 41-108. REGION5_CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	SUBREGION0_DISABLE	R/W	0h	Disable Sub Region-0 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
15-13	RESERVED	R	0h	Reserved
12-8	SIZE	R/W	0h	Size of the region Region size = 2^{SIZE}*1KB 0x0 : 1KB 0x1 : 2KB .... 0x16: Covers full 32 bit address range (4GB) 0x17 to 0x1F : Reserved (same as 0x0) Reset type: CM.RESETn
7-6	RESERVED	R	0h	Reserved
5-4	PROT_TYPE	R/W	0h	Defines access permission for region-5 00: Read Only 10: No Access, Read and Write are not allowed 01: Full Access, both Read and Write are allowed 11: Reserved (Read only) Reset type: CM.RESETn
3-1	RESERVED	R	0h	Reserved
0	ENABLE	R/W	0h	Enable Region-5 0: Region Disabled 1: Region Enabled Reset type: CM.RESETn

41.12.7.19 REGION6_STARTADDRESSS Register (Offset = 70h) [Reset = 0h]

REGION6_STARTADDRESSS is shown in Figure 41-102 and described in Table 41-109.

Return to the Summary Table.

Region 6 start address register

Figure 41-102. REGION6_STARTADDRESSS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
START_ADDR																															
R/W-0h																															

Table 41-109. REGION6_STARTADDRESSS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	START_ADDR	R/W	0h	Start address of region-6. Address written to this register must be aligned to multiples of region size. Address must be >=1KB boundary Reset type: CM.RESETn

41.12.7.20 REGION6_CONFIG Register (Offset = 74h) [Reset = 0h]

REGION6_CONFIG is shown in Figure 41-103 and described in Table 41-110.

Return to the Summary Table.

Region 6 configuration register

Figure 41-103. REGION6_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
SUBREGION7_DISABLE	SUBREGION6_DISABLE	SUBREGION5_DISABLE	SUBREGION4_DISABLE	SUBREGION3_DISABLE	SUBREGION2_DISABLE	SUBREGION1_DISABLE	SUBREGION0_DISABLE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED			SIZE				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
RESERVED		PROT_TYPE		RESERVED			ENABLE
R-0h		R/W-0h		R-0h			R/W-0h

Table 41-110. REGION6_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23	SUBREGION7_DISABLE	R/W	0h	Disable Sub Region-7 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
22	SUBREGION6_DISABLE	R/W	0h	Disable Sub Region-6 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
21	SUBREGION5_DISABLE	R/W	0h	Disable Sub Region-5 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
20	SUBREGION4_DISABLE	R/W	0h	Disable Sub Region-4 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
19	SUBREGION3_DISABLE	R/W	0h	Disable Sub Region-3 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
18	SUBREGION2_DISABLE	R/W	0h	Disable Sub Region-2 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
17	SUBREGION1_DISABLE	R/W	0h	Disable Sub Region-1 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn

Table 41-110. REGION6_CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	SUBREGION0_DISABLE	R/W	0h	Disable Sub Region-0 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
15-13	RESERVED	R	0h	Reserved
12-8	SIZE	R/W	0h	Size of the region Region size = 2^{SIZE}*1KB 0x0 : 1KB 0x1 : 2KB .... 0x16: Covers full 32 bit address range (4GB) 0x17 to 0x1F : Reserved (same as 0x0) Reset type: CM.RESETn
7-6	RESERVED	R	0h	Reserved
5-4	PROT_TYPE	R/W	0h	Defines access permission for region-6 00: Read Only 10: No Access, Read and Write are not allowed 01: Full Access, both Read and Write are allowed 11: Reserved (Read only) Reset type: CM.RESETn
3-1	RESERVED	R	0h	Reserved
0	ENABLE	R/W	0h	Enable Region-6 0: Region Disabled 1: Region Enabled Reset type: CM.RESETn

41.12.7.21 REGION7_STARTADDRESSS Register (Offset = 78h) [Reset = 0h]

REGION7_STARTADDRESSS is shown in Figure 41-104 and described in Table 41-111.

Return to the Summary Table.

Region 7 start address register

Figure 41-104. REGION7_STARTADDRESSS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
START_ADDR																															
R/W-0h																															

Table 41-111. REGION7_STARTADDRESSS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	START_ADDR	R/W	0h	Start address of region-7. Address written to this register must be aligned to multiples of region size. Address must be >=1KB boundary Reset type: CM.RESETn

41.12.7.22 REGION7_CONFIG Register (Offset = 7Ch) [Reset = 0h]

REGION7_CONFIG is shown in Figure 41-105 and described in Table 41-112.

Return to the Summary Table.

Region 7 configuration register

Figure 41-105. REGION7_CONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
SUBREGION7_DISABLE	SUBREGION6_DISABLE	SUBREGION5_DISABLE	SUBREGION4_DISABLE	SUBREGION3_DISABLE	SUBREGION2_DISABLE	SUBREGION1_DISABLE	SUBREGION0_DISABLE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED			SIZE				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
RESERVED		PROT_TYPE		RESERVED			ENABLE
R-0h		R/W-0h		R-0h			R/W-0h

Table 41-112. REGION7_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23	SUBREGION7_DISABLE	R/W	0h	Disable Sub Region-7 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
22	SUBREGION6_DISABLE	R/W	0h	Disable Sub Region-6 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
21	SUBREGION5_DISABLE	R/W	0h	Disable Sub Region-5 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
20	SUBREGION4_DISABLE	R/W	0h	Disable Sub Region-4 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
19	SUBREGION3_DISABLE	R/W	0h	Disable Sub Region-3 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
18	SUBREGION2_DISABLE	R/W	0h	Disable Sub Region-2 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
17	SUBREGION1_DISABLE	R/W	0h	Disable Sub Region-1 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn

Table 41-112. REGION7_CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	SUBREGION0_DISABLE	R/W	0h	Disable Sub Region-0 0: Sub region enabled 1: Sub region disabled Reset type: CM.RESETn
15-13	RESERVED	R	0h	Reserved
12-8	SIZE	R/W	0h	Size of the region Region size = 2^{SIZE}*1KB 0x0 : 1KB 0x1 : 2KB .... 0x16: Covers full 32 bit address range (4GB) 0x17 to 0x1F : Reserved (same as 0x0) Reset type: CM.RESETn
7-6	RESERVED	R	0h	Reserved
5-4	PROT_TYPE	R/W	0h	Defines access permission for region-7 00: Read Only 10: No Access, Read and Write are not allowed 01: Full Access, both Read and Write are allowed 11: Reserved (Read only) Reset type: CM.RESETn
3-1	RESERVED	R	0h	Reserved
0	ENABLE	R/W	0h	Enable Region-7 0: Region Disabled 1: Region Enabled Reset type: CM.RESETn

41.12.8 CM_NMI_INTRUPT_REGS Registers

Table 41-113 lists the memory-mapped registers for the CM_NMI_INTRUPT_REGS registers. All register offset addresses not listed in Table 41-113 should be considered as reserved locations and the register contents should not be modified.

Table 41-113. CM_NMI_INTRUPT_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CMNMICFG	CM NMI Configuration Register		Go
4h	CMNMIFLG	CM NMI Flag Register		Go
8h	CMNMIFLGCLR	CMNMI Flag Clear Register		Go
Ch	CMNMIFLGFRC	CMNMI Flag Force Register		Go
10h	CMNMIWDCNT	CMNMI Watchdog Counter Register		Go
14h	CMNMIWDPRD	CMNMI Watchdog Period Register		Go
18h	CMNMISHDWFLG	CMNMI Shadow Flag Register		Go

Complex bit access types are encoded to fit into small table cells. Table 41-114 shows the codes that are used for access types in this section.

Table 41-114. CM_NMI_INTRUPT_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

41.12.8.1 CMNMICFG Register (Offset = 0h) [Reset = 0h]

CMNMICFG is shown in Figure 41-106 and described in Table 41-115.

Return to the Summary Table.

CM NMI Configuration Register

Figure 41-106. CMNMICFG Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							NMIE
R-0h							R/W1S-0h

Table 41-115. CMNMICFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to any of the fields will suceed only if a value 0x6789 appears on the KEY field. Reset type: CM.RESETn
15-1	RESERVED	R	0h	Reserved
0	NMIE	R/W1S	0h	When set to 1 any condition will generate an NMI interrupt to the CM4 CPU and kick off the NMI watchdog counter. As part of boot sequence this bit should be set after the device security related initialization is complete. 0 NMI disabled 1 NMI enabled Reset type: CM.RESETn

41.12.8.2 CMNMIFLG Register (Offset = 4h) [Reset = 0h]

CMNMIFLG is shown in Figure 41-107 and described in Table 41-116.

Return to the Summary Table.

CM NMI Flag Register

Figure 41-107. CMNMIFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	ECATNMI	WWDNMI	MCANUNCERR	FLUNCERR	MEMUNCERR	CLOCKFAIL	NMIINT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 41-116. CMNMIFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	ECATNMI	R	0h	0 No reset request from EtherCAT IP. 1 NMI generated from EtherCAT IP. No further NMI pulses are generated until this flag is cleared by the user. Reset type: CM.RESETn
5	WWDNMI	R	0h	CM Windowed Watchdog NMI Flag: This bits indicates if CM watch dog generated an NMI. 0 CM WWD has not generated an NMI 1 CM WWD has generated an NMI Reset type: CM.RESETn
4	MCANUNCERR	R	0h	MCAN Uncorrectable Error NMI Flag: This bit indicates if an uncorrectable error occurred on an access MCAN message RAM, and that condition is latched. This bit can only be cleared by the user writing to the corresponding clear bit in the CMNMIFLGCLR register or by an XRSn reset: 0,No CM4 Flash uncorrectable error condition pending 1,CM4 Flash uncorrectable error condition generated Reset type: CM.RESETn
3	FLUNCERR	R	0h	Flash Uncorrectable Error NMI Flag: This bit indicates if an uncorrectable error occurred on a CM4 Flash access and that condition is latched. This bit can only be cleared by the user writing to the corresponding clear bit in the CMNMIFLGCLR register or by an XRSn reset: 0,No CM4 Flash uncorrectable error condition pending 1,CM4 Flash uncorrectable error condition generated Reset type: CM.RESETn

Table 41-116. CMNMIFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MEMUNCERR	R	0h	RAM/ROM Uncorrectable Error NMI Flag: This bit indicates if an uncorrectable error occurred on a RAM (Including peripheral RAMs)/ROM access (by any master) and that condition is latched. This bit can only be cleared by the user writing to the corresponding clear bit in the CMNMIFLGCLR register or by an XRSn reset: 0,No RAM/ROM uncorrectable error condition pending 1,RAM/ROM uncorrectable error condition generated Reset type: CM.RESETn
1	CLOCKFAIL	R	0h	Clock Fail Interrupt Flag: These bits indicates if the CLOCKFAIL condition is latched. These bits can only be cleared by the user writing to the respective bit in the CMNMIFLGCLR register or by an XRSn reset: 0,No CLOCKFAIL Condition Pending 1,CLOCKFAIL Condition Generated Reset type: CM.RESETn
0	NMIINT	R	0h	NMI Interrupt Flag: This bit indicates if an NMI interrupt was generated. This bit can only be cleared by the user writing to the respective bit in the CMNMIFLGCLR register or by an XRSn reset: 0 No NMI Interrupt Generated 1 NMI Interrupt Generated No further NMI interrupts pulses are generated until this flag is cleared by the user. Reset type: CM.RESETn

41.12.8.3 CMNMIFLGCLR Register (Offset = 8h) [Reset = 0h]

CMNMIFLGCLR is shown in Figure 41-108 and described in Table 41-117.

Return to the Summary Table.

CMNMI Flag Clear Register

Figure 41-108. CMNMIFLGCLR Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	ECATNMI	WWDNMI	MCANUNCERR	FLUNCERR	MEMUNCERR	CLOCKFAIL	NMIINT
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 41-117. CMNMIFLGCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to any of the fields will suceed only if a value 0x5674 appears on the KEY field. Reset type: CM.RESETn
15-7	RESERVED	R	0h	Reserved
6	ECATNMI	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NCMMIFLG and CMNMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: CM.RESETn
5	WWDNMI	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NCMMIFLG and CMNMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: CM.RESETn
4	MCANUNCERR	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NCMMIFLG and CMNMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: CM.RESETn

Table 41-117. CMNMIFLGCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	FLUNCERR	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NCMMIFLG and CMNMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: CM.RESETn
2	MEMUNCERR	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NCMMIFLG and CMNMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: CM.RESETn
1	CLOCKFAIL	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NCMMIFLG and CMNMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: CM.RESETn
0	NMIINT	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NCMMIFLG and CMNMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: CM.RESETn

41.12.8.4 CMNMIFLGFRC Register (Offset = Ch) [Reset = 0h]

CMNMIFLGFRC is shown in Figure 41-109 and described in Table 41-118.

Return to the Summary Table.

CMNMI Flag Force Register

Figure 41-109. CMNMIFLGFRC Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	ECATNMI	WWDNMI	MCANUNCERR	FLUNCERR	MEMUNCERR	CLOCKFAIL	RESERVED
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0h

Table 41-118. CMNMIFLGFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to any of the fields will suceed only if a value 0x2732 appears on the KEY field. Reset type: CM.RESETn
15-7	RESERVED	R	0h	Reserved
6	ECATNMI	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the CMNMIFLG and CMNMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: CM.RESETn
5	WWDNMI	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the CMNMIFLG and CMNMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: CM.RESETn
4	MCANUNCERR	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the CMNMIFLG and CMNMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: CM.RESETn
3	FLUNCERR	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the CMNMIFLG and CMNMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: CM.RESETn
2	MEMUNCERR	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the CMNMIFLG and CMNMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: CM.RESETn

Table 41-118. CMNMIFLGFRC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	CLOCKFAIL	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the CMNMIFLG and CMNMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: CM.RESETn
0	RESERVED	R	0h	Reserved

41.12.8.5 CMNMIWDCNT Register (Offset = 10h) [Reset = 0h]

CMNMIWDCNT is shown in Figure 41-110 and described in Table 41-119.

Return to the Summary Table.

CMNMI Watchdog Counter Register

Figure 41-110. CMNMIWDCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																NMIWDCNT															
R-0h																R-0h															

Table 41-119. CMNMIWDCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	NMIWDCNT	R	0h	NMI Watchdog Counter: This 16-bit incremental counter will start incrementing whenever any one of the enabled FAIL flags are set. If the counter reaches the period value, an NMIRSn signal is fired which will then resets the system. If no enabled FAIL flag is set, then the counter will reset to zero and remain at zero until an enabled FAIL flag is set. Normally, the software would respond to the NMI interrupt generated and clear the offending FLAG(s) before the NMI watchdog triggers a reset. In some situations, the software may decide to allow the watchdog to reset the device anyway. The counter is clocked at the CMCLK rate. Reset type: CM.RESETn

41.12.8.6 CMNMIWDPRD Register (Offset = 14h) [Reset = FFFFh]

CMNMIWDPRD is shown in Figure 41-111 and described in Table 41-120.

Return to the Summary Table.

CMNMI Watchdog Period Register

Figure 41-111. CMNMIWDPRD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																NMIWDPRD															
R-0/W-0h																R/W-FFFFh															

Table 41-120. CMNMIWDPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to any of the fields of this register will suceed only if a value 0x9238 appears on the KEY field. Reset type: CM.RESETn
15-0	NMIWDPRD	R/W	FFFFh	NMI Watchdog Period: This 16-bit value contains the period value at which a reset is generated when the watchdog counter matches. At reset this value is set at the maximum. The software can decrease the period value at initialization time. Note:If a PERIOD value is written that is smaller than the current counter value, the counter will continue counting until it overflows and starts counting up again from 0. After the overflow, once the COUNTER value equals the new PERIOD value, an NMIRS is forced which resets the watchdog counter. Reset type: CM.RESETn

41.12.8.7 CMNMISHDWFLG Register (Offset = 18h) [Reset = 0h]

CMNMISHDWFLG is shown in Figure 41-112 and described in Table 41-121.

Return to the Summary Table.

CMNMI Shadow Flag Register

Figure 41-112. CMNMISHDWFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	ECATNMI	WWDNMI	MCANUNCERR	FLUNCERR	MEMUNCERR	CLOCKFAIL	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 41-121. CMNMISHDWFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	ECATNMI	R	0h	Shadow NMI Flags: When an CMNMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that CMNMIFLGFRC and CMNMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is resetted only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
5	WWDNMI	R	0h	Shadow NMI Flags: When an CMNMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that CMNMIFLGFRC and CMNMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is resetted only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
4	MCANUNCERR	R	0h	Shadow NMI Flags: When an CMNMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that CMNMIFLGFRC and CMNMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is resetted only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn

Table 41-121. CMNMISHDWFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	FLUNCERR	R	0h	Shadow NMI Flags: When an CMNMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that CMNMIFLGFRC and CMNMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is resetted only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
2	MEMUNCERR	R	0h	Shadow NMI Flags: When an CMNMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that CMNMIFLGFRC and CMNMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is resetted only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
1	CLOCKFAIL	R	0h	Shadow NMI Flags: When an CMNMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that CMNMIFLGFRC and CMNMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is resetted only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
0	RESERVED	R	0h	Reserved

41.12.9 NVIC Registers

Table 41-122 lists the memory-mapped registers for the NVIC registers. All register offset addresses not listed in Table 41-122 should be considered as reserved locations and the register contents should not be modified.

Table 41-122. NVIC Registers

Offset	Acronym	Register Name	Write Protection	Section
100h	NVIC_ISER0	NVIC Interrupt Set Enable Register 0		Go
104h	NVIC_ISER1	NVIC Interrupt Set Enable Register 1		Go
180h	NVIC_ICER0	NVIC Interrupt Clear Enable Register 0		Go
184h	NVIC_ICER1	NVIC Interrupt Clear Enable Register 1		Go
200h	NVIC_ISPR0	NVIC Interrupt Set Pending Register 0		Go
204h	NVIC_ISPR1	NVIC Interrupt Set Pending Register 1		Go
208h	NVIC_ISPR2	NVIC Interrupt Set Pending Register 2		Go
280h	NVIC_ICPR0	NVIC Interrupt Clear Pending Register 0		Go
284h	NVIC_ICPR1	NVIC Interrupt Clear Pending Register 1		Go
300h	NVIC_IABR0	NVIC Interrupt Active Bit Register 0		Go
304h	NVIC_IABR1	NVIC Interrupt Active Bit Register 1		Go
400h	NVIC_IPR0	NVIC Interrupt Priority Register 0		Go
404h	NVIC_IPR1	NVIC Interrupt Priority Register 1		Go
408h	NVIC_IPR2	NVIC Interrupt Priority Register 2		Go
40Ch	NVIC_IPR3	NVIC Interrupt Priority Register 3		Go
410h	NVIC_IPR4	NVIC Interrupt Priority Register 4		Go
414h	NVIC_IPR5	NVIC Interrupt Priority Register 5		Go
418h	NVIC_IPR6	NVIC Interrupt Priority Register 6		Go
41Ch	NVIC_IPR7	NVIC Interrupt Priority Register 7		Go
420h	NVIC_IPR8	NVIC Interrupt Priority Register 8		Go
424h	NVIC_IPR9	NVIC Interrupt Priority Register 9		Go
428h	NVIC_IPR10	NVIC Interrupt Priority Register 10		Go
42Ch	NVIC_IPR11	NVIC Interrupt Priority Register 11		Go
430h	NVIC_IPR12	NVIC Interrupt Priority Register 12		Go
434h	NVIC_IPR13	NVIC Interrupt Priority Register 13		Go
438h	NVIC_IPR14	NVIC Interrupt Priority Register 14		Go
43Ch	NVIC_IPR15	NVIC Interrupt Priority Register 15		Go
F00h	STIR	Software Trigger Interrupt Register		Go

Complex bit access types are encoded to fit into small table cells. Table 41-123 shows the codes that are used for access types in this section.

Table 41-123. NVIC Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value

Table 41-123. NVIC Access Type Codes (continued)

Access Type	Code	Description
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

41.12.9.1 NVIC_ISER0 Register (Offset = 100h) [Reset = 0h]

NVIC_ISER0 is shown in Figure 41-113 and described in Table 41-124.

Return to the Summary Table.

NVIC Interrupt Set Enable Register 0

Figure 41-113. NVIC_ISER0 Register

31	30	29	28	27	26	25	24
SETENA31	SETENA30	SETENA29	SETENA28	SETENA27	SETENA26	SETENA25	SETENA24
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
23	22	21	20	19	18	17	16
SETENA23	SETENA22	SETENA21	SETENA20	SETENA19	SETENA18	SETENA17	SETENA16
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
15	14	13	12	11	10	9	8
SETENA15	SETENA14	SETENA13	SETENA12	SETENA11	SETENA10	SETENA9	SETENA8
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
SETENA7	SETENA6	SETENA5	SETENA4	SETENA3	SETENA2	SETENA1	SETENA0
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

Table 41-124. NVIC_ISER0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SETENA31	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt31. Read: 0 = interrupt31 disabled 1 = interrupt31 enabled. Reset type: CM.SYSRESETn
30	SETENA30	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt30. Read: 0 = interrupt30 disabled 1 = interrupt30 enabled. Reset type: CM.SYSRESETn
29	SETENA29	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt29. Read: 0 = interrupt29 disabled 1 = interrupt29 enabled. Reset type: CM.SYSRESETn
28	SETENA28	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt28. Read: 0 = interrupt28 disabled 1 = interrupt28 enabled. Reset type: CM.SYSRESETn

Table 41-124. NVIC_ISER0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	SETENA27	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt27. Read: 0 = interrupt27 disabled 1 = interrupt27 enabled. Reset type: CM.SYSRESETn
26	SETENA26	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt26. Read: 0 = interrupt26 disabled 1 = interrupt26 enabled. Reset type: CM.SYSRESETn
25	SETENA25	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt25. Read: 0 = interrupt25 disabled 1 = interrupt25 enabled. Reset type: CM.SYSRESETn
24	SETENA24	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt24. Read: 0 = interrupt24 disabled 1 = interrupt24 enabled. Reset type: CM.SYSRESETn
23	SETENA23	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt23. Read: 0 = interrupt23 disabled 1 = interrupt23 enabled. Reset type: CM.SYSRESETn
22	SETENA22	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt22. Read: 0 = interrupt22 disabled 1 = interrupt22 enabled. Reset type: CM.SYSRESETn
21	SETENA21	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt21. Read: 0 = interrupt21 disabled 1 = interrupt21 enabled. Reset type: CM.SYSRESETn

Table 41-124. NVIC_ISER0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	SETENA20	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt20. Read: 0 = interrupt20 disabled 1 = interrupt20 enabled. Reset type: CM.SYSRESETn
19	SETENA19	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt19. Read: 0 = interrupt19 disabled 1 = interrupt19 enabled. Reset type: CM.SYSRESETn
18	SETENA18	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt18. Read: 0 = interrupt18 disabled 1 = interrupt18 enabled. Reset type: CM.SYSRESETn
17	SETENA17	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt17. Read: 0 = interrupt17 disabled 1 = interrupt17 enabled. Reset type: CM.SYSRESETn
16	SETENA16	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt16. Read: 0 = interrupt16 disabled 1 = interrupt16 enabled. Reset type: CM.SYSRESETn
15	SETENA15	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt15. Read: 0 = interrupt15 disabled 1 = interrupt15 enabled. Reset type: CM.SYSRESETn
14	SETENA14	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt14. Read: 0 = interrupt14 disabled 1 = interrupt14 enabled. Reset type: CM.SYSRESETn

Table 41-124. NVIC_ISER0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	SETENA13	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt13. Read: 0 = interrupt13 disabled 1 = interrupt13 enabled. Reset type: CM.SYSRESETn
12	SETENA12	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt12. Read: 0 = interrupt12 disabled 1 = interrupt12 enabled. Reset type: CM.SYSRESETn
11	SETENA11	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt11. Read: 0 = interrupt11 disabled 1 = interrupt11 enabled. Reset type: CM.SYSRESETn
10	SETENA10	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt10. Read: 0 = interrupt10 disabled 1 = interrupt10 enabled. Reset type: CM.SYSRESETn
9	SETENA9	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt9. Read: 0 = interrupt9 disabled 1 = interrupt9 enabled. Reset type: CM.SYSRESETn
8	SETENA8	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt8. Read: 0 = interrupt8 disabled 1 = interrupt8 enabled. Reset type: CM.SYSRESETn
7	SETENA7	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt7. Read: 0 = interrupt7 disabled 1 = interrupt7 enabled. Reset type: CM.SYSRESETn

Table 41-124. NVIC_ISER0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	SETENA6	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt6. Read: 0 = interrupt6 disabled 1 = interrupt6 enabled. Reset type: CM.SYSRESETn
5	SETENA5	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt5. Read: 0 = interrupt5 disabled 1 = interrupt5 enabled. Reset type: CM.SYSRESETn
4	SETENA4	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt4. Read: 0 = interrupt4 disabled 1 = interrupt4 enabled. Reset type: CM.SYSRESETn
3	SETENA3	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt3. Read: 0 = interrupt3 disabled 1 = interrupt3 enabled. Reset type: CM.SYSRESETn
2	SETENA2	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt2. Read: 0 = interrupt2 disabled 1 = interrupt2 enabled. Reset type: CM.SYSRESETn
1	SETENA1	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt1. Read: 0 = interrupt1 disabled 1 = interrupt1 enabled. Reset type: CM.SYSRESETn
0	SETENA0	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt0. Read: 0 = interrupt0 disabled 1 = interrupt0 enabled. Reset type: CM.SYSRESETn

41.12.9.2 NVIC_ISER1 Register (Offset = 104h) [Reset = 0h]

NVIC_ISER1 is shown in Figure 41-114 and described in Table 41-125.

Return to the Summary Table.

NVIC Interrupt Set Enable Register 1

Figure 41-114. NVIC_ISER1 Register

31	30	29	28	27	26	25	24
SETENA63	SETENA62	SETENA61	SETENA60	SETENA59	SETENA58	SETENA57	SETENA56
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
23	22	21	20	19	18	17	16
SETENA55	SETENA54	SETENA53	SETENA52	SETENA51	SETENA50	SETENA49	SETENA48
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
15	14	13	12	11	10	9	8
SETENA47	SETENA46	SETENA45	SETENA44	SETENA43	SETENA42	SETENA41	SETENA40
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
SETENA39	SETENA38	SETENA37	SETENA36	SETENA35	SETENA34	SETENA33	SETENA32
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

Table 41-125. NVIC_ISER1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SETENA63	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt63. Read: 0 = interrupt63 disabled 1 = interrupt63 enabled. Reset type: CM.SYSRESETn
30	SETENA62	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt62. Read: 0 = interrupt62 disabled 1 = interrupt62 enabled. Reset type: CM.SYSRESETn
29	SETENA61	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt61. Read: 0 = interrupt61 disabled 1 = interrupt61 enabled. Reset type: CM.SYSRESETn
28	SETENA60	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt60. Read: 0 = interrupt60 disabled 1 = interrupt60 enabled. Reset type: CM.SYSRESETn

Table 41-125. NVIC_ISER1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	SETENA59	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt59. Read: 0 = interrupt59 disabled 1 = interrupt59 enabled. Reset type: CM.SYSRESETn
26	SETENA58	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt58. Read: 0 = interrupt58 disabled 1 = interrupt58 enabled. Reset type: CM.SYSRESETn
25	SETENA57	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt57. Read: 0 = interrupt57 disabled 1 = interrupt57 enabled. Reset type: CM.SYSRESETn
24	SETENA56	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt56. Read: 0 = interrupt56 disabled 1 = interrupt56 enabled. Reset type: CM.SYSRESETn
23	SETENA55	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt55. Read: 0 = interrupt55 disabled 1 = interrupt55 enabled. Reset type: CM.SYSRESETn
22	SETENA54	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt54. Read: 0 = interrupt54 disabled 1 = interrupt54 enabled. Reset type: CM.SYSRESETn
21	SETENA53	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt53. Read: 0 = interrupt53 disabled 1 = interrupt53 enabled. Reset type: CM.SYSRESETn

Table 41-125. NVIC_ISER1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	SETENA52	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt52. Read: 0 = interrupt52 disabled 1 = interrupt52 enabled. Reset type: CM.SYSRESETn
19	SETENA51	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt51. Read: 0 = interrupt51 disabled 1 = interrupt51 enabled. Reset type: CM.SYSRESETn
18	SETENA50	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt50. Read: 0 = interrupt50 disabled 1 = interrupt50 enabled. Reset type: CM.SYSRESETn
17	SETENA49	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt49. Read: 0 = interrupt49 disabled 1 = interrupt49 enabled. Reset type: CM.SYSRESETn
16	SETENA48	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt48. Read: 0 = interrupt48 disabled 1 = interrupt48 enabled. Reset type: CM.SYSRESETn
15	SETENA47	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt47. Read: 0 = interrupt47 disabled 1 = interrupt47 enabled. Reset type: CM.SYSRESETn
14	SETENA46	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt46. Read: 0 = interrupt46 disabled 1 = interrupt46 enabled. Reset type: CM.SYSRESETn

Table 41-125. NVIC_ISER1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	SETENA45	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt45. Read: 0 = interrupt45 disabled 1 = interrupt45 enabled. Reset type: CM.SYSRESETn
12	SETENA44	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt44. Read: 0 = interrupt44 disabled 1 = interrupt44 enabled. Reset type: CM.SYSRESETn
11	SETENA43	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt43. Read: 0 = interrupt43 disabled 1 = interrupt43 enabled. Reset type: CM.SYSRESETn
10	SETENA42	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt42. Read: 0 = interrupt42 disabled 1 = interrupt42 enabled. Reset type: CM.SYSRESETn
9	SETENA41	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt41. Read: 0 = interrupt41 disabled 1 = interrupt41 enabled. Reset type: CM.SYSRESETn
8	SETENA40	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt40. Read: 0 = interrupt40 disabled 1 = interrupt40 enabled. Reset type: CM.SYSRESETn
7	SETENA39	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt39. Read: 0 = interrupt39 disabled 1 = interrupt39 enabled. Reset type: CM.SYSRESETn

Table 41-125. NVIC_ISER1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	SETENA38	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt38. Read: 0 = interrupt38 disabled 1 = interrupt38 enabled. Reset type: CM.SYSRESETn
5	SETENA37	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt37. Read: 0 = interrupt37 disabled 1 = interrupt37 enabled. Reset type: CM.SYSRESETn
4	SETENA36	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt36. Read: 0 = interrupt36 disabled 1 = interrupt36 enabled. Reset type: CM.SYSRESETn
3	SETENA35	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt35. Read: 0 = interrupt35 disabled 1 = interrupt35 enabled. Reset type: CM.SYSRESETn
2	SETENA34	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt34. Read: 0 = interrupt34 disabled 1 = interrupt34 enabled. Reset type: CM.SYSRESETn
1	SETENA33	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt33. Read: 0 = interrupt33 disabled 1 = interrupt33 enabled. Reset type: CM.SYSRESETn
0	SETENA32	R/W1S	0h	Interrupt set-enable bits. Write: =0 = no effect 1 = enable interrupt32. Read: 0 = interrupt32 disabled 1 = interrupt32 enabled. Reset type: CM.SYSRESETn

41.12.9.3 NVIC_ICER0 Register (Offset = 180h) [Reset = 0h]

NVIC_ICER0 is shown in Figure 41-115 and described in Table 41-126.

Return to the Summary Table.

NVIC Interrupt Clear Enable Register 0

Figure 41-115. NVIC_ICER0 Register

31	30	29	28	27	26	25	24
CLRENA31	CLRENA30	CLRENA29	CLRENA28	CLRENA27	CLRENA26	CLRENA25	CLRENA24
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
23	22	21	20	19	18	17	16
CLRENA23	CLRENA22	CLRENA21	CLRENA20	CLRENA19	CLRENA18	CLRENA17	CLRENA16
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
15	14	13	12	11	10	9	8
CLRENA15	CLRENA14	CLRENA13	CLRENA12	CLRENA11	CLRENA10	CLRENA9	CLRENA8
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
CLRENA7	CLRENA6	CLRENA5	CLRENA4	CLRENA3	CLRENA2	CLRENA1	CLRENA0
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

Table 41-126. NVIC_ICER0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CLRENA31	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt31. Read: 0 = interrupt31 disabled 1 = interrupt31 enabled. Reset type: CM.SYSRESETn
30	CLRENA30	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt30. Read: 0 = interrupt30 disabled 1 = interrupt30 enabled. Reset type: CM.SYSRESETn
29	CLRENA29	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt29. Read: 0 = interrupt29 disabled 1 = interrupt29 enabled. Reset type: CM.SYSRESETn
28	CLRENA28	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt28. Read: 0 = interrupt28 disabled 1 = interrupt28 enabled. Reset type: CM.SYSRESETn

Table 41-126. NVIC_ICER0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	CLRENA27	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt27. Read: 0 = interrupt27 disabled 1 = interrupt27 enabled. Reset type: CM.SYSRESETn
26	CLRENA26	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt26. Read: 0 = interrupt26 disabled 1 = interrupt26 enabled. Reset type: CM.SYSRESETn
25	CLRENA25	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt25. Read: 0 = interrupt25 disabled 1 = interrupt25 enabled. Reset type: CM.SYSRESETn
24	CLRENA24	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt24. Read: 0 = interrupt24 disabled 1 = interrupt24 enabled. Reset type: CM.SYSRESETn
23	CLRENA23	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt23. Read: 0 = interrupt23 disabled 1 = interrupt23 enabled. Reset type: CM.SYSRESETn
22	CLRENA22	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt22. Read: 0 = interrupt22 disabled 1 = interrupt22 enabled. Reset type: CM.SYSRESETn
21	CLRENA21	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt21. Read: 0 = interrupt21 disabled 1 = interrupt21 enabled. Reset type: CM.SYSRESETn

Table 41-126. NVIC_ICER0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	CLRENA20	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt20. Read: 0 = interrupt20 disabled 1 = interrupt20 enabled. Reset type: CM.SYSRESETn
19	CLRENA19	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt19. Read: 0 = interrupt19 disabled 1 = interrupt19 enabled. Reset type: CM.SYSRESETn
18	CLRENA18	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt18. Read: 0 = interrupt18 disabled 1 = interrupt18 enabled. Reset type: CM.SYSRESETn
17	CLRENA17	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt17. Read: 0 = interrupt17 disabled 1 = interrupt17 enabled. Reset type: CM.SYSRESETn
16	CLRENA16	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt16. Read: 0 = interrupt16 disabled 1 = interrupt16 enabled. Reset type: CM.SYSRESETn
15	CLRENA15	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt15. Read: 0 = interrupt15 disabled 1 = interrupt15 enabled. Reset type: CM.SYSRESETn
14	CLRENA14	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt14. Read: 0 = interrupt14 disabled 1 = interrupt14 enabled. Reset type: CM.SYSRESETn

Table 41-126. NVIC_ICER0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	CLRENA13	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt13. Read: 0 = interrupt13 disabled 1 = interrupt13 enabled. Reset type: CM.SYSRESETn
12	CLRENA12	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt12. Read: 0 = interrupt12 disabled 1 = interrupt12 enabled. Reset type: CM.SYSRESETn
11	CLRENA11	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt11. Read: 0 = interrupt11 disabled 1 = interrupt11 enabled. Reset type: CM.SYSRESETn
10	CLRENA10	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt10. Read: 0 = interrupt10 disabled 1 = interrupt10 enabled. Reset type: CM.SYSRESETn
9	CLRENA9	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt9. Read: 0 = interrupt9 disabled 1 = interrupt9 enabled. Reset type: CM.SYSRESETn
8	CLRENA8	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt8. Read: 0 = interrupt8 disabled 1 = interrupt8 enabled. Reset type: CM.SYSRESETn
7	CLRENA7	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt7. Read: 0 = interrupt7 disabled 1 = interrupt7 enabled. Reset type: CM.SYSRESETn

Table 41-126. NVIC_ICER0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	CLRENA6	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt6. Read: 0 = interrupt6 disabled 1 = interrupt6 enabled. Reset type: CM.SYSRESETn
5	CLRENA5	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt5. Read: 0 = interrupt5 disabled 1 = interrupt5 enabled. Reset type: CM.SYSRESETn
4	CLRENA4	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt4. Read: 0 = interrupt4 disabled 1 = interrupt4 enabled. Reset type: CM.SYSRESETn
3	CLRENA3	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt3. Read: 0 = interrupt3 disabled 1 = interrupt3 enabled. Reset type: CM.SYSRESETn
2	CLRENA2	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt2. Read: 0 = interrupt2 disabled 1 = interrupt2 enabled. Reset type: CM.SYSRESETn
1	CLRENA1	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt1. Read: 0 = interrupt1 disabled 1 = interrupt1 enabled. Reset type: CM.SYSRESETn
0	CLRENA0	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt0. Read: 0 = interrupt0 disabled 1 = interrupt0 enabled. Reset type: CM.SYSRESETn

41.12.9.4 NVIC_ICER1 Register (Offset = 184h) [Reset = 0h]

NVIC_ICER1 is shown in Figure 41-116 and described in Table 41-127.

Return to the Summary Table.

NVIC Interrupt Clear Enable Register 1

Figure 41-116. NVIC_ICER1 Register

31	30	29	28	27	26	25	24
CLRENA63	CLRENA62	CLRENA61	CLRENA60	CLRENA59	CLRENA58	CLRENA57	CLRENA56
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
23	22	21	20	19	18	17	16
CLRENA55	CLRENA54	CLRENA53	CLRENA52	CLRENA51	CLRENA50	CLRENA49	CLRENA48
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
15	14	13	12	11	10	9	8
CLRENA47	CLRENA46	CLRENA45	CLRENA44	CLRENA43	CLRENA42	CLRENA41	CLRENA40
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
CLRENA39	CLRENA38	CLRENA37	CLRENA36	CLRENA35	CLRENA34	CLRENA33	CLRENA32
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

Table 41-127. NVIC_ICER1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CLRENA63	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt63. Read: 0 = interrupt63 disabled 1 = interrupt63 enabled. Reset type: CM.SYSRESETn
30	CLRENA62	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt62. Read: 0 = interrupt62 disabled 1 = interrupt62 enabled. Reset type: CM.SYSRESETn
29	CLRENA61	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt61. Read: 0 = interrupt61 disabled 1 = interrupt61 enabled. Reset type: CM.SYSRESETn
28	CLRENA60	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt60. Read: 0 = interrupt60 disabled 1 = interrupt60 enabled. Reset type: CM.SYSRESETn

Table 41-127. NVIC_ICER1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	CLRENA59	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt59. Read: 0 = interrupt59 disabled 1 = interrupt59 enabled. Reset type: CM.SYSRESETn
26	CLRENA58	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt58. Read: 0 = interrupt58 disabled 1 = interrupt58 enabled. Reset type: CM.SYSRESETn
25	CLRENA57	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt57. Read: 0 = interrupt57 disabled 1 = interrupt57 enabled. Reset type: CM.SYSRESETn
24	CLRENA56	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt56. Read: 0 = interrupt56 disabled 1 = interrupt56 enabled. Reset type: CM.SYSRESETn
23	CLRENA55	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt55. Read: 0 = interrupt55 disabled 1 = interrupt55 enabled. Reset type: CM.SYSRESETn
22	CLRENA54	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt54. Read: 0 = interrupt54 disabled 1 = interrupt54 enabled. Reset type: CM.SYSRESETn
21	CLRENA53	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt53. Read: 0 = interrupt53 disabled 1 = interrupt53 enabled. Reset type: CM.SYSRESETn

Table 41-127. NVIC_ICER1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	CLRENA52	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt52. Read: 0 = interrupt52 disabled 1 = interrupt52 enabled. Reset type: CM.SYSRESETn
19	CLRENA51	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt51. Read: 0 = interrupt51 disabled 1 = interrupt51 enabled. Reset type: CM.SYSRESETn
18	CLRENA50	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt50. Read: 0 = interrupt50 disabled 1 = interrupt50 enabled. Reset type: CM.SYSRESETn
17	CLRENA49	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt49. Read: 0 = interrupt49 disabled 1 = interrupt49 enabled. Reset type: CM.SYSRESETn
16	CLRENA48	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt48. Read: 0 = interrupt48 disabled 1 = interrupt48 enabled. Reset type: CM.SYSRESETn
15	CLRENA47	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt47. Read: 0 = interrupt47 disabled 1 = interrupt47 enabled. Reset type: CM.SYSRESETn
14	CLRENA46	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt46. Read: 0 = interrupt46 disabled 1 = interrupt46 enabled. Reset type: CM.SYSRESETn

Table 41-127. NVIC_ICER1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	CLRENA45	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt45. Read: 0 = interrupt45 disabled 1 = interrupt45 enabled. Reset type: CM.SYSRESETn
12	CLRENA44	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt44. Read: 0 = interrupt44 disabled 1 = interrupt44 enabled. Reset type: CM.SYSRESETn
11	CLRENA43	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt43. Read: 0 = interrupt43 disabled 1 = interrupt43 enabled. Reset type: CM.SYSRESETn
10	CLRENA42	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt42. Read: 0 = interrupt42 disabled 1 = interrupt42 enabled. Reset type: CM.SYSRESETn
9	CLRENA41	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt41. Read: 0 = interrupt41 disabled 1 = interrupt41 enabled. Reset type: CM.SYSRESETn
8	CLRENA40	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt40. Read: 0 = interrupt40 disabled 1 = interrupt40 enabled. Reset type: CM.SYSRESETn
7	CLRENA39	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt39. Read: 0 = interrupt39 disabled 1 = interrupt39 enabled. Reset type: CM.SYSRESETn

Table 41-127. NVIC_ICER1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	CLRENA38	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt38. Read: 0 = interrupt38 disabled 1 = interrupt38 enabled. Reset type: CM.SYSRESETn
5	CLRENA37	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt37. Read: 0 = interrupt37 disabled 1 = interrupt37 enabled. Reset type: CM.SYSRESETn
4	CLRENA36	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt36. Read: 0 = interrupt36 disabled 1 = interrupt36 enabled. Reset type: CM.SYSRESETn
3	CLRENA35	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt35. Read: 0 = interrupt35 disabled 1 = interrupt35 enabled. Reset type: CM.SYSRESETn
2	CLRENA34	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt34. Read: 0 = interrupt34 disabled 1 = interrupt34 enabled. Reset type: CM.SYSRESETn
1	CLRENA33	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt33. Read: 0 = interrupt33 disabled 1 = interrupt33 enabled. Reset type: CM.SYSRESETn
0	CLRENA32	R/W1S	0h	Interrupt clear enable bits. Write: =0 = no effect 1 = disable interrupt32. Read: 0 = interrupt32 disabled 1 = interrupt32 enabled. Reset type: CM.SYSRESETn

41.12.9.5 NVIC_ISPR0 Register (Offset = 200h) [Reset = 0h]

NVIC_ISPR0 is shown in Figure 41-117 and described in Table 41-128.

Return to the Summary Table.

NVIC Interrupt Set Pending Register 0

Figure 41-117. NVIC_ISPR0 Register

31	30	29	28	27	26	25	24
SETPEND31	SETPEND30	SETPEND29	SETPEND28	SETPEND27	SETPEND26	SETPEND25	SETPEND24
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
23	22	21	20	19	18	17	16
SETPEND23	SETPEND22	SETPEND21	SETPEND20	SETPEND19	SETPEND18	SETPEND17	SETPEND16
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
15	14	13	12	11	10	9	8
SETPEND15	SETPEND14	SETPEND13	SETPEND12	SETPEND11	SETPEND10	SETPEND9	SETPEND8
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
SETPEND7	SETPEND6	SETPEND5	SETPEND4	SETPEND3	SETPEND2	SETPEND1	SETPEND0
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

Table 41-128. NVIC_ISPR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SETPEND31	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 31 state to pending. Read: 0 = interrupt31 is not pending 1 = interrupt31 is pending. Reset type: CM.SYSRESETn
30	SETPEND30	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 30 state to pending. Read: 0 = interrupt30 is not pending 1 = interrupt30 is pending. Reset type: CM.SYSRESETn
29	SETPEND29	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 29 state to pending. Read: 0 = interrupt29 is not pending 1 = interrupt29 is pending. Reset type: CM.SYSRESETn
28	SETPEND28	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 28 state to pending. Read: 0 = interrupt28 is not pending 1 = interrupt28 is pending. Reset type: CM.SYSRESETn

Table 41-128. NVIC_ISPR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	SETPEND27	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 27 state to pending. Read: 0 = interrupt27 is not pending 1 = interrupt27 is pending. Reset type: CM.SYSRESETn
26	SETPEND26	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 26 state to pending. Read: 0 = interrupt26 is not pending 1 = interrupt26 is pending. Reset type: CM.SYSRESETn
25	SETPEND25	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 25 state to pending. Read: 0 = interrupt25 is not pending 1 = interrupt25 is pending. Reset type: CM.SYSRESETn
24	SETPEND24	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 24 state to pending. Read: 0 = interrupt24 is not pending 1 = interrupt24 is pending. Reset type: CM.SYSRESETn
23	SETPEND23	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 23 state to pending. Read: 0 = interrupt23 is not pending 1 = interrupt23 is pending. Reset type: CM.SYSRESETn
22	SETPEND22	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 22 state to pending. Read: 0 = interrupt22 is not pending 1 = interrupt22 is pending. Reset type: CM.SYSRESETn
21	SETPEND21	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 21 state to pending. Read: 0 = interrupt21 is not pending 1 = interrupt21 is pending. Reset type: CM.SYSRESETn

Table 41-128. NVIC_ISPR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	SETPEND20	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 20 state to pending. Read: 0 = interrupt20 is not pending 1 = interrupt20 is pending. Reset type: CM.SYSRESETn
19	SETPEND19	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 19 state to pending. Read: 0 = interrupt19 is not pending 1 = interrupt19 is pending. Reset type: CM.SYSRESETn
18	SETPEND18	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 18 state to pending. Read: 0 = interrupt18 is not pending 1 = interrupt18 is pending. Reset type: CM.SYSRESETn
17	SETPEND17	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 17 state to pending. Read: 0 = interrupt17 is not pending 1 = interrupt17 is pending. Reset type: CM.SYSRESETn
16	SETPEND16	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 16 state to pending. Read: 0 = interrupt16 is not pending 1 = interrupt16 is pending. Reset type: CM.SYSRESETn
15	SETPEND15	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 15 state to pending. Read: 0 = interrupt15 is not pending 1 = interrupt15 is pending. Reset type: CM.SYSRESETn
14	SETPEND14	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 14 state to pending. Read: 0 = interrupt14 is not pending 1 = interrupt14 is pending. Reset type: CM.SYSRESETn

Table 41-128. NVIC_ISPR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	SETPEND13	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 13 state to pending. Read: 0 = interrupt13 is not pending 1 = interrupt13 is pending. Reset type: CM.SYSRESETn
12	SETPEND12	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 12 state to pending. Read: 0 = interrupt12 is not pending 1 = interrupt12 is pending. Reset type: CM.SYSRESETn
11	SETPEND11	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 11 state to pending. Read: 0 = interrupt11 is not pending 1 = interrupt11 is pending. Reset type: CM.SYSRESETn
10	SETPEND10	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 10 state to pending. Read: 0 = interrupt10 is not pending 1 = interrupt10 is pending. Reset type: CM.SYSRESETn
9	SETPEND9	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 9 state to pending. Read: 0 = interrupt9 is not pending 1 = interrupt9 is pending. Reset type: CM.SYSRESETn
8	SETPEND8	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 8 state to pending. Read: 0 = interrupt8 is not pending 1 = interrupt8 is pending. Reset type: CM.SYSRESETn
7	SETPEND7	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 7 state to pending. Read: 0 = interrupt7 is not pending 1 = interrupt7 is pending. Reset type: CM.SYSRESETn

Table 41-128. NVIC_ISPR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	SETPEND6	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 6 state to pending. Read: 0 = interrupt6 is not pending 1 = interrupt6 is pending. Reset type: CM.SYSRESETn
5	SETPEND5	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 5 state to pending. Read: 0 = interrupt5 is not pending 1 = interrupt5 is pending. Reset type: CM.SYSRESETn
4	SETPEND4	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 4 state to pending. Read: 0 = interrupt4 is not pending 1 = interrupt4 is pending. Reset type: CM.SYSRESETn
3	SETPEND3	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 3 state to pending. Read: 0 = interrupt3 is not pending 1 = interrupt3 is pending. Reset type: CM.SYSRESETn
2	SETPEND2	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 2 state to pending. Read: 0 = interrupt2 is not pending 1 = interrupt2 is pending. Reset type: CM.SYSRESETn
1	SETPEND1	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 1 state to pending. Read: 0 = interrupt1 is not pending 1 = interrupt1 is pending. Reset type: CM.SYSRESETn
0	SETPEND0	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 0 state to pending. Read: 0 = interrupt0 is not pending 1 = interrupt0 is pending. Reset type: CM.SYSRESETn

41.12.9.6 NVIC_ISPR1 Register (Offset = 204h) [Reset = 0h]

NVIC_ISPR1 is shown in Figure 41-118 and described in Table 41-129.

Return to the Summary Table.

NVIC Interrupt Set Pending Register 1

Figure 41-118. NVIC_ISPR1 Register

31	30	29	28	27	26	25	24
SETPEND63	SETPEND62	SETPEND61	SETPEND60	SETPEND59	SETPEND58	SETPEND57	SETPEND56
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
23	22	21	20	19	18	17	16
SETPEND55	SETPEND54	SETPEND53	SETPEND52	SETPEND51	SETPEND50	SETPEND49	SETPEND48
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
15	14	13	12	11	10	9	8
SETPEND47	SETPEND46	SETPEND45	SETPEND44	SETPEND43	SETPEND42	SETPEND41	SETPEND40
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
SETPEND39	SETPEND38	SETPEND37	SETPEND36	SETPEND35	SETPEND34	SETPEND33	SETPEND32
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

Table 41-129. NVIC_ISPR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SETPEND63	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 63 state to pending. Read: 0 = interrupt63 is not pending 1 = interrupt63 is pending. Reset type: CM.SYSRESETn
30	SETPEND62	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 62 state to pending. Read: 0 = interrupt62 is not pending 1 = interrupt62 is pending. Reset type: CM.SYSRESETn
29	SETPEND61	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 61 state to pending. Read: 0 = interrupt61 is not pending 1 = interrupt61 is pending. Reset type: CM.SYSRESETn
28	SETPEND60	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 60 state to pending. Read: 0 = interrupt60 is not pending 1 = interrupt60 is pending. Reset type: CM.SYSRESETn

Table 41-129. NVIC_ISPR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	SETPEND59	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 59 state to pending. Read: 0 = interrupt59 is not pending 1 = interrupt59 is pending. Reset type: CM.SYSRESETn
26	SETPEND58	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 58 state to pending. Read: 0 = interrupt58 is not pending 1 = interrupt58 is pending. Reset type: CM.SYSRESETn
25	SETPEND57	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 57 state to pending. Read: 0 = interrupt57 is not pending 1 = interrupt57 is pending. Reset type: CM.SYSRESETn
24	SETPEND56	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 56 state to pending. Read: 0 = interrupt56 is not pending 1 = interrupt56 is pending. Reset type: CM.SYSRESETn
23	SETPEND55	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 55 state to pending. Read: 0 = interrupt55 is not pending 1 = interrupt55 is pending. Reset type: CM.SYSRESETn
22	SETPEND54	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 54 state to pending. Read: 0 = interrupt54 is not pending 1 = interrupt54 is pending. Reset type: CM.SYSRESETn
21	SETPEND53	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 53 state to pending. Read: 0 = interrupt53 is not pending 1 = interrupt53 is pending. Reset type: CM.SYSRESETn

Table 41-129. NVIC_ISPR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	SETPEND52	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 52 state to pending. Read: 0 = interrupt52 is not pending 1 = interrupt52 is pending. Reset type: CM.SYSRESETn
19	SETPEND51	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 51 state to pending. Read: 0 = interrupt51 is not pending 1 = interrupt51 is pending. Reset type: CM.SYSRESETn
18	SETPEND50	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 50 state to pending. Read: 0 = interrupt50 is not pending 1 = interrupt50 is pending. Reset type: CM.SYSRESETn
17	SETPEND49	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 49 state to pending. Read: 0 = interrupt49 is not pending 1 = interrupt49 is pending. Reset type: CM.SYSRESETn
16	SETPEND48	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 48 state to pending. Read: 0 = interrupt48 is not pending 1 = interrupt48 is pending. Reset type: CM.SYSRESETn
15	SETPEND47	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 47 state to pending. Read: 0 = interrupt47 is not pending 1 = interrupt47 is pending. Reset type: CM.SYSRESETn
14	SETPEND46	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 46 state to pending. Read: 0 = interrupt46 is not pending 1 = interrupt46 is pending. Reset type: CM.SYSRESETn

Table 41-129. NVIC_ISPR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	SETPEND45	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 45 state to pending. Read: 0 = interrupt45 is not pending 1 = interrupt45 is pending. Reset type: CM.SYSRESETn
12	SETPEND44	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 44 state to pending. Read: 0 = interrupt44 is not pending 1 = interrupt44 is pending. Reset type: CM.SYSRESETn
11	SETPEND43	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 43 state to pending. Read: 0 = interrupt43 is not pending 1 = interrupt43 is pending. Reset type: CM.SYSRESETn
10	SETPEND42	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 42 state to pending. Read: 0 = interrupt42 is not pending 1 = interrupt42 is pending. Reset type: CM.SYSRESETn
9	SETPEND41	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 41 state to pending. Read: 0 = interrupt41 is not pending 1 = interrupt41 is pending. Reset type: CM.SYSRESETn
8	SETPEND40	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 40 state to pending. Read: 0 = interrupt40 is not pending 1 = interrupt40 is pending. Reset type: CM.SYSRESETn
7	SETPEND39	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 39 state to pending. Read: 0 = interrupt39 is not pending 1 = interrupt39 is pending. Reset type: CM.SYSRESETn

Table 41-129. NVIC_ISPR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	SETPEND38	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 38 state to pending. Read: 0 = interrupt38 is not pending 1 = interrupt38 is pending. Reset type: CM.SYSRESETn
5	SETPEND37	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 37 state to pending. Read: 0 = interrupt37 is not pending 1 = interrupt37 is pending. Reset type: CM.SYSRESETn
4	SETPEND36	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 36 state to pending. Read: 0 = interrupt36 is not pending 1 = interrupt36 is pending. Reset type: CM.SYSRESETn
3	SETPEND35	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 35 state to pending. Read: 0 = interrupt35 is not pending 1 = interrupt35 is pending. Reset type: CM.SYSRESETn
2	SETPEND34	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 34 state to pending. Read: 0 = interrupt34 is not pending 1 = interrupt34 is pending. Reset type: CM.SYSRESETn
1	SETPEND33	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 33 state to pending. Read: 0 = interrupt33 is not pending 1 = interrupt33 is pending. Reset type: CM.SYSRESETn
0	SETPEND32	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 32 state to pending. Read: 0 = interrupt32 is not pending 1 = interrupt32 is pending. Reset type: CM.SYSRESETn

41.12.9.7 NVIC_ISPR2 Register (Offset = 208h) [Reset = 0h]

NVIC_ISPR2 is shown in Figure 41-119 and described in Table 41-130.

Return to the Summary Table.

NVIC Interrupt Set Pending Register 2

Figure 41-119. NVIC_ISPR2 Register

31	30	29	28	27	26	25	24
SETPEND95	SETPEND94	SETPEND93	SETPEND92	SETPEND91	SETPEND90	SETPEND89	SETPEND88
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
23	22	21	20	19	18	17	16
SETPEND87	SETPEND86	SETPEND85	SETPEND84	SETPEND83	SETPEND82	SETPEND81	SETPEND80
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
15	14	13	12	11	10	9	8
SETPEND79	SETPEND78	SETPEND77	SETPEND76	SETPEND75	SETPEND74	SETPEND73	SETPEND72
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
SETPEND71	SETPEND70	SETPEND69	SETPEND68	SETPEND67	SETPEND66	SETPEND65	SETPEND64
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

Table 41-130. NVIC_ISPR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SETPEND95	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 95 state to pending. Read: 0 = interrupt95 is not pending 1 = interrupt95 is pending. Reset type: CM.SYSRESETn
30	SETPEND94	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 94 state to pending. Read: 0 = interrupt94 is not pending 1 = interrupt94 is pending. Reset type: CM.SYSRESETn
29	SETPEND93	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 93 state to pending. Read: 0 = interrupt93 is not pending 1 = interrupt93 is pending. Reset type: CM.SYSRESETn
28	SETPEND92	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 92 state to pending. Read: 0 = interrupt92 is not pending 1 = interrupt92 is pending. Reset type: CM.SYSRESETn

Table 41-130. NVIC_ISPR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	SETPEND91	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 91 state to pending. Read: 0 = interrupt91 is not pending 1 = interrupt91 is pending. Reset type: CM.SYSRESETn
26	SETPEND90	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 90 state to pending. Read: 0 = interrupt90 is not pending 1 = interrupt90 is pending. Reset type: CM.SYSRESETn
25	SETPEND89	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 89 state to pending. Read: 0 = interrupt89 is not pending 1 = interrupt89 is pending. Reset type: CM.SYSRESETn
24	SETPEND88	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 88 state to pending. Read: 0 = interrupt88 is not pending 1 = interrupt88 is pending. Reset type: CM.SYSRESETn
23	SETPEND87	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 87 state to pending. Read: 0 = interrupt87 is not pending 1 = interrupt87 is pending. Reset type: CM.SYSRESETn
22	SETPEND86	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 86 state to pending. Read: 0 = interrupt86 is not pending 1 = interrupt86 is pending. Reset type: CM.SYSRESETn
21	SETPEND85	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 85 state to pending. Read: 0 = interrupt85 is not pending 1 = interrupt85 is pending. Reset type: CM.SYSRESETn

Table 41-130. NVIC_ISPR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	SETPEND84	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 84 state to pending. Read: 0 = interrupt84 is not pending 1 = interrupt84 is pending. Reset type: CM.SYSRESETn
19	SETPEND83	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 83 state to pending. Read: 0 = interrupt83 is not pending 1 = interrupt83 is pending. Reset type: CM.SYSRESETn
18	SETPEND82	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 82 state to pending. Read: 0 = interrupt82 is not pending 1 = interrupt82 is pending. Reset type: CM.SYSRESETn
17	SETPEND81	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 81 state to pending. Read: 0 = interrupt81 is not pending 1 = interrupt81 is pending. Reset type: CM.SYSRESETn
16	SETPEND80	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 80 state to pending. Read: 0 = interrupt80 is not pending 1 = interrupt80 is pending. Reset type: CM.SYSRESETn
15	SETPEND79	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 79 state to pending. Read: 0 = interrupt79 is not pending 1 = interrupt79 is pending. Reset type: CM.SYSRESETn
14	SETPEND78	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 78 state to pending. Read: 0 = interrupt78 is not pending 1 = interrupt78 is pending. Reset type: CM.SYSRESETn

Table 41-130. NVIC_ISPR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	SETPEND77	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 77 state to pending. Read: 0 = interrupt77 is not pending 1 = interrupt77 is pending. Reset type: CM.SYSRESETn
12	SETPEND76	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 76 state to pending. Read: 0 = interrupt76 is not pending 1 = interrupt76 is pending. Reset type: CM.SYSRESETn
11	SETPEND75	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 75 state to pending. Read: 0 = interrupt75 is not pending 1 = interrupt75 is pending. Reset type: CM.SYSRESETn
10	SETPEND74	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 74 state to pending. Read: 0 = interrupt74 is not pending 1 = interrupt74 is pending. Reset type: CM.SYSRESETn
9	SETPEND73	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 73 state to pending. Read: 0 = interrupt73 is not pending 1 = interrupt73 is pending. Reset type: CM.SYSRESETn
8	SETPEND72	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 72 state to pending. Read: 0 = interrupt72 is not pending 1 = interrupt72 is pending. Reset type: CM.SYSRESETn
7	SETPEND71	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 71 state to pending. Read: 0 = interrupt71 is not pending 1 = interrupt71 is pending. Reset type: CM.SYSRESETn

Table 41-130. NVIC_ISPR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	SETPEND70	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 70 state to pending. Read: 0 = interrupt70 is not pending 1 = interrupt70 is pending. Reset type: CM.SYSRESETn
5	SETPEND69	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 69 state to pending. Read: 0 = interrupt69 is not pending 1 = interrupt69 is pending. Reset type: CM.SYSRESETn
4	SETPEND68	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 68 state to pending. Read: 0 = interrupt68 is not pending 1 = interrupt68 is pending. Reset type: CM.SYSRESETn
3	SETPEND67	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 67 state to pending. Read: 0 = interrupt67 is not pending 1 = interrupt67 is pending. Reset type: CM.SYSRESETn
2	SETPEND66	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 66 state to pending. Read: 0 = interrupt66 is not pending 1 = interrupt66 is pending. Reset type: CM.SYSRESETn
1	SETPEND65	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 65 state to pending. Read: 0 = interrupt65 is not pending 1 = interrupt65 is pending. Reset type: CM.SYSRESETn
0	SETPEND64	R/W1S	0h	Set interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 64 state to pending. Read: 0 = interrupt64 is not pending 1 = interrupt64 is pending. Reset type: CM.SYSRESETn

41.12.9.8 NVIC_ICPR0 Register (Offset = 280h) [Reset = 0h]

NVIC_ICPR0 is shown in Figure 41-120 and described in Table 41-131.

Return to the Summary Table.

NVIC Interrupt Clear Pending Register 0

Figure 41-120. NVIC_ICPR0 Register

31	30	29	28	27	26	25	24
CLRPEND31	CLRPEND30	CLRPEND29	CLRPEND28	CLRPEND27	CLRPEND26	CLRPEND25	CLRPEND24
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
23	22	21	20	19	18	17	16
CLRPEND23	CLRPEND22	CLRPEND21	CLRPEND20	CLRPEND19	CLRPEND18	CLRPEND17	CLRPEND16
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
15	14	13	12	11	10	9	8
CLRPEND15	CLRPEND14	CLRPEND13	CLRPEND12	CLRPEND11	CLRPEND10	CLRPEND9	CLRPEND8
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
CLRPEND7	CLRPEND6	CLRPEND5	CLRPEND4	CLRPEND3	CLRPEND2	CLRPEND1	CLRPEND0
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

Table 41-131. NVIC_ICPR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CLRPEND31	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 31 state to not pending. Read: 0 = interrupt31 is not pending 1 = interrupt31 is pending. Reset type: CM.SYSRESETn
30	CLRPEND30	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 30 state to not pending. Read: 0 = interrupt30 is not pending 1 = interrupt30 is pending. Reset type: CM.SYSRESETn
29	CLRPEND29	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 29 state to not pending. Read: 0 = interrupt29 is not pending 1 = interrupt29 is pending. Reset type: CM.SYSRESETn
28	CLRPEND28	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 28 state to not pending. Read: 0 = interrupt28 is not pending 1 = interrupt28 is pending. Reset type: CM.SYSRESETn

Table 41-131. NVIC_ICPR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	CLRPEND27	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 27 state to not pending. Read: 0 = interrupt27 is not pending 1 = interrupt27 is pending. Reset type: CM.SYSRESETn
26	CLRPEND26	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 26 state to not pending. Read: 0 = interrupt26 is not pending 1 = interrupt26 is pending. Reset type: CM.SYSRESETn
25	CLRPEND25	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 25 state to not pending. Read: 0 = interrupt25 is not pending 1 = interrupt25 is pending. Reset type: CM.SYSRESETn
24	CLRPEND24	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 24 state to not pending. Read: 0 = interrupt24 is not pending 1 = interrupt24 is pending. Reset type: CM.SYSRESETn
23	CLRPEND23	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 23 state to not pending. Read: 0 = interrupt23 is not pending 1 = interrupt23 is pending. Reset type: CM.SYSRESETn
22	CLRPEND22	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 22 state to not pending. Read: 0 = interrupt22 is not pending 1 = interrupt22 is pending. Reset type: CM.SYSRESETn
21	CLRPEND21	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 21 state to not pending. Read: 0 = interrupt21 is not pending 1 = interrupt21 is pending. Reset type: CM.SYSRESETn

Table 41-131. NVIC_ICPR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	CLRPEND20	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 20 state to not pending. Read: 0 = interrupt20 is not pending 1 = interrupt20 is pending. Reset type: CM.SYSRESETn
19	CLRPEND19	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 19 state to not pending. Read: 0 = interrupt19 is not pending 1 = interrupt19 is pending. Reset type: CM.SYSRESETn
18	CLRPEND18	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 18 state to not pending. Read: 0 = interrupt18 is not pending 1 = interrupt18 is pending. Reset type: CM.SYSRESETn
17	CLRPEND17	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 17 state to not pending. Read: 0 = interrupt17 is not pending 1 = interrupt17 is pending. Reset type: CM.SYSRESETn
16	CLRPEND16	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 16 state to not pending. Read: 0 = interrupt16 is not pending 1 = interrupt16 is pending. Reset type: CM.SYSRESETn
15	CLRPEND15	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 15 state to not pending. Read: 0 = interrupt15 is not pending 1 = interrupt15 is pending. Reset type: CM.SYSRESETn
14	CLRPEND14	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 14 state to not pending. Read: 0 = interrupt14 is not pending 1 = interrupt14 is pending. Reset type: CM.SYSRESETn

Table 41-131. NVIC_ICPR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	CLRPEND13	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 13 state to not pending. Read: 0 = interrupt13 is not pending 1 = interrupt13 is pending. Reset type: CM.SYSRESETn
12	CLRPEND12	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 12 state to not pending. Read: 0 = interrupt12 is not pending 1 = interrupt12 is pending. Reset type: CM.SYSRESETn
11	CLRPEND11	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 11 state to not pending. Read: 0 = interrupt11 is not pending 1 = interrupt11 is pending. Reset type: CM.SYSRESETn
10	CLRPEND10	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 10 state to not pending. Read: 0 = interrupt10 is not pending 1 = interrupt10 is pending. Reset type: CM.SYSRESETn
9	CLRPEND9	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 9 state to not pending. Read: 0 = interrupt9 is not pending 1 = interrupt9 is pending. Reset type: CM.SYSRESETn
8	CLRPEND8	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 8 state to not pending. Read: 0 = interrupt8 is not pending 1 = interrupt8 is pending. Reset type: CM.SYSRESETn
7	CLRPEND7	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 7 state to not pending. Read: 0 = interrupt7 is not pending 1 = interrupt7 is pending. Reset type: CM.SYSRESETn

Table 41-131. NVIC_ICPR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	CLRPEND6	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 6 state to not pending. Read: 0 = interrupt6 is not pending 1 = interrupt6 is pending. Reset type: CM.SYSRESETn
5	CLRPEND5	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 5 state to not pending. Read: 0 = interrupt5 is not pending 1 = interrupt5 is pending. Reset type: CM.SYSRESETn
4	CLRPEND4	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 4 state to not pending. Read: 0 = interrupt4 is not pending 1 = interrupt4 is pending. Reset type: CM.SYSRESETn
3	CLRPEND3	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 3 state to not pending. Read: 0 = interrupt3 is not pending 1 = interrupt3 is pending. Reset type: CM.SYSRESETn
2	CLRPEND2	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 2 state to not pending. Read: 0 = interrupt2 is not pending 1 = interrupt2 is pending. Reset type: CM.SYSRESETn
1	CLRPEND1	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 1 state to not pending. Read: 0 = interrupt1 is not pending 1 = interrupt1 is pending. Reset type: CM.SYSRESETn
0	CLRPEND0	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 0 state to not pending. Read: 0 = interrupt0 is not pending 1 = interrupt0 is pending. Reset type: CM.SYSRESETn

41.12.9.9 NVIC_ICPR1 Register (Offset = 284h) [Reset = 0h]

NVIC_ICPR1 is shown in Figure 41-121 and described in Table 41-132.

Return to the Summary Table.

NVIC Interrupt Clear Pending Register 1

Figure 41-121. NVIC_ICPR1 Register

31	30	29	28	27	26	25	24
CLRPEND63	CLRPEND62	CLRPEND61	CLRPEND60	CLRPEND59	CLRPEND58	CLRPEND57	CLRPEND56
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
23	22	21	20	19	18	17	16
CLRPEND55	CLRPEND54	CLRPEND53	CLRPEND52	CLRPEND51	CLRPEND50	CLRPEND49	CLRPEND48
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
15	14	13	12	11	10	9	8
CLRPEND47	CLRPEND46	CLRPEND45	CLRPEND44	CLRPEND43	CLRPEND42	CLRPEND41	CLRPEND40
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
CLRPEND39	CLRPEND38	CLRPEND37	CLRPEND36	CLRPEND35	CLRPEND34	CLRPEND33	CLRPEND32
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

Table 41-132. NVIC_ICPR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CLRPEND63	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 63 state to not pending. Read: 0 = interrupt63 is not pending 1 = interrupt63 is pending. Reset type: CM.SYSRESETn
30	CLRPEND62	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 62 state to not pending. Read: 0 = interrupt62 is not pending 1 = interrupt62 is pending. Reset type: CM.SYSRESETn
29	CLRPEND61	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 61 state to not pending. Read: 0 = interrupt61 is not pending 1 = interrupt61 is pending. Reset type: CM.SYSRESETn
28	CLRPEND60	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 60 state to not pending. Read: 0 = interrupt60 is not pending 1 = interrupt60 is pending. Reset type: CM.SYSRESETn

Table 41-132. NVIC_ICPR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	CLRPEND59	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 59 state to not pending. Read: 0 = interrupt59 is not pending 1 = interrupt59 is pending. Reset type: CM.SYSRESETn
26	CLRPEND58	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 58 state to not pending. Read: 0 = interrupt58 is not pending 1 = interrupt58 is pending. Reset type: CM.SYSRESETn
25	CLRPEND57	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 57 state to not pending. Read: 0 = interrupt57 is not pending 1 = interrupt57 is pending. Reset type: CM.SYSRESETn
24	CLRPEND56	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 56 state to not pending. Read: 0 = interrupt56 is not pending 1 = interrupt56 is pending. Reset type: CM.SYSRESETn
23	CLRPEND55	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 55 state to not pending. Read: 0 = interrupt55 is not pending 1 = interrupt55 is pending. Reset type: CM.SYSRESETn
22	CLRPEND54	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 54 state to not pending. Read: 0 = interrupt54 is not pending 1 = interrupt54 is pending. Reset type: CM.SYSRESETn
21	CLRPEND53	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 53 state to not pending. Read: 0 = interrupt53 is not pending 1 = interrupt53 is pending. Reset type: CM.SYSRESETn

Table 41-132. NVIC_ICPR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	CLRPEND52	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 52 state to not pending. Read: 0 = interrupt52 is not pending 1 = interrupt52 is pending. Reset type: CM.SYSRESETn
19	CLRPEND51	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 51 state to not pending. Read: 0 = interrupt51 is not pending 1 = interrupt51 is pending. Reset type: CM.SYSRESETn
18	CLRPEND50	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 50 state to not pending. Read: 0 = interrupt50 is not pending 1 = interrupt50 is pending. Reset type: CM.SYSRESETn
17	CLRPEND49	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 49 state to not pending. Read: 0 = interrupt49 is not pending 1 = interrupt49 is pending. Reset type: CM.SYSRESETn
16	CLRPEND48	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 48 state to not pending. Read: 0 = interrupt48 is not pending 1 = interrupt48 is pending. Reset type: CM.SYSRESETn
15	CLRPEND47	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 47 state to not pending. Read: 0 = interrupt47 is not pending 1 = interrupt47 is pending. Reset type: CM.SYSRESETn
14	CLRPEND46	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 46 state to not pending. Read: 0 = interrupt46 is not pending 1 = interrupt46 is pending. Reset type: CM.SYSRESETn

Table 41-132. NVIC_ICPR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	CLRPEND45	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 45 state to not pending. Read: 0 = interrupt45 is not pending 1 = interrupt45 is pending. Reset type: CM.SYSRESETn
12	CLRPEND44	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 44 state to not pending. Read: 0 = interrupt44 is not pending 1 = interrupt44 is pending. Reset type: CM.SYSRESETn
11	CLRPEND43	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 43 state to not pending. Read: 0 = interrupt43 is not pending 1 = interrupt43 is pending. Reset type: CM.SYSRESETn
10	CLRPEND42	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 42 state to not pending. Read: 0 = interrupt42 is not pending 1 = interrupt42 is pending. Reset type: CM.SYSRESETn
9	CLRPEND41	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 41 state to not pending. Read: 0 = interrupt41 is not pending 1 = interrupt41 is pending. Reset type: CM.SYSRESETn
8	CLRPEND40	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 40 state to not pending. Read: 0 = interrupt40 is not pending 1 = interrupt40 is pending. Reset type: CM.SYSRESETn
7	CLRPEND39	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 39 state to not pending. Read: 0 = interrupt39 is not pending 1 = interrupt39 is pending. Reset type: CM.SYSRESETn

Table 41-132. NVIC_ICPR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	CLRPEND38	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 38 state to not pending. Read: 0 = interrupt38 is not pending 1 = interrupt38 is pending. Reset type: CM.SYSRESETn
5	CLRPEND37	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 37 state to not pending. Read: 0 = interrupt37 is not pending 1 = interrupt37 is pending. Reset type: CM.SYSRESETn
4	CLRPEND36	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 36 state to not pending. Read: 0 = interrupt36 is not pending 1 = interrupt36 is pending. Reset type: CM.SYSRESETn
3	CLRPEND35	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 35 state to not pending. Read: 0 = interrupt35 is not pending 1 = interrupt35 is pending. Reset type: CM.SYSRESETn
2	CLRPEND34	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 34 state to not pending. Read: 0 = interrupt34 is not pending 1 = interrupt34 is pending. Reset type: CM.SYSRESETn
1	CLRPEND33	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 33 state to not pending. Read: 0 = interrupt33 is not pending 1 = interrupt33 is pending. Reset type: CM.SYSRESETn
0	CLRPEND32	R/W1S	0h	Clear interrupt pending bits. Write: =0 = no effect 1 = changes interrupt 32 state to not pending. Read: 0 = interrupt32 is not pending 1 = interrupt32 is pending. Reset type: CM.SYSRESETn

41.12.9.10 NVIC_IABR0 Register (Offset = 300h) [Reset = 0h]

NVIC_IABR0 is shown in Figure 41-122 and described in Table 41-133.

Return to the Summary Table.

NVIC Interrupt Active Bit Register 0

Figure 41-122. NVIC_IABR0 Register

31	30	29	28	27	26	25	24
ACTIVE31	ACTIVE30	ACTIVE29	ACTIVE28	ACTIVE27	ACTIVE26	ACTIVE25	ACTIVE24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
ACTIVE23	ACTIVE22	ACTIVE21	ACTIVE20	ACTIVE19	ACTIVE18	ACTIVE17	ACTIVE16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
ACTIVE15	ACTIVE14	ACTIVE13	ACTIVE12	ACTIVE11	ACTIVE10	ACTIVE9	ACTIVE8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
ACTIVE7	ACTIVE6	ACTIVE5	ACTIVE4	ACTIVE3	ACTIVE2	ACTIVE1	ACTIVE0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 41-133. NVIC_IABR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ACTIVE31	R	0h	Active interrupt bits. 0 = interrupt31 is not active. 1 = interrupt31 is active. Reset type: CM.SYSRESETn
30	ACTIVE30	R	0h	Active interrupt bits. 0 = interrupt30 is not active. 1 = interrupt30 is active. Reset type: CM.SYSRESETn
29	ACTIVE29	R	0h	Active interrupt bits. 0 = interrupt29 is not active. 1 = interrupt29 is active. Reset type: CM.SYSRESETn
28	ACTIVE28	R	0h	Active interrupt bits. 0 = interrupt28 is not active. 1 = interrupt28 is active. Reset type: CM.SYSRESETn
27	ACTIVE27	R	0h	Active interrupt bits. 0 = interrupt27 is not active. 1 = interrupt27 is active. Reset type: CM.SYSRESETn
26	ACTIVE26	R	0h	Active interrupt bits. 0 = interrupt26 is not active. 1 = interrupt26 is active. Reset type: CM.SYSRESETn
25	ACTIVE25	R	0h	Active interrupt bits. 0 = interrupt25 is not active. 1 = interrupt25 is active. Reset type: CM.SYSRESETn
24	ACTIVE24	R	0h	Active interrupt bits. 0 = interrupt24 is not active. 1 = interrupt24 is active. Reset type: CM.SYSRESETn

Table 41-133. NVIC_IABR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	ACTIVE23	R	0h	Active interrupt bits. 0 = interrupt23 is not active. 1 = interrupt23 is active. Reset type: CM.SYSRESETn
22	ACTIVE22	R	0h	Active interrupt bits. 0 = interrupt22 is not active. 1 = interrupt22 is active. Reset type: CM.SYSRESETn
21	ACTIVE21	R	0h	Active interrupt bits. 0 = interrupt21 is not active. 1 = interrupt21 is active. Reset type: CM.SYSRESETn
20	ACTIVE20	R	0h	Active interrupt bits. 0 = interrupt20 is not active. 1 = interrupt20 is active. Reset type: CM.SYSRESETn
19	ACTIVE19	R	0h	Active interrupt bits. 0 = interrupt19 is not active. 1 = interrupt19 is active. Reset type: CM.SYSRESETn
18	ACTIVE18	R	0h	Active interrupt bits. 0 = interrupt18 is not active. 1 = interrupt18 is active. Reset type: CM.SYSRESETn
17	ACTIVE17	R	0h	Active interrupt bits. 0 = interrupt17 is not active. 1 = interrupt17 is active. Reset type: CM.SYSRESETn
16	ACTIVE16	R	0h	Active interrupt bits. 0 = interrupt16 is not active. 1 = interrupt16 is active. Reset type: CM.SYSRESETn
15	ACTIVE15	R	0h	Active interrupt bits. 0 = interrupt15 is not active. 1 = interrupt15 is active. Reset type: CM.SYSRESETn
14	ACTIVE14	R	0h	Active interrupt bits. 0 = interrupt14 is not active. 1 = interrupt14 is active. Reset type: CM.SYSRESETn
13	ACTIVE13	R	0h	Active interrupt bits. 0 = interrupt13 is not active. 1 = interrupt13 is active. Reset type: CM.SYSRESETn
12	ACTIVE12	R	0h	Active interrupt bits. 0 = interrupt12 is not active. 1 = interrupt12 is active. Reset type: CM.SYSRESETn
11	ACTIVE11	R	0h	Active interrupt bits. 0 = interrupt11 is not active. 1 = interrupt11 is active. Reset type: CM.SYSRESETn
10	ACTIVE10	R	0h	Active interrupt bits. 0 = interrupt10 is not active. 1 = interrupt10 is active. Reset type: CM.SYSRESETn

Table 41-133. NVIC_IABR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	ACTIVE9	R	0h	Active interrupt bits. 0 = interrupt9 is not active. 1 = interrupt9 is active. Reset type: CM.SYSRESETn
8	ACTIVE8	R	0h	Active interrupt bits. 0 = interrupt8 is not active. 1 = interrupt8 is active. Reset type: CM.SYSRESETn
7	ACTIVE7	R	0h	Active interrupt bits. 0 = interrupt7 is not active. 1 = interrupt7 is active. Reset type: CM.SYSRESETn
6	ACTIVE6	R	0h	Active interrupt bits. 0 = interrupt6 is not active. 1 = interrupt6 is active. Reset type: CM.SYSRESETn
5	ACTIVE5	R	0h	Active interrupt bits. 0 = interrupt5 is not active. 1 = interrupt5 is active. Reset type: CM.SYSRESETn
4	ACTIVE4	R	0h	Active interrupt bits. 0 = interrupt4 is not active. 1 = interrupt4 is active. Reset type: CM.SYSRESETn
3	ACTIVE3	R	0h	Active interrupt bits. 0 = interrupt3 is not active. 1 = interrupt3 is active. Reset type: CM.SYSRESETn
2	ACTIVE2	R	0h	Active interrupt bits. 0 = interrupt2 is not active. 1 = interrupt2 is active. Reset type: CM.SYSRESETn
1	ACTIVE1	R	0h	Active interrupt bits. 0 = interrupt1 is not active. 1 = interrupt1 is active. Reset type: CM.SYSRESETn
0	ACTIVE0	R	0h	Active interrupt bits. 0 = interrupt0 is not active. 1 = interrupt0 is active. Reset type: CM.SYSRESETn

41.12.9.11 NVIC_IABR1 Register (Offset = 304h) [Reset = 0h]

NVIC_IABR1 is shown in Figure 41-123 and described in Table 41-134.

Return to the Summary Table.

NVIC Interrupt Active Bit Register 1

Figure 41-123. NVIC_IABR1 Register

31	30	29	28	27	26	25	24
ACTIVE63	ACTIVE62	ACTIVE61	ACTIVE60	ACTIVE59	ACTIVE58	ACTIVE57	ACTIVE56
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
ACTIVE55	ACTIVE54	ACTIVE53	ACTIVE52	ACTIVE51	ACTIVE50	ACTIVE49	ACTIVE48
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
ACTIVE47	ACTIVE46	ACTIVE45	ACTIVE44	ACTIVE43	ACTIVE42	ACTIVE41	ACTIVE40
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
ACTIVE39	ACTIVE38	ACTIVE37	ACTIVE36	ACTIVE35	ACTIVE34	ACTIVE33	ACTIVE32
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 41-134. NVIC_IABR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ACTIVE63	R	0h	Active interrupt bits. 0 = interrupt63 is not active. 1 = interrupt63 is active. Reset type: CM.SYSRESETn
30	ACTIVE62	R	0h	Active interrupt bits. 0 = interrupt62 is not active. 1 = interrupt62 is active. Reset type: CM.SYSRESETn
29	ACTIVE61	R	0h	Active interrupt bits. 0 = interrupt61 is not active. 1 = interrupt61 is active. Reset type: CM.SYSRESETn
28	ACTIVE60	R	0h	Active interrupt bits. 0 = interrupt60 is not active. 1 = interrupt60 is active. Reset type: CM.SYSRESETn
27	ACTIVE59	R	0h	Active interrupt bits. 0 = interrupt59 is not active. 1 = interrupt59 is active. Reset type: CM.SYSRESETn
26	ACTIVE58	R	0h	Active interrupt bits. 0 = interrupt58 is not active. 1 = interrupt58 is active. Reset type: CM.SYSRESETn
25	ACTIVE57	R	0h	Active interrupt bits. 0 = interrupt57 is not active. 1 = interrupt57 is active. Reset type: CM.SYSRESETn
24	ACTIVE56	R	0h	Active interrupt bits. 0 = interrupt56 is not active. 1 = interrupt56 is active. Reset type: CM.SYSRESETn

Table 41-134. NVIC_IABR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	ACTIVE55	R	0h	Active interrupt bits. 0 = interrupt55 is not active. 1 = interrupt55 is active. Reset type: CM.SYSRESETn
22	ACTIVE54	R	0h	Active interrupt bits. 0 = interrupt54 is not active. 1 = interrupt54 is active. Reset type: CM.SYSRESETn
21	ACTIVE53	R	0h	Active interrupt bits. 0 = interrupt53 is not active. 1 = interrupt53 is active. Reset type: CM.SYSRESETn
20	ACTIVE52	R	0h	Active interrupt bits. 0 = interrupt52 is not active. 1 = interrupt52 is active. Reset type: CM.SYSRESETn
19	ACTIVE51	R	0h	Active interrupt bits. 0 = interrupt51 is not active. 1 = interrupt51 is active. Reset type: CM.SYSRESETn
18	ACTIVE50	R	0h	Active interrupt bits. 0 = interrupt50 is not active. 1 = interrupt50 is active. Reset type: CM.SYSRESETn
17	ACTIVE49	R	0h	Active interrupt bits. 0 = interrupt49 is not active. 1 = interrupt49 is active. Reset type: CM.SYSRESETn
16	ACTIVE48	R	0h	Active interrupt bits. 0 = interrupt48 is not active. 1 = interrupt48 is active. Reset type: CM.SYSRESETn
15	ACTIVE47	R	0h	Active interrupt bits. 0 = interrupt47 is not active. 1 = interrupt47 is active. Reset type: CM.SYSRESETn
14	ACTIVE46	R	0h	Active interrupt bits. 0 = interrupt46 is not active. 1 = interrupt46 is active. Reset type: CM.SYSRESETn
13	ACTIVE45	R	0h	Active interrupt bits. 0 = interrupt45 is not active. 1 = interrupt45 is active. Reset type: CM.SYSRESETn
12	ACTIVE44	R	0h	Active interrupt bits. 0 = interrupt44 is not active. 1 = interrupt44 is active. Reset type: CM.SYSRESETn
11	ACTIVE43	R	0h	Active interrupt bits. 0 = interrupt43 is not active. 1 = interrupt43 is active. Reset type: CM.SYSRESETn
10	ACTIVE42	R	0h	Active interrupt bits. 0 = interrupt42 is not active. 1 = interrupt42 is active. Reset type: CM.SYSRESETn

Table 41-134. NVIC_IABR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	ACTIVE41	R	0h	Active interrupt bits. 0 = interrupt41 is not active. 1 = interrupt41 is active. Reset type: CM.SYSRESETn
8	ACTIVE40	R	0h	Active interrupt bits. 0 = interrupt40 is not active. 1 = interrupt40 is active. Reset type: CM.SYSRESETn
7	ACTIVE39	R	0h	Active interrupt bits. 0 = interrupt39 is not active. 1 = interrupt39 is active. Reset type: CM.SYSRESETn
6	ACTIVE38	R	0h	Active interrupt bits. 0 = interrupt38 is not active. 1 = interrupt38 is active. Reset type: CM.SYSRESETn
5	ACTIVE37	R	0h	Active interrupt bits. 0 = interrupt37 is not active. 1 = interrupt37 is active. Reset type: CM.SYSRESETn
4	ACTIVE36	R	0h	Active interrupt bits. 0 = interrupt36 is not active. 1 = interrupt36 is active. Reset type: CM.SYSRESETn
3	ACTIVE35	R	0h	Active interrupt bits. 0 = interrupt35 is not active. 1 = interrupt35 is active. Reset type: CM.SYSRESETn
2	ACTIVE34	R	0h	Active interrupt bits. 0 = interrupt34 is not active. 1 = interrupt34 is active. Reset type: CM.SYSRESETn
1	ACTIVE33	R	0h	Active interrupt bits. 0 = interrupt33 is not active. 1 = interrupt33 is active. Reset type: CM.SYSRESETn
0	ACTIVE32	R	0h	Active interrupt bits. 0 = interrupt32 is not active. 1 = interrupt32 is active. Reset type: CM.SYSRESETn

41.12.9.12 NVIC_IPR0 Register (Offset = 400h) [Reset = 0h]

NVIC_IPR0 is shown in Figure 41-124 and described in Table 41-135.

Return to the Summary Table.

NVIC Interrupt Priority Register 0

Figure 41-124. NVIC_IPR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_3			RESERVED					PRI_2			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_1			RESERVED					PRI_0			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				

Table 41-135. NVIC_IPR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	PRI_3	R/W	0h	Priority of interrupt 3. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
28-24	RESERVED	R	0h	Reserved
23-21	PRI_2	R/W	0h	Priority of interrupt 2. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
20-16	RESERVED	R	0h	Reserved
15-13	PRI_1	R/W	0h	Priority of interrupt 1. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
12-8	RESERVED	R	0h	Reserved
7-5	PRI_0	R/W	0h	Priority of interrupt 0. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
4-0	RESERVED	R	0h	Reserved

41.12.9.13 NVIC_IPR1 Register (Offset = 404h) [Reset = 0h]

NVIC_IPR1 is shown in Figure 41-125 and described in Table 41-136.

Return to the Summary Table.

NVIC Interrupt Priority Register 1

Figure 41-125. NVIC_IPR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_7			RESERVED					PRI_6			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_5			RESERVED					PRI_4			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				

Table 41-136. NVIC_IPR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	PRI_7	R/W	0h	Priority of interrupt 7. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
28-24	RESERVED	R	0h	Reserved
23-21	PRI_6	R/W	0h	Priority of interrupt 6. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
20-16	RESERVED	R	0h	Reserved
15-13	PRI_5	R/W	0h	Priority of interrupt 5. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
12-8	RESERVED	R	0h	Reserved
7-5	PRI_4	R/W	0h	Priority of interrupt 4. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
4-0	RESERVED	R	0h	Reserved

41.12.9.14 NVIC_IPR2 Register (Offset = 408h) [Reset = 0h]

NVIC_IPR2 is shown in Figure 41-126 and described in Table 41-137.

Return to the Summary Table.

NVIC Interrupt Priority Register 2

Figure 41-126. NVIC_IPR2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_11			RESERVED					PRI_10			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_9			RESERVED					PRI_8			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				

Table 41-137. NVIC_IPR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	PRI_11	R/W	0h	Priority of interrupt 11. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
28-24	RESERVED	R	0h	Reserved
23-21	PRI_10	R/W	0h	Priority of interrupt 10. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
20-16	RESERVED	R	0h	Reserved
15-13	PRI_9	R/W	0h	Priority of interrupt 9. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
12-8	RESERVED	R	0h	Reserved
7-5	PRI_8	R/W	0h	Priority of interrupt 8. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
4-0	RESERVED	R	0h	Reserved

41.12.9.15 NVIC_IPR3 Register (Offset = 40Ch) [Reset = 0h]

NVIC_IPR3 is shown in Figure 41-127 and described in Table 41-138.

Return to the Summary Table.

NVIC Interrupt Priority Register 3

Figure 41-127. NVIC_IPR3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_15			RESERVED					PRI_14			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_13			RESERVED					PRI_12			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				

Table 41-138. NVIC_IPR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	PRI_15	R/W	0h	Priority of interrupt 15. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
28-24	RESERVED	R	0h	Reserved
23-21	PRI_14	R/W	0h	Priority of interrupt 14. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
20-16	RESERVED	R	0h	Reserved
15-13	PRI_13	R/W	0h	Priority of interrupt 13. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
12-8	RESERVED	R	0h	Reserved
7-5	PRI_12	R/W	0h	Priority of interrupt 12. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
4-0	RESERVED	R	0h	Reserved

41.12.9.16 NVIC_IPR4 Register (Offset = 410h) [Reset = 0h]

NVIC_IPR4 is shown in Figure 41-128 and described in Table 41-139.

Return to the Summary Table.

NVIC Interrupt Priority Register 4

Figure 41-128. NVIC_IPR4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_19			RESERVED					PRI_18			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_17			RESERVED					PRI_16			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				

Table 41-139. NVIC_IPR4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	PRI_19	R/W	0h	Priority of interrupt 19. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
28-24	RESERVED	R	0h	Reserved
23-21	PRI_18	R/W	0h	Priority of interrupt 18. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
20-16	RESERVED	R	0h	Reserved
15-13	PRI_17	R/W	0h	Priority of interrupt 17. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
12-8	RESERVED	R	0h	Reserved
7-5	PRI_16	R/W	0h	Priority of interrupt 16. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
4-0	RESERVED	R	0h	Reserved

41.12.9.17 NVIC_IPR5 Register (Offset = 414h) [Reset = 0h]

NVIC_IPR5 is shown in Figure 41-129 and described in Table 41-140.

Return to the Summary Table.

NVIC Interrupt Priority Register 5

Figure 41-129. NVIC_IPR5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_23			RESERVED					PRI_22			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_21			RESERVED					PRI_20			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				

Table 41-140. NVIC_IPR5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	PRI_23	R/W	0h	Priority of interrupt 23. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
28-24	RESERVED	R	0h	Reserved
23-21	PRI_22	R/W	0h	Priority of interrupt 22. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
20-16	RESERVED	R	0h	Reserved
15-13	PRI_21	R/W	0h	Priority of interrupt 21. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
12-8	RESERVED	R	0h	Reserved
7-5	PRI_20	R/W	0h	Priority of interrupt 20. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
4-0	RESERVED	R	0h	Reserved

41.12.9.18 NVIC_IPR6 Register (Offset = 418h) [Reset = 0h]

NVIC_IPR6 is shown in Figure 41-130 and described in Table 41-141.

Return to the Summary Table.

NVIC Interrupt Priority Register 6

Figure 41-130. NVIC_IPR6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_27			RESERVED					PRI_26			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_25			RESERVED					PRI_24			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				

Table 41-141. NVIC_IPR6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	PRI_27	R/W	0h	Priority of interrupt 27. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
28-24	RESERVED	R	0h	Reserved
23-21	PRI_26	R/W	0h	Priority of interrupt 26. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
20-16	RESERVED	R	0h	Reserved
15-13	PRI_25	R/W	0h	Priority of interrupt 25. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
12-8	RESERVED	R	0h	Reserved
7-5	PRI_24	R/W	0h	Priority of interrupt 24. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
4-0	RESERVED	R	0h	Reserved

41.12.9.19 NVIC_IPR7 Register (Offset = 41Ch) [Reset = 0h]

NVIC_IPR7 is shown in Figure 41-131 and described in Table 41-142.

Return to the Summary Table.

NVIC Interrupt Priority Register 7

Figure 41-131. NVIC_IPR7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_31			RESERVED					PRI_30			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_29			RESERVED					PRI_28			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				

Table 41-142. NVIC_IPR7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	PRI_31	R/W	0h	Priority of interrupt 31. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
28-24	RESERVED	R	0h	Reserved
23-21	PRI_30	R/W	0h	Priority of interrupt 30. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
20-16	RESERVED	R	0h	Reserved
15-13	PRI_29	R/W	0h	Priority of interrupt 29. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
12-8	RESERVED	R	0h	Reserved
7-5	PRI_28	R/W	0h	Priority of interrupt 28. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
4-0	RESERVED	R	0h	Reserved

41.12.9.20 NVIC_IPR8 Register (Offset = 420h) [Reset = 0h]

NVIC_IPR8 is shown in Figure 41-132 and described in Table 41-143.

Return to the Summary Table.

NVIC Interrupt Priority Register 8

Figure 41-132. NVIC_IPR8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_35			RESERVED					PRI_34			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_33			RESERVED					PRI_32			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				

Table 41-143. NVIC_IPR8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	PRI_35	R/W	0h	Priority of interrupt 35. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
28-24	RESERVED	R	0h	Reserved
23-21	PRI_34	R/W	0h	Priority of interrupt 34. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
20-16	RESERVED	R	0h	Reserved
15-13	PRI_33	R/W	0h	Priority of interrupt 33. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
12-8	RESERVED	R	0h	Reserved
7-5	PRI_32	R/W	0h	Priority of interrupt 32. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
4-0	RESERVED	R	0h	Reserved

41.12.9.21 NVIC_IPR9 Register (Offset = 424h) [Reset = 0h]

NVIC_IPR9 is shown in Figure 41-133 and described in Table 41-144.

Return to the Summary Table.

NVIC Interrupt Priority Register 9

Figure 41-133. NVIC_IPR9 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_39			RESERVED					PRI_38			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_37			RESERVED					PRI_36			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				

Table 41-144. NVIC_IPR9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	PRI_39	R/W	0h	Priority of interrupt 39. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
28-24	RESERVED	R	0h	Reserved
23-21	PRI_38	R/W	0h	Priority of interrupt 38. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
20-16	RESERVED	R	0h	Reserved
15-13	PRI_37	R/W	0h	Priority of interrupt 37. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
12-8	RESERVED	R	0h	Reserved
7-5	PRI_36	R/W	0h	Priority of interrupt 36. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
4-0	RESERVED	R	0h	Reserved

41.12.9.22 NVIC_IPR10 Register (Offset = 428h) [Reset = 0h]

NVIC_IPR10 is shown in Figure 41-134 and described in Table 41-145.

Return to the Summary Table.

NVIC Interrupt Priority Register 10

Figure 41-134. NVIC_IPR10 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_43			RESERVED					PRI_42			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_41			RESERVED					PRI_40			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				

Table 41-145. NVIC_IPR10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	PRI_43	R/W	0h	Priority of interrupt 43. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
28-24	RESERVED	R	0h	Reserved
23-21	PRI_42	R/W	0h	Priority of interrupt 42. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
20-16	RESERVED	R	0h	Reserved
15-13	PRI_41	R/W	0h	Priority of interrupt 41. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
12-8	RESERVED	R	0h	Reserved
7-5	PRI_40	R/W	0h	Priority of interrupt 40. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
4-0	RESERVED	R	0h	Reserved

41.12.9.23 NVIC_IPR11 Register (Offset = 42Ch) [Reset = 0h]

NVIC_IPR11 is shown in Figure 41-135 and described in Table 41-146.

Return to the Summary Table.

NVIC Interrupt Priority Register 11

Figure 41-135. NVIC_IPR11 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_47			RESERVED					PRI_46			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_45			RESERVED					PRI_44			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				

Table 41-146. NVIC_IPR11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	PRI_47	R/W	0h	Priority of interrupt 47. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
28-24	RESERVED	R	0h	Reserved
23-21	PRI_46	R/W	0h	Priority of interrupt 46. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
20-16	RESERVED	R	0h	Reserved
15-13	PRI_45	R/W	0h	Priority of interrupt 45. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
12-8	RESERVED	R	0h	Reserved
7-5	PRI_44	R/W	0h	Priority of interrupt 44. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
4-0	RESERVED	R	0h	Reserved

41.12.9.24 NVIC_IPR12 Register (Offset = 430h) [Reset = 0h]

NVIC_IPR12 is shown in Figure 41-136 and described in Table 41-147.

Return to the Summary Table.

NVIC Interrupt Priority Register 12

Figure 41-136. NVIC_IPR12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_51			RESERVED					PRI_50			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_49			RESERVED					PRI_48			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				

Table 41-147. NVIC_IPR12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	PRI_51	R/W	0h	Priority of interrupt 51. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
28-24	RESERVED	R	0h	Reserved
23-21	PRI_50	R/W	0h	Priority of interrupt 50. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
20-16	RESERVED	R	0h	Reserved
15-13	PRI_49	R/W	0h	Priority of interrupt 49. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
12-8	RESERVED	R	0h	Reserved
7-5	PRI_48	R/W	0h	Priority of interrupt 48. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
4-0	RESERVED	R	0h	Reserved

41.12.9.25 NVIC_IPR13 Register (Offset = 434h) [Reset = 0h]

NVIC_IPR13 is shown in Figure 41-137 and described in Table 41-148.

Return to the Summary Table.

NVIC Interrupt Priority Register 13

Figure 41-137. NVIC_IPR13 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_55			RESERVED					PRI_54			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_53			RESERVED					PRI_52			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				

Table 41-148. NVIC_IPR13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	PRI_55	R/W	0h	Priority of interrupt 55. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
28-24	RESERVED	R	0h	Reserved
23-21	PRI_54	R/W	0h	Priority of interrupt 54. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
20-16	RESERVED	R	0h	Reserved
15-13	PRI_53	R/W	0h	Priority of interrupt 53. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
12-8	RESERVED	R	0h	Reserved
7-5	PRI_52	R/W	0h	Priority of interrupt 52. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
4-0	RESERVED	R	0h	Reserved

41.12.9.26 NVIC_IPR14 Register (Offset = 438h) [Reset = 0h]

NVIC_IPR14 is shown in Figure 41-138 and described in Table 41-149.

Return to the Summary Table.

NVIC Interrupt Priority Register 14

Figure 41-138. NVIC_IPR14 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_59			RESERVED					PRI_58			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_57			RESERVED					PRI_56			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				

Table 41-149. NVIC_IPR14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	PRI_59	R/W	0h	Priority of interrupt 59. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
28-24	RESERVED	R	0h	Reserved
23-21	PRI_58	R/W	0h	Priority of interrupt 58. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
20-16	RESERVED	R	0h	Reserved
15-13	PRI_57	R/W	0h	Priority of interrupt 57. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
12-8	RESERVED	R	0h	Reserved
7-5	PRI_56	R/W	0h	Priority of interrupt 56. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
4-0	RESERVED	R	0h	Reserved

41.12.9.27 NVIC_IPR15 Register (Offset = 43Ch) [Reset = 0h]

NVIC_IPR15 is shown in Figure 41-139 and described in Table 41-150.

Return to the Summary Table.

NVIC Interrupt Priority Register 15

Figure 41-139. NVIC_IPR15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_63			RESERVED					PRI_62			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_61			RESERVED					PRI_60			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				

Table 41-150. NVIC_IPR15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	PRI_63	R/W	0h	Priority of interrupt 63. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
28-24	RESERVED	R	0h	Reserved
23-21	PRI_62	R/W	0h	Priority of interrupt 62. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
20-16	RESERVED	R	0h	Reserved
15-13	PRI_61	R/W	0h	Priority of interrupt 61. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
12-8	RESERVED	R	0h	Reserved
7-5	PRI_60	R/W	0h	Priority of interrupt 60. Each implementation-defined priority field can hold a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. Register priority value fields are eight bits wide, and non-implemented low-order bits read as zero and ignore writes. Reset type: CM.SYSRESETn
4-0	RESERVED	R	0h	Reserved

41.12.9.28 STIR Register (Offset = F00h) [Reset = 0h]

STIR is shown in Figure 41-140 and described in Table 41-151.

Return to the Summary Table.

Software Trigger Interrupt Register

Figure 41-140. STIR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																							INTID								
R-0h																							R/W-0h								

Table 41-151. STIR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8-0	INTID	R/W	0h	Interrupt ID of the interrupt to trigger, in the range 0-239. For example, a value of 0x03 specifies interrupt IRQ3. Reset type: CM.SYSRESETn

41.12.10 SCB Registers

Table 41-152 lists the memory-mapped registers for the SCB registers. All register offset addresses not listed in Table 41-152 should be considered as reserved locations and the register contents should not be modified.

Table 41-152. SCB Registers

Offset	Acronym	Register Name	Write Protection	Section
8h	ACTLR	Auxiliary Control Register		Go
D00h	CPUID	CPUID Base Register		Go
D04h	ICSR	Interrupt Control and State Register		Go
D08h	VTOR	Vector Table Offset Register		Go
D0Ch	AIRCR	Application Interrupt and Reset Control Register		Go
D10h	SCR	System Control Register		Go
D14h	CCR	Configuration and Control Register		Go
D18h	SHPR1	System Handler Priority Register 1		Go
D1Ch	SHPR2	System Handler Priority Register 2		Go
D20h	SHPR3	System Handler Priority Register 3		Go
D24h	SHCSRS	System Handler Control and State Register		Go
D28h	CFSR	Configurable Fault Status Register		Go
D2Ch	HFSR	HardFault Status Register		Go
D34h	MMFAR	MemManage Fault Address Register		Go
D38h	BFAR	BusFault Address Register		Go
D3Ch	AFSR	Auxiliary Fault Status Register		Go

Complex bit access types are encoded to fit into small table cells. Table 41-153 shows the codes that are used for access types in this section.

Table 41-153. SCB Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 41-153. SCB Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

41.12.10.1 ACTLR Register (Offset = 8h) [Reset = 0h]

ACTLR is shown in Figure 41-141 and described in Table 41-154.

Return to the Summary Table.

Auxiliary Control Register

Figure 41-141. ACTLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						RESERVED	DISFPCA
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED					DISFOLD	DISDEFWBUF	DISMCYCINT
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 41-154. ACTLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	RESERVED	R/W	0h	Reserved
8	DISFPCA	R/W	0h	Disables automatic update of CONTROL.FPCA. Reset type: CM.SYSRESETn
7-3	RESERVED	R	0h	Reserved
2	DISFOLD	R/W	0h	When set to 1, disables IT folding. see About IT folding for more information. Reset type: CM.SYSRESETn
1	DISDEFWBUF	R/W	0h	When set to 1, disables write buffer use during default memory map accesses. This causes all BusFaults to be precise BusFaults but decreases performance because any store to memory must complete,before the processor can execute the next instruction. Note This bit only affects write buffers implemented in the Cortex-M4 processor. Reset type: CM.SYSRESETn
0	DISMCYCINT	R/W	0h	When set to 1, disables interruption of load multiple and store multiple instructions. This increases the,interrupt latency of the processor because any LDM or STM must complete before the processor,can stack the current state and enter the interrupt handler. Reset type: CM.SYSRESETn

41.12.10.2 CPUID Register (Offset = D00h) [Reset = 410FC240h]

CPUID is shown in Figure 41-142 and described in Table 41-155.

Return to the Summary Table.

CPUID Base Register

Figure 41-142. CPUID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Implementer								Variant				Constant			
R-41h								R-0h				R-Fh			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PartNo												Revision			
R-C24h												R-0h			

Table 41-155. CPUID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	Implementer	R	41h	Implementer code: 0x41 = ARM Reset type: CM.SYSRESETn
23-20	Variant	R	0h	Variant number, the r value in the rnpn product revision identifier: 0x0 = Revision 0 Reset type: CM.SYSRESETn
19-16	Constant	R	Fh	Reads as 0xF Reset type: CM.SYSRESETn
15-4	PartNo	R	C24h	Part number of the processor: 0xC24 = Cortex-M4 Reset type: CM.SYSRESETn
3-0	Revision	R	0h	Revision number, the p value in the rnpn product revision identifier: 0x0 = Patch 0 Reset type: CM.SYSRESETn

41.12.10.3 ICSR Register (Offset = D04h) [Reset = 0h]

ICSR is shown in Figure 41-143 and described in Table 41-156.

Return to the Summary Table.

Interrupt Control and State Register

Figure 41-143. ICSR Register

31	30	29	28	27	26	25	24
NMIPENDSET	RESERVED		PENDSVSET	PENDSVCLR	PENDSTSET	PENDSTCLR	RESERVED
R/W-0h	R-0h		R/W-0h	R-0/W1S-0h	R/W-0h	R-0/W1S-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED	ISRPENDING	RESERVED				VECTPENDING	
R-0h	R-0h	R-0h				R-0h	
15	14	13	12	11	10	9	8
VECTPENDING				RETTOBASE	RESERVED		VECTACTIVE
R-0h				R-0h	R-0h		R-0h
7	6	5	4	3	2	1	0
VECTACTIVE							
R-0h							

Table 41-156. ICSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NMIPENDSET	R/W	0h	NMI set-pending bit. Write.. 0 = no effect 1 = changes NMI exception state to pending. Read.. 0 = NMI exception is not pending 1 = NMI exception is pending. Because NMI is the highest-priority exception, normally the processor enter the NMI exception handler as soon as it registers a write of 1 to this bit, and entering the handler clears this bit to 0. A read of this bit by the NMI exception handler returns 1 only if the NMI signal is reasserted while the processor is executing that handler. Reset type: CM.SYSRESETn
30-29	RESERVED	R	0h	Reserved
28	PENDSVSET	R/W	0h	PendSV set-pending bit. Write.. 0 = no effect 1 = changes PendSV exception state to pending. Read.. 0 = PendSV exception is not pending 1 = PendSV exception is pending. Writing 1 to this bit is the only way to set the PendSV exception state to pending. Reset type: CM.SYSRESETn
27	PENDSVCLR	R-0/W1S	0h	PendSV clear-pending bit. Write.. 0 = no effect 1 = removes the pending state from the PendSV exception. Reset type: CM.SYSRESETn

Table 41-156. ICSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	PENDSTSET	R/W	0h	SysTick exception set-pending bit. Write.. 0 = no effect 1 = changes SysTick exception state to pending. Read.. 0 = SysTick exception is not pending 1 = SysTick exception is pending. When you write to the ICSR, the effect is Unpredictable if you.. write 1 to the PENDSVSET bit and write 1 to the PENDSVCLR bit write 1 to the PENDSTSET bit and write 1 to the PENDSTCLR bit. Reset type: CM.SYSRESETn
25	PENDSTCLR	R-0/W1S	0h	SysTick exception clear-pending bit. Write.. 0 = no effect 1 = removes the pending state from the SysTick exception. This bit is WO. On a register read its value is Unknown. Reset type: CM.SYSRESETn
24	RESERVED	R	0h	Reserved
23	RESERVED	R	0h	Reserved
22	ISRPENDING	R	0h	Interrupt pending flag, excluding NMI and Faults.. 0 = interrupt not pending 1 = interrupt pending. Reset type: CM.SYSRESETn
21-18	RESERVED	R	0h	Reserved
17-12	VECTPENDING	R	0h	Indicates the exception number of the highest priority pending enabled exception: 0 = no pending exceptions Nonzero = the exception number of the highest priority pending enabled exception. The value indicated by this field includes the effect of the BASEPRI and FAULTMASK registers, but not any effect of the PRIMASK register. Reset type: CM.SYSRESETn
11	RETTOBASE	R	0h	Indicates whether there are preempted active exceptions.. 0 = there are preempted active exceptions to execute 1 = there are no active exceptions, or the currently-executing exception is the only active exception. Reset type: CM.SYSRESETn
10-9	RESERVED	R	0h	Reserved
8-0	VECTACTIVE	R	0h	Contains the active exception number: 0 = Thread mode Nonzero = The exception numbera of the currently active exception. Note Subtract 16 from this value to obtain the CMSIS IRQ number required to index into the Interrupt Clear-Enable, Set-Enable, Clear-Pending, Set-Pending, or Priority Register. Reset type: CM.SYSRESETn

41.12.10.4 VTOR Register (Offset = D08h) [Reset = 0h]

VTOR is shown in Figure 41-144 and described in Table 41-157.

Return to the Summary Table.

Vector Table Offset Register

Figure 41-144. VTOR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TBLOFF															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBLOFF									RESERVED						
R/W-0h									R-0h						

Table 41-157. VTOR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	TBLOFF	R/W	0h	Vector table base offset field. It contains bits[29:7] of the offset of the table base from the bottom of the memory map. Note Bit[29] determines whether the vector table is in the code or SRAM memory region: 0 = code 1 = SRAM. In implementations bit[29] is sometimes called the TBLBASE bit. Reset type: CM.SYSRESETn
6-0	RESERVED	R	0h	Reserved

41.12.10.5 AIRCR Register (Offset = D0Ch) [Reset = FA050000h]

AIRCR is shown in Figure 41-145 and described in Table 41-158.

Return to the Summary Table.

Application Interrupt and Reset Control Register

Figure 41-145. AIRCR Register

31	30	29	28	27	26	25	24
VECTKEY							
R/W-FA05h							
23	22	21	20	19	18	17	16
VECTKEY							
R/W-FA05h							
15	14	13	12	11	10	9	8
ENDIANNESS	RESERVED				PRIGROUP		
R-0h	R-0h				R/W-0h		
7	6	5	4	3	2	1	0
RESERVED					SYSRESETREQ	VECTCLRACTIVE	VECTRESET
R-0h					R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 41-158. AIRCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	VECTKEY	R/W	FA05h	Register key.. Reads as 0xFA05 On writes, write 0x5FA to VECTKEY, otherwise the write is ignored. Reset type: CM.SYSRESETn
15	ENDIANNESS	R	0h	Data endianness bit is implementation defined.. 0 = Little-endian 1 = Big-endian. Reset type: CM.SYSRESETn
14-11	RESERVED	R	0h	Reserved
10-8	PRIGROUP	R/W	0h	Interrupt priority grouping field is implementation defined. This field determines the split of group priority from subpriority. PRIGROUP Binary pointa Group priority bits Subpriority bits Group priorities Subpriorities 0b000 bxxxxxxx.y [7:1] [0] 128 2 0b001 bxxxxxx.yy [7:2] [1:0] 64 4 0b010 bxxxxx.yyy [7:3] [2:0] 32 8 0b011 bxxxx.yyyy [7:4] [3:0] 16 16 0b100 bxxx.yyyyy [7:5] [4:0] 8 32 0b101 bxx.yyyyyy [7:6] [5:0] 4 64 0b110 bx.yyyyyyy [7] [6:0] 2 128 0b111 b.yyyyyyyy None [7:0] 1 256 Reset type: CM.SYSRESETn
7-3	RESERVED	R	0h	Reserved

Table 41-158. AIRCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	SYSRESETREQ	R-0/W1S	0h	System reset request bit is implementation defined.. 0 = no system reset request 1 = asserts a signal to the outer system that requests a reset. This is intended to force a large system reset of all major components except for debug. This bit reads as 0. See you vendor documentation for more information about the use of this signal in your implementation. Reset type: CM.SYSRESETn
1	VECTCLRACTIVE	R-0/W1S	0h	Reserved for Debug use. This bit reads as 0. When writing to the register you must write 0 to this bit, otherwise behavior is Unpredictable. Reset type: CM.SYSRESETn
0	VECTRESET	R-0/W1S	0h	Reserved for Debug use. This bit reads as 0. When writing to the register you must write 0 to this bit, otherwise behavior is Unpredictable Reset type: CM.SYSRESETn

41.12.10.6 SCR Register (Offset = D10h) [Reset = 0h]

SCR is shown in Figure 41-146 and described in Table 41-159.

Return to the Summary Table.

System Control Register

Figure 41-146. SCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	RESERVED	RESERVED	SLEEPONEXIT	RESERVED
R-0h			R/W-0h	R-0h	R/W-0h	R/W-0h	R-0h

Table 41-159. SCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	SLEEPONEXIT	R/W	0h	Indicates sleep-on-exit when returning from Handler mode to Thread mode.. 0 = do not sleep when returning to Thread mode. 1 = enter sleep, or deep sleep, on return from an ISR. Setting this bit to 1 enables an interrupt driven application to avoid returning to an empty main application. Reset type: CM.SYSRESETn
0	RESERVED	R	0h	Reserved

41.12.10.7 CCR Register (Offset = D14h) [Reset = 200h]

CCR is shown in Figure 41-147 and described in Table 41-160.

Return to the Summary Table.

Configuration and Control Register

Figure 41-147. CCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						STKALIGN	BFHFNMIGN
R-0h						R/W-1h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED			DIV_0_TRP	UNALIGN_TRP	RESERVED	USERSETMPEND	NONBASETHRDENA
R-0h			R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h

Table 41-160. CCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	STKALIGN	R/W	1h	Indicates stack alignment on exception entry.. 0 = 4-byte aligned 1 = 8-byte aligned. On exception entry, the processor uses bit 9 of the stacked PSR to indicate the stack alignment. On return from the exception it uses this stacked bit to restore the correct stack alignment. Reset type: CM.SYSRESETn
8	BFHFNMIGN	R/W	0h	Enables handlers with priority -1 or -2 to ignore data BusFaults caused by load and store instructions. This applies to the hard fault, NMI, and FAULTMASK escalated handlers.. 0 = data bus faults caused by load and store instructions cause a lock-up 1 = handlers running at priority -1 and -2 ignore data bus faults caused by load and store instructions. Set this bit to 1 only when the handler and its data are in absolutely safe memory. The normal use of this bit is to probe system devices and bridges to detect control path problems and fix them. Reset type: CM.SYSRESETn
7-5	RESERVED	R	0h	Reserved
4	DIV_0_TRP	R/W	0h	Enables faulting or halting when the processor executes an SDIV or UDIV instruction with a divisor of 0.. 0 = do not trap divide by 0 1 = trap divide by 0. When this bit is set to 0, a divide by zero returns a quotient of 0. Reset type: CM.SYSRESETn

Table 41-160. CCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	UNALIGN_TRP	R/W	0h	Enables unaligned access traps.. 0 = do not trap unaligned halfword and word accesses 1 = trap unaligned halfword and word accesses. If this bit is set to 1, an unaligned access generates a UsageFault. Unaligned LDM, STM, LDRD, and STRD instructions always fault irrespective of whether UNALIGN_TRP is set to 1. Reset type: CM.SYSRESETn
2	RESERVED	R	0h	Reserved
1	USERSETMPEND	R/W	0h	Enables unprivileged software access to the STIR. 0 = disable 1 = enable. Reset type: CM.SYSRESETn
0	NONBASETHRDENA	R/W	0h	Indicates how the processor enters Thread mode.. 0 = processor can enter Thread mode only when no exception is active. 1 = processor can enter Thread mode from any level under the control of an EXC_RETURN. EXC_RETURN[31:0] Description 0xFFFFFFF1 Return to Handler mode, exception return uses non-floating-point state from the MSP and execution uses MSP after return. 0xFFFFFFF9 Return to Thread mode, exception return uses non-floating-point state from MSP and execution uses MSP after return. 0xFFFFFFFD Return to Thread mode, exception return uses non-floating-point state from the PSP and execution uses PSP after return. 0xFFFFFFE1 Return to Handler mode, exception return uses floating-point-state from MSP and execution uses MSP after return. 0xFFFFFFE9 Return to Thread mode, exception return uses floating-point state from MSP and execution uses MSP after return. 0xFFFFFFED Return to Thread mode, exception return uses floating-point state from PSP and execution uses PSP after return. Reset type: CM.SYSRESETn

41.12.10.8 SHPR1 Register (Offset = D18h) [Reset = 0h]

SHPR1 is shown in Figure 41-148 and described in Table 41-161.

Return to the Summary Table.

System Handler Priority Register 1

Figure 41-148. SHPR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								PRI_6			RESERVED				
R-0h								R/W-0h			R-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_5			RESERVED					PRI_4			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				

Table 41-161. SHPR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-21	PRI_6	R/W	0h	Priority of system handler 6, UsageFault Reset type: CM.SYSRESETn
20-16	RESERVED	R	0h	Reserved
15-13	PRI_5	R/W	0h	Priority of system handler 5, BusFault Reset type: CM.SYSRESETn
12-8	RESERVED	R	0h	Reserved
7-5	PRI_4	R/W	0h	Priority of system handler 4, MemManage Reset type: CM.SYSRESETn
4-0	RESERVED	R	0h	Reserved

41.12.10.9 SHPR2 Register (Offset = D1Ch) [Reset = 0h]

SHPR2 is shown in Figure 41-149 and described in Table 41-162.

Return to the Summary Table.

System Handler Priority Register 2

Figure 41-149. SHPR2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_11			RESERVED												
R/W-0h			R-0h												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															
R-0h															

Table 41-162. SHPR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	PRI_11	R/W	0h	Priority of system handler 11, SVCall Reset type: CM.SYSRESETn
28-0	RESERVED	R	0h	Reserved

41.12.10.10 SHPR3 Register (Offset = D20h) [Reset = 0h]

SHPR3 is shown in Figure 41-150 and described in Table 41-163.

Return to the Summary Table.

System Handler Priority Register 3

Figure 41-150. SHPR3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_15			RESERVED					PRI_14			RESERVED				
R/W-0h			R-0h					R/W-0h			R-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															
R-0h															

Table 41-163. SHPR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	PRI_15	R/W	0h	Priority of system handler 15, SysTick exception Reset type: CM.SYSRESETn
28-24	RESERVED	R	0h	Reserved
23-21	PRI_14	R/W	0h	Priority of system handler 14, PendSV Reset type: CM.SYSRESETn
20-0	RESERVED	R	0h	Reserved

41.12.10.11 SHCSRS Register (Offset = D24h) [Reset = 0h]

SHCSRS is shown in Figure 41-151 and described in Table 41-164.

Return to the Summary Table.

System Handler Control and State Register

Figure 41-151. SHCSRS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED					USGFAULTENA	BUSFAULTENA	MEMFAULTENA
R-0h					R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
SVCALLPENDED	BUSFAULTPENDED	MEMFAULTPENDED	USGFAULTPENDED	SYSTICKACT	PENDSVACT	RESERVED	MONITORACT
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
SVCALLACT	RESERVED			USGFAULTACT	RESERVED	BUSFAULTACT	MEMFAULTACT
R-0h	R-0h			R-0h	R-0h	R-0h	R-0h

Table 41-164. SHCSRS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved
18	USGFAULTENA	R/W	0h	UsageFault enable bit, set to 1 to enable Note:Enable bits, set to 1 to enable the exception, or set to 0 to disable the exception. Reset type: CM.SYSRESETn
17	BUSFAULTENA	R/W	0h	BusFault enable bit, set to 1 to enable Note:Enable bits, set to 1 to enable the exception, or set to 0 to disable the exception. Reset type: CM.SYSRESETn
16	MEMFAULTENA	R/W	0h	MemManage enable bit, set to 1 to enable Note:Enable bits, set to 1 to enable the exception, or set to 0 to disable the exception. Reset type: CM.SYSRESETn
15	SVCALLPENDED	R/W	0h	SVCall pending bit, reads as 1 if exception is pending Note:Pending bits, read as 1 if the exception is pending, or as 0 if it is not pending. You can write to these bits to change the pending status of the exceptions. Reset type: CM.SYSRESETn
14	BUSFAULTPENDED	R/W	0h	BusFault exception pending bit, reads as 1 if exception is pending Note:Pending bits, read as 1 if the exception is pending, or as 0 if it is not pending. You can write to these bits to change the pending status of the exceptions. Reset type: CM.SYSRESETn
13	MEMFAULTPENDED	R/W	0h	MemManage exception pending bit, reads as 1 if exception is pending Note:Pending bits, read as 1 if the exception is pending, or as 0 if it is not pending. You can write to these bits to change the pending status of the exceptions. Reset type: CM.SYSRESETn

Table 41-164. SHCSRS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	USGFAULTPENDED	R/W	0h	UsageFault exception pending bit, reads as 1 if exception is pending Note:Pending bits, read as 1 if the exception is pending, or as 0 if it is not pending. You can write to these bits to change the pending status of the exceptions. Reset type: CM.SYSRESETn
11	SYSTICKACT	R	0h	SysTick exception active bit, reads as 1 if exception is active Note:Active bits, read as 1 if the exception is active, or as 0 if it is not active. You can write to these bits to change the active status of the exceptions, but see the Caution below: Software that changes the value of an active bit in this register without correct adjustment to the stacked content can cause the processor to generate a fault exception. Ensure software that writes to this register retains and subsequently restores the current active status. After you have enabled the system handlers, if you have to change the value of a bit in this register you must use a read-modify-write procedure to ensure that you change only the required bit. Reset type: CM.SYSRESETn
10	PENDSVACT	R	0h	PendSV exception active bit, reads as 1 if exception is active Note:Active bits, read as 1 if the exception is active, or as 0 if it is not active. You can write to these bits to change the active status of the exceptions, but see the Caution below: Software that changes the value of an active bit in this register without correct adjustment to the stacked content can cause the processor to generate a fault exception. Ensure software that writes to this register retains and subsequently restores the current active status. After you have enabled the system handlers, if you have to change the value of a bit in this register you must use a read-modify-write procedure to ensure that you change only the required bit. Reset type: CM.SYSRESETn
9	RESERVED	R	0h	Reserved
8	MONITORACT	R	0h	Debug monitor active bit, reads as 1 if Debug monitor is active Note:Active bits, read as 1 if the exception is active, or as 0 if it is not active. You can write to these bits to change the active status of the exceptions, but see the Caution below: Software that changes the value of an active bit in this register without correct adjustment to the stacked content can cause the processor to generate a fault exception. Ensure software that writes to this register retains and subsequently restores the current active status. After you have enabled the system handlers, if you have to change the value of a bit in this register you must use a read-modify-write procedure to ensure that you change only the required bit. Reset type: CM.SYSRESETn

Table 41-164. SHCSRS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	SVCALLACT	R	0h	SVCall active bit, reads as 1 if SVC call is active Note:Active bits, read as 1 if the exception is active, or as 0 if it is not active. You can write to these bits to change the active status of the exceptions, but see the Caution below: Software that changes the value of an active bit in this register without correct adjustment to the stacked content can cause the processor to generate a fault exception. Ensure software that writes to this register retains and subsequently restores the current active status. After you have enabled the system handlers, if you have to change the value of a bit in this register you must use a read-modify-write procedure to ensure that you change only the required bit. Reset type: CM.SYSRESETn
6-4	RESERVED	R	0h	Reserved
3	USGFAULTACT	R	0h	UsageFault exception active bit, reads as 1 if exception is active Note:Active bits, read as 1 if the exception is active, or as 0 if it is not active. You can write to these bits to change the active status of the exceptions, but see the Caution below: Software that changes the value of an active bit in this register without correct adjustment to the stacked content can cause the processor to generate a fault exception. Ensure software that writes to this register retains and subsequently restores the current active status. After you have enabled the system handlers, if you have to change the value of a bit in this register you must use a read-modify-write procedure to ensure that you change only the required bit. Reset type: CM.SYSRESETn
2	RESERVED	R	0h	Reserved
1	BUSFAULTACT	R	0h	BusFault exception active bit, reads as 1 if exception is active Note:Active bits, read as 1 if the exception is active, or as 0 if it is not active. You can write to these bits to change the active status of the exceptions, but see the Caution below: Software that changes the value of an active bit in this register without correct adjustment to the stacked content can cause the processor to generate a fault exception. Ensure software that writes to this register retains and subsequently restores the current active status. After you have enabled the system handlers, if you have to change the value of a bit in this register you must use a read-modify-write procedure to ensure that you change only the required bit. Reset type: CM.SYSRESETn

Table 41-164. SHCSRS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	MEMFAULTACT	R	0h	MemManage exception active bit, reads as 1 if exception is active Note:Active bits, read as 1 if the exception is active, or as 0 if it is not active. You can write to these bits to change the active status of the exceptions, but see the Caution below: Software that changes the value of an active bit in this register without correct adjustment to the stacked content can cause the processor to generate a fault exception. Ensure software that writes to this register retains and subsequently restores the current active status. After you have enabled the system handlers, if you have to change the value of a bit in this register you must use a read-modify-write procedure to ensure that you change only the required bit. Reset type: CM.SYSRESETn

41.12.10.12 CFSR Register (Offset = D28h) [Reset = 0h]

CFSR is shown in Figure 41-152 and described in Table 41-165.

Return to the Summary Table.

Configurable Fault Status Register

Figure 41-152. CFSR Register

31	30	29	28	27	26	25	24
RESERVED						DIVBYZERO	UNALIGNED
R-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED				NOCP	INVPC	INVSTATE	UNDEFINSTR
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
BFARVALID	RESERVED	RESERVED	STKERR	UNSTKERR	IMPRECISERR	PRECISERR	IBUSERR
R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MMARVALID	RESERVED	RESERVED	MSTKERR	MUNSTKERR	RESERVED	DACCVIOL	IACCVIOL
R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h

Table 41-165. CFSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved
25	DIVBYZERO	R/W	0h	Divide by zero UsageFault: 0 = no divide by zero fault, or divide by zero trapping not enabled 1 = the processor has executed an SDIV or UDIV instruction with a divisor of 0. When the processor sets this bit to 1, the PC value stacked for the exception return points to the instruction that performed the divide by zero. Enable trapping of divide by zero by setting the DIV_0_TRP bit in the CCR to 1. Note:The UFSR bits are sticky. This means as one or more fault occurs, the associated bits are set to 1. A bit that is set to 1 is cleared to 0 only by writing 1 to that bit, or by a reset. Reset type: CM.SYSRESETn
24	UNALIGNED	R/W	0h	Unaligned access UsageFault: 0 = no unaligned access fault, or unaligned access trapping not enabled 1 = the processor has made an unaligned memory access. Enable trapping of unaligned accesses by setting the UNALIGN_TRP bit in the CCR to 1. Unaligned LDM, STM, LDRD, and STRD instructions always fault irrespective of the setting of UNALIGN_TRP. Note:The UFSR bits are sticky. This means as one or more fault occurs, the associated bits are set to 1. A bit that is set to 1 is cleared to 0 only by writing 1 to that bit, or by a reset. Reset type: CM.SYSRESETn
23-20	RESERVED	R	0h	Reserved

Table 41-165. CFSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	NOCP	R/W	0h	No coprocessor UsageFault. The processor does not support coprocessor instructions: 0 = no UsageFault caused by attempting to access a coprocessor 1 = the processor has attempted to access a coprocessor. Note:The UFSR bits are sticky. This means as one or more fault occurs, the associated bits are set to 1. A bit that is set to 1 is cleared to 0 only by writing 1 to that bit, or by a reset. Reset type: CM.SYSRESETn
18	INVPC	R/W	0h	Invalid PC load UsageFault, caused by an invalid PC load by EXC_RETURN: 0 = no invalid PC load UsageFault 1 = the processor has attempted an illegal load of EXC_RETURN to the PC, as a result of an invalid context, or an invalid EXC_RETURN value. When this bit is set to 1, the PC value stacked for the exception return points to the instruction that tried to perform the illegal load of the PC. Note:The UFSR bits are sticky. This means as one or more fault occurs, the associated bits are set to 1. A bit that is set to 1 is cleared to 0 only by writing 1 to that bit, or by a reset. Reset type: CM.SYSRESETn
17	INVSTATE	R/W	0h	Invalid state UsageFault: 0 = no invalid state UsageFault 1 = the processor has attempted to execute an instruction that makes illegal use of the EPSR. When this bit is set to 1, the PC value stacked for the exception return points to the instruction that attempted the illegal use of the EPSR. This bit is not set to 1 if an undefined instruction uses the EPSR. Note:The UFSR bits are sticky. This means as one or more fault occurs, the associated bits are set to 1. A bit that is set to 1 is cleared to 0 only by writing 1 to that bit, or by a reset. Reset type: CM.SYSRESETn
16	UNDEFINSTR	R/W	0h	Undefined instruction UsageFault: 0 = no undefined instruction UsageFault 1 = the processor has attempted to execute an undefined instruction. When this bit is set to 1, the PC value stacked for the exception return points to the undefined instruction. An undefined instruction is an instruction that the processor cannot decode Note:The UFSR bits are sticky. This means as one or more fault occurs, the associated bits are set to 1. A bit that is set to 1 is cleared to 0 only by writing 1 to that bit, or by a reset. Reset type: CM.SYSRESETn
15	BFARVALID	R/W	0h	BusFault Address Register (BFAR) valid flag: 0 = value in BFAR is not a valid fault address 1 = BFAR holds a valid fault address. The processor sets this bit to 1 after a BusFault where the address is known. Other faults can set this bit to 0, such as a MemManage fault occurring later. If a BusFault occurs and is escalated to a hard fault because of priority, the hard fault handler must set this bit to 0. This prevents problems if returning to a stacked active BusFault handler whose BFAR value has been overwritten. Reset type: CM.SYSRESETn
14	RESERVED	R	0h	Reserved
13	RESERVED	R/W	0h	Reserved

Table 41-165. CFSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	STKERR	R/W	0h	BusFault on stacking for exception entry: 0 = no stacking fault 1 = stacking for an exception entry has caused one or more BusFaults. When the processor sets this bit to 1, the SP is still adjusted but the values in the context area on the stack might be incorrect. The processor does not write a fault address to the BFAR. Reset type: CM.SYSRESETn
11	UNSTKERR	R/W	0h	BusFault on unstacking for a return from exception: 0 = no unstacking fault 1 = unstack for an exception return has caused one or more BusFaults. This fault is chained to the handler. This means that when the processor sets this bit to 1, the original return stack is still present. The processor does not adjust the SP from the failing return, does not performed a new save, and does not write a fault address to the BFAR. Reset type: CM.SYSRESETn
10	IMPRECISERR	R/W	0h	Imprecise data bus error: 0 = no imprecise data bus error 1 = a data bus error has occurred, but the return address in the stack frame is not related to the instruction that caused the error. When the processor sets this bit to 1, it does not write a fault address to the BFAR. This is an asynchronous fault. Therefore, if it is detected when the priority of the current process is higher than the BusFault priority, the BusFault becomes pending and becomes active only when the processor returns from all higher priority processes. If a precise fault occurs before the processor enters the handler for the imprecise BusFault, the handler detects both IMPRECISERR set to 1 and one of the precise fault status bits set to 1. Reset type: CM.SYSRESETn
9	PRECISERR	R/W	0h	Precise data bus error: 0 = no precise data bus error 1 = a data bus error has occurred, and the PC value stacked for the exception return points to the instruction that caused the fault. When the processor sets this bit is 1, it writes the faulting address to the BFAR. Reset type: CM.SYSRESETn
8	IBUSERR	R/W	0h	Instruction bus error: 0 = no instruction bus error 1 = instruction bus error. The processor detects the instruction bus error on prefetching an instruction, but it sets the IBUSERR flag to 1 only if it attempts to issue the faulting instruction. When the processor sets this bit is 1, it does not write a fault address to the BFAR. Reset type: CM.SYSRESETn
7	MMARVALID	R/W	0h	MemManage Fault Address Register (MMFAR) valid flag.. 0 = value in MMAR is not a valid fault address 1 = MMAR holds a valid fault address. If a MemManage fault occurs and is escalated to a HardFault because of priority, the HardFault handler must set this bit to 0. This prevents problems on return to a stacked active MemManage fault handler whose MMAR value has been overwritten. Reset type: CM.SYSRESETn
6	RESERVED	R	0h	Reserved

Table 41-165. CFSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	RESERVED	R/W	0h	Reserved
4	MSTKERR	R/W	0h	MemManage fault on stacking for exception entry.. 0 = no stacking fault 1 = stacking for an exception entry has caused one or more access violations. When this bit is 1, the SP is still adjusted but the values in the context area on the stack might be incorrect. The processor has not written a fault address to the MMAR. Reset type: CM.SYSRESETn
3	MUNSTKERR	R/W	0h	MemManage fault on unstacking for a return from exception.. 0 = no unstacking fault 1 = unstack for an exception return has caused one or more access violations. This fault is chained to the handler. This means that when this bit is 1, the original return stack is still present. The processor has not adjusted the SP from the failing return, and has not performed a new save. The processor has not written a fault address to the MMAR. Reset type: CM.SYSRESETn
2	RESERVED	R	0h	Reserved
1	DACCVIOL	R/W	0h	Data access violation flag.. 0 = no data access violation fault 1 = the processor attempted a load or store at a location that does not permit the operation. When this bit is 1, the PC value stacked for the exception return points to the faulting instruction. The processor has loaded the MMAR with the address of the attempted access. Reset type: CM.SYSRESETn
0	IACCVIOL	R/W	0h	Instruction access violation flag.. 0 = no instruction access violation fault 1 = the processor attempted an instruction fetch from a location that does not permit execution. This fault occurs on any access to an XN region, even when the MPU is disabled or not present. When this bit is 1, the PC value stacked for the exception return points to the faulting instruction. The processor has not written a fault address to the MMAR. Reset type: CM.SYSRESETn

41.12.10.13 HFSR Register (Offset = D2Ch) [Reset = 0h]

HFSR is shown in Figure 41-153 and described in Table 41-166.

Return to the Summary Table.

HardFault Status Register

Figure 41-153. HFSR Register

31	30	29	28	27	26	25	24
DEBUGEVT	FORCED	RESERVED					
R/W-0h	R/W-0h	R-0h					
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						VECTTBL	RESERVED
R-0h						R/W-0h	R-0h

Table 41-166. HFSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	DEBUGEVT	R/W	0h	Reserved for Debug use. When writing to the register you must write 0 to this bit, otherwise behavior is Unpredictable. Reset type: CM.SYSRESETn
30	FORCED	R/W	0h	Indicates a forced hard fault, generated by escalation of a fault with configurable priority that cannot be handles, either because of priority or because it is disabled: 0 = no forced HardFault 1 = forced HardFault. When this bit is set to 1, the HardFault handler must read the other fault status registers to find the cause of the fault. Reset type: CM.SYSRESETn
29-2	RESERVED	R	0h	Reserved
1	VECTTBL	R/W	0h	Indicates a BusFault on a vector table read during exception processing: 0 = no BusFault on vector table read 1 = BusFault on vector table read. This error is always handled by the hard fault handler. When this bit is set to 1, the PC value stacked for the exception return points to the instruction that was preempted by the exception. Reset type: CM.SYSRESETn
0	RESERVED	R	0h	Reserved

41.12.10.14 MMFAR Register (Offset = D34h) [Reset = 0h]

MMFAR is shown in Figure 41-154 and described in Table 41-167.

Return to the Summary Table.

MemManage Fault Address Register

Figure 41-154. MMFAR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS																															
R/W-0h																															

Table 41-167. MMFAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRESS	R/W	0h	When the MMARVALID bit of the MMFSR is set to 1, this field holds the address of the location that generated the MemManage fault Reset type: CM.SYSRESETn

41.12.10.15 BFAR Register (Offset = D38h) [Reset = 0h]

BFAR is shown in Figure 41-155 and described in Table 41-168.

Return to the Summary Table.

BusFault Address Register

Figure 41-155. BFAR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS																															
R/W-0h																															

Table 41-168. BFAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRESS	R/W	0h	When the BFARVALID bit of the BFSR is set to 1, this field holds the address of the location that generated the BusFault Reset type: CM.SYSRESETn

41.12.10.16 AFSR Register (Offset = D3Ch) [Reset = 0h]

AFSR is shown in Figure 41-156 and described in Table 41-169.

Return to the Summary Table.

Auxiliary Fault Status Register

Figure 41-156. AFSR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS																															
R/W-0h																															

Table 41-169. AFSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRESS	R/W	0h	Implementation defined. The bits map to the AUXFAULT input signals. Reset type: CM.SYSRESETn

41.12.11 CSFR Registers

Table 41-170 lists the memory-mapped registers for the CSFR registers. All register offset addresses not listed in Table 41-170 should be considered as reserved locations and the register contents should not be modified.

Table 41-170. CSFR Registers

Offset	Acronym	Register Name	Write Protection	Section
D28h	MMSR	MemManage Fault Status Register		Go
D29h	BFSR	BusFault Status Register		Go
D2Ah	UFSR	UsageFault Status Register		Go

Complex bit access types are encoded to fit into small table cells. Table 41-171 shows the codes that are used for access types in this section.

Table 41-171. CSFR Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

41.12.11.1 MMSR Register (Offset = D28h) [Reset = 0h]

MMSR is shown in Figure 41-157 and described in Table 41-172.

Return to the Summary Table.

MemManage Fault Status Register

Figure 41-157. MMSR Register

7	6	5	4	3	2	1	0
MMARVALID	RESERVED	RESERVED	MSTKERR	MUNSTKERR	RESERVED	DACCVIOL	IACCVIOL
R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h

Table 41-172. MMSR Register Field Descriptions

Bit	Field	Type	Reset	Description
7	MMARVALID	R/W	0h	MemManage Fault Address Register (MMFAR) valid flag.. 0 = value in MMAR is not a valid fault address 1 = MMAR holds a valid fault address. If a MemManage fault occurs and is escalated to a HardFault because of priority, the HardFault handler must set this bit to 0. This prevents problems on return to a stacked active MemManage fault handler whose MMAR value has been overwritten. Reset type: CM.SYSRESETn
6	RESERVED	R	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	MSTKERR	R/W	0h	MemManage fault on stacking for exception entry.. 0 = no stacking fault 1 = stacking for an exception entry has caused one or more access violations. When this bit is 1, the SP is still adjusted but the values in the context area on the stack might be incorrect. The processor has not written a fault address to the MMAR. Reset type: CM.SYSRESETn
3	MUNSTKERR	R/W	0h	MemManage fault on unstacking for a return from exception.. 0 = no unstacking fault 1 = unstack for an exception return has caused one or more access violations. This fault is chained to the handler. This means that when this bit is 1, the original return stack is still present. The processor has not adjusted the SP from the failing return, and has not performed a new save. The processor has not written a fault address to the MMAR. Reset type: CM.SYSRESETn
2	RESERVED	R	0h	Reserved
1	DACCVIOL	R/W	0h	Data access violation flag.. 0 = no data access violation fault 1 = the processor attempted a load or store at a location that does not permit the operation. When this bit is 1, the PC value stacked for the exception return points to the faulting instruction. The processor has loaded the MMAR with the address of the attempted access. Reset type: CM.SYSRESETn

Table 41-172. MMSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	IACCVIOL	R/W	0h	Instruction access violation flag.. 0 = no instruction access violation fault 1 = the processor attempted an instruction fetch from a location that does not permit execution. This fault occurs on any access to an XN region, even when the MPU is disabled or not present. When this bit is 1, the PC value stacked for the exception return points to the faulting instruction. The processor has not written a fault address to the MMAR. Reset type: CM.SYSRESETn

41.12.11.2 BFSR Register (Offset = D29h) [Reset = 0h]

BFSR is shown in Figure 41-158 and described in Table 41-173.

Return to the Summary Table.

BusFault Status Register

Figure 41-158. BFSR Register

7	6	5	4	3	2	1	0
BFARVALID	RESERVED	RESERVED	STKERR	UNSTKERR	IMPRECISERR	PRECISERR	IBUSERR
R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 41-173. BFSR Register Field Descriptions

Bit	Field	Type	Reset	Description
7	BFARVALID	R/W	0h	BusFault Address Register (BFAR) valid flag: 0 = value in BFAR is not a valid fault address 1 = BFAR holds a valid fault address. The processor sets this bit to 1 after a BusFault where the address is known. Other faults can set this bit to 0, such as a MemManage fault occurring later. If a BusFault occurs and is escalated to a hard fault because of priority, the hard fault handler must set this bit to 0. This prevents problems if returning to a stacked active BusFault handler whose BFAR value has been overwritten. Reset type: CM.SYSRESETn
6	RESERVED	R	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	STKERR	R/W	0h	BusFault on stacking for exception entry: 0 = no stacking fault 1 = stacking for an exception entry has caused one or more BusFaults. When the processor sets this bit to 1, the SP is still adjusted but the values in the context area on the stack might be incorrect. The processor does not write a fault address to the BFAR. Reset type: CM.SYSRESETn
3	UNSTKERR	R/W	0h	BusFault on unstacking for a return from exception: 0 = no unstacking fault 1 = unstack for an exception return has caused one or more BusFaults. This fault is chained to the handler. This means that when the processor sets this bit to 1, the original return stack is still present. The processor does not adjust the SP from the failing return, does not performed a new save, and does not write a fault address to the BFAR. Reset type: CM.SYSRESETn

Table 41-173. BFSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	IMPRECISERR	R/W	0h	Imprecise data bus error: 0 = no imprecise data bus error 1 = a data bus error has occurred, but the return address in the stack frame is not related to the instruction that caused the error. When the processor sets this bit to 1, it does not write a fault address to the BFAR. This is an asynchronous fault. Therefore, if it is detected when the priority of the current process is higher than the BusFault priority, the BusFault becomes pending and becomes active only when the processor returns from all higher priority processes. If a precise fault occurs before the processor enters the handler for the imprecise BusFault, the handler detects both IMPRECISERR set to 1 and one of the precise fault status bits set to 1. Reset type: CM.SYSRESETn
1	PRECISERR	R/W	0h	Precise data bus error: 0 = no precise data bus error 1 = a data bus error has occurred, and the PC value stacked for the exception return points to the instruction that caused the fault. When the processor sets this bit is 1, it writes the faulting address to the BFAR. Reset type: CM.SYSRESETn
0	IBUSERR	R/W	0h	Instruction bus error: 0 = no instruction bus error 1 = instruction bus error. The processor detects the instruction bus error on prefetching an instruction, but it sets the IBUSERR flag to 1 only if it attempts to issue the faulting instruction. When the processor sets this bit is 1, it does not write a fault address to the BFAR. Reset type: CM.SYSRESETn

41.12.11.3 UFSR Register (Offset = D2Ah) [Reset = 0h]

UFSR is shown in Figure 41-159 and described in Table 41-174.

Return to the Summary Table.

UsageFault Status Register

Figure 41-159. UFSR Register

15	14	13	12	11	10	9	8
RESERVED						DIVBYZERO	UNALIGNED
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED				NOCP	INVPC	INVSTATE	UNDEFINSTR
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 41-174. UFSR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9	DIVBYZERO	R/W	0h	Divide by zero UsageFault: 0 = no divide by zero fault, or divide by zero trapping not enabled 1 = the processor has executed an SDIV or UDIV instruction with a divisor of 0. When the processor sets this bit to 1, the PC value stacked for the exception return points to the instruction that performed the divide by zero. Enable trapping of divide by zero by setting the DIV_0_TRP bit in the CCR to 1. Note:The UFSR bits are sticky. This means as one or more fault occurs, the associated bits are set to 1. A bit that is set to 1 is cleared to 0 only by writing 1 to that bit, or by a reset. Reset type: CM.SYSRESETn
8	UNALIGNED	R/W	0h	Unaligned access UsageFault: 0 = no unaligned access fault, or unaligned access trapping not enabled 1 = the processor has made an unaligned memory access. Enable trapping of unaligned accesses by setting the UNALIGN_TRP bit in the CCR to 1. Unaligned LDM, STM, LDRD, and STRD instructions always fault irrespective of the setting of UNALIGN_TRP. Note:The UFSR bits are sticky. This means as one or more fault occurs, the associated bits are set to 1. A bit that is set to 1 is cleared to 0 only by writing 1 to that bit, or by a reset. Reset type: CM.SYSRESETn
7-4	RESERVED	R	0h	Reserved
3	NOCP	R/W	0h	No coprocessor UsageFault. The processor does not support coprocessor instructions: 0 = no UsageFault caused by attempting to access a coprocessor 1 = the processor has attempted to access a coprocessor. Note:The UFSR bits are sticky. This means as one or more fault occurs, the associated bits are set to 1. A bit that is set to 1 is cleared to 0 only by writing 1 to that bit, or by a reset. Reset type: CM.SYSRESETn

Table 41-174. UFSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	INVPC	R/W	0h	Invalid PC load UsageFault, caused by an invalid PC load by EXC_RETURN: 0 = no invalid PC load UsageFault 1 = the processor has attempted an illegal load of EXC_RETURN to the PC, as a result of an invalid context, or an invalid EXC_RETURN value. When this bit is set to 1, the PC value stacked for the exception return points to the instruction that tried to perform the illegal load of the PC. Note:The UFSR bits are sticky. This means as one or more fault occurs, the associated bits are set to 1. A bit that is set to 1 is cleared to 0 only by writing 1 to that bit, or by a reset. Reset type: CM.SYSRESETn
1	INVSTATE	R/W	0h	Invalid state UsageFault: 0 = no invalid state UsageFault 1 = the processor has attempted to execute an instruction that makes illegal use of the EPSR. When this bit is set to 1, the PC value stacked for the exception return points to the instruction that attempted the illegal use of the EPSR. This bit is not set to 1 if an undefined instruction uses the EPSR. Note:The UFSR bits are sticky. This means as one or more fault occurs, the associated bits are set to 1. A bit that is set to 1 is cleared to 0 only by writing 1 to that bit, or by a reset. Reset type: CM.SYSRESETn
0	UNDEFINSTR	R/W	0h	Undefined instruction UsageFault: 0 = no undefined instruction UsageFault 1 = the processor has attempted to execute an undefined instruction. When this bit is set to 1, the PC value stacked for the exception return points to the undefined instruction. An undefined instruction is an instruction that the processor cannot decode Note:The UFSR bits are sticky. This means as one or more fault occurs, the associated bits are set to 1. A bit that is set to 1 is cleared to 0 only by writing 1 to that bit, or by a reset. Reset type: CM.SYSRESETn

41.12.12 SYSTICK Registers

Table 41-175 lists the memory-mapped registers for the SYSTICK registers. All register offset addresses not listed in Table 41-175 should be considered as reserved locations and the register contents should not be modified.

Table 41-175. SYSTICK Registers

Offset	Acronym	Register Name	Write Protection	Section
10h	SYST_CSR	Privileged a SysTick Control and Status Register		Go
14h	SYST_RVR	Privileged Unknown SysTick Reload Value Register		Go
18h	SYST_CVR	Privileged Unknown SysTick Current Value Register		Go
1Ch	SYST_CALIB	Privileged -a SysTick Calibration Value Register		Go

Complex bit access types are encoded to fit into small table cells. Table 41-176 shows the codes that are used for access types in this section.

Table 41-176. SYSTICK Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

41.12.12.1 SYST_CSR Register (Offset = 10h) [Reset = 4h]

SYST_CSR is shown in Figure 41-160 and described in Table 41-177.

Return to the Summary Table.

Privileged a SysTick Control and Status Register

Figure 41-160. SYST_CSR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							COUNTFLAG
R-0h							R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					CLKSOURCE	TICKINT	ENABLE
R-0h					R/W-1h	R/W-0h	R/W-0h

Table 41-177. SYST_CSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	COUNTFLAG	R/W	0h	Returns 1 if timer counted to 0 since last time this was read. Reset type: CM.SYSRESETn
15-3	RESERVED	R	0h	Reserved
2	CLKSOURCE	R/W	1h	Indicates the clock source: 0 = external clock 1 = processor clock. Reset type: CM.SYSRESETn
1	TICKINT	R/W	0h	Enables SysTick exception request: 0 = counting down to zero does not assert the SysTick exception request 1 = counting down to zero asserts the SysTick exception request. Software can use COUNTFLAG to determine if SysTick has ever counted to zero. Reset type: CM.SYSRESETn
0	ENABLE	R/W	0h	Enables the counter: 0 = counter disabled 1 = counter enabled. Reset type: CM.SYSRESETn

41.12.12.2 SYST_RVR Register (Offset = 14h) [Reset = 0h]

SYST_RVR is shown in Figure 41-161 and described in Table 41-178.

Return to the Summary Table.

Privileged Unknown SysTick Reload Value Register

Figure 41-161. SYST_RVR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RELOAD																							
R-0h								R/W-0h																							

Table 41-178. SYST_RVR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-0	RELOAD	R/W	0h	Value to load into the SYST_CVR register when the counter is enabled and when it reaches 0. The RELOAD value can be any value in the range 0x00000001-0x00FFFFFF. A start value of 0 is possible, but has no effect because the SysTick exception request and COUNTFLAG are activated when counting from 1 to 0. The RELOAD value is calculated according to its use. For example, to generate a multi-shot timer with a period of N processor clock cycles, use a RELOAD value of N-1. If the SysTick interrupt is required every 100 clock pulses, set RELOAD to 99. Reset type: CM.SYSRESETn

41.12.12.3 SYST_CVR Register (Offset = 18h) [Reset = 0h]

SYST_CVR is shown in Figure 41-162 and described in Table 41-179.

Return to the Summary Table.

Privileged Unknown SysTick Current Value Register

Figure 41-162. SYST_CVR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								CURRENT																							
R-0h								R/W-0h																							

Table 41-179. SYST_CVR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-0	CURRENT	R/W	0h	Reads return the current value of the SysTick counter. A write of any value clears the field to 0, and also clears the SYST_CSR COUNTFLAG bit to 0. Reset type: CM.SYSRESETn

41.12.12.4 SYST_CALIB Register (Offset = 1Ch) [Reset = 0h]

SYST_CALIB is shown in Figure 41-163 and described in Table 41-180.

Return to the Summary Table.

Privileged -a SysTick Calibration Value Register

Figure 41-163. SYST_CALIB Register

31	30	29	28	27	26	25	24
NOREF	SKEW	RESERVED					
R/W-0h	R/W-0h	R-0h					
23	22	21	20	19	18	17	16
TENMS							
R/W-0h							
15	14	13	12	11	10	9	8
TENMS							
R/W-0h							
7	6	5	4	3	2	1	0
TENMS							
R/W-0h							

Table 41-180. SYST_CALIB Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NOREF	R/W	0h	Indicates whether the device provides a reference clock to the processor: 0 = reference clock provided 1 = no reference clock provided. If your device does not provide a reference clock, the SYST_CSR.CLKSOURCE bit reads-as-one and ignores writes. Reset type: CM.SYSRESETn
30	SKEW	R/W	0h	Indicates whether the TENMS value is exact: 0 = TENMS value is exact 1 = TENMS value is inexact, or not given. An inexact TENMS value can affect the suitability of SysTick as a software real time clock. Reset type: CM.SYSRESETn
29-24	RESERVED	R	0h	Reserved
23-0	TENMS	R/W	0h	Reload value for 10ms (100Hz) timing, subject to system clock skew errors. If the value reads as zero, the calibration value is not known. Reset type: CM.SYSRESETn

41.12.13 MPU Registers

Table 41-181 lists the memory-mapped registers for the MPU registers. All register offset addresses not listed in Table 41-181 should be considered as reserved locations and the register contents should not be modified.

Table 41-181. MPU Registers

Offset	Acronym	Register Name	Write Protection	Section
D90h	MPU_TYPE	MPU Type Register		Go
D94h	MPU_CTRL	MPU Control Register		Go
D98h	MPU_RNR	MPU Region Number Register		Go
D9Ch	MPU_RBAR	MPU Region Base Address Register		Go
DA0h	MPU_RASR	MPU Region Attribute and Size Register		Go
DA4h	MPU_RBAR_A1	Alias of RBAR		Go
DA8h	MPU_RASR_A1	Alias of RASR		Go
DACh	MPU_RBAR_A2	Alias of RBAR		Go
DB0h	MPU_RASR_A2	Alias of RASR		Go
DB4h	MPU_RBAR_A3	Alias of RBAR		Go
DB8h	MPU_RASR_A3	Alias of RASR		Go

Complex bit access types are encoded to fit into small table cells. Table 41-182 shows the codes that are used for access types in this section.

Table 41-182. MPU Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

41.12.13.1 MPU_TYPE Register (Offset = D90h) [Reset = 800h]

MPU_TYPE is shown in Figure 41-164 and described in Table 41-183.

Return to the Summary Table.

MPU Type Register

Figure 41-164. MPU_TYPE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
IREGION							
R-0h							
15	14	13	12	11	10	9	8
DREGION							
R-8h							
7	6	5	4	3	2	1	0
RESERVED							SEPARATE
R-0h							R-0h

Table 41-183. MPU_TYPE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	IREGION	R	0h	Indicates the number of supported MPU instruction regions. Always contains 0x00. The MPU memory map is unified and is described by the DREGION field. Reset type: CM.SYSRESETn
15-8	DREGION	R	8h	Indicates the number of supported MPU data regions: 0x08 = Eight MPU regions. Reset type: CM.SYSRESETn
7-1	RESERVED	R	0h	Reserved
0	SEPARATE	R	0h	Indicates support for unified or separate instruction and date memory maps: 0 = unified. Reset type: CM.SYSRESETn

41.12.13.2 MPU_CTRL Register (Offset = D94h) [Reset = 0h]

MPU_CTRL is shown in Figure 41-165 and described in Table 41-184.

Return to the Summary Table.

MPU Control Register

Figure 41-165. MPU_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					PRIVDEFENA	HFNMIENA	ENABLE
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 41-184. MPU_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2	PRIVDEFENA	R/W	0h	Enables privileged software access to the default memory map: 0 = If the MPU is enabled, disables use of the default memory map. Any memory access to a location not covered by any enabled region causes a fault. 1 = If the MPU is enabled, enables use of the default memory map as a background region for privileged software accesses. When enabled, the background region acts as if it is region number -1. Any region that is defined and enabled has priority over this default map. If the MPU is disabled, the processor ignores this bit. Reset type: CM.SYSRESETn
1	HFNMIENA	R/W	0h	Enables the operation of MPU during hard fault, NMI, and FAULTMASK handlers. When the MPU is enabled: 0 = MPU is disabled during hard fault, NMI, and FAULTMASK handlers, regardless of the value of the ENABLE bit 1 = the MPU is enabled during hard fault, NMI, and FAULTMASK handlers. When the MPU is disabled, if this bit is set to 1 the behavior is Unpredictable. Reset type: CM.SYSRESETn
0	ENABLE	R/W	0h	Enables the MPU: 0 = MPU disabled 1 = MPU enabled. Reset type: CM.SYSRESETn

41.12.13.3 MPU_RNR Register (Offset = D98h) [Reset = 0h]

MPU_RNR is shown in Figure 41-166 and described in Table 41-185.

Return to the Summary Table.

MPU Region Number Register

Figure 41-166. MPU_RNR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								REGION							
R-0h																								R/W-0h							

Table 41-185. MPU_RNR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	REGION	R/W	0h	Indicates the MPU region referenced by the MPU_RBAR and MPU_RASR registers. The MPU supports 8 memory regions, so the permitted values of this field are 0-7. Reset type: CM.SYSRESETn

41.12.13.4 MPU_RBAR Register (Offset = D9Ch) [Reset = 0h]

MPU_RBAR is shown in Figure 41-167 and described in Table 41-186.

Return to the Summary Table.

MPU Region Base Address Register

Figure 41-167. MPU_RBAR Register

31	30	29	28	27	26	25	24
ADDR							
R/W-0h							
23	22	21	20	19	18	17	16
ADDR							
R/W-0h							
15	14	13	12	11	10	9	8
ADDR							
R/W-0h							
7	6	5	4	3	2	1	0
ADDR			VALID	REGION			
R/W-0h			R/W-0h	R/W-0h			

Table 41-186. MPU_RBAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	ADDR	R/W	0h	Region base address field. The ADDR field is bits[31:N] of the MPU_RBAR. The region size, as specified by the SIZE field in the MPU_RASR, defines the value of N: N = Log2(Region size in bytes), If the region size is configured to 4GB, in the MPU_RASR, there is no valid ADDR field. In this case, the region occupies the complete memory map, and the base address is 0x00000000. The base address is aligned to the size of the region. For example, a 64KB region must be aligned on a multiple of 64KB, for example, at 0x00010000 or 0x00020000. Reset type: CM.SYSRESETn
4	VALID	R/W	0h	MPU Region Number valid bit: Write: 0 = MPU_RNR not changed, and the processor: updates the base address for the region specified in the MPU_RNR ignores the value of the REGION field 1 = the processor: updates the value of the MPU_RNR to the value of the REGION field updates the base address for the region specified in the REGION field. Always reads as zero. Reset type: CM.SYSRESETn
3-0	REGION	R/W	0h	MPU region field: For the behavior on writes, see the description of the VALID field. On reads, returns the current region number, as specified by the RNR. Reset type: CM.SYSRESETn

41.12.13.5 MPU_RASR Register (Offset = DA0h) [Reset = 0h]

MPU_RASR is shown in Figure 41-168 and described in Table 41-187.

Return to the Summary Table.

MPU Region Attribute and Size Register

Figure 41-168. MPU_RASR Register

31	30	29	28	27	26	25	24
RESERVED			XN	RESERVED	AP		
R-0h			R/W-0h	R-0h	R/W-0h		
23	22	21	20	19	18	17	16
RESERVED			TEX		S	C	B
R-0h			R/W-0h		R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
SRD							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED		SIZE					ENABLE
R-0h		R/W-0h					R/W-0h

Table 41-187. MPU_RASR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved
28	XN	R/W	0h	Instruction access disable bit: 0 = instruction fetches enabled 1 = instruction fetches disabled. Reset type: CM.SYSRESETn
27	RESERVED	R	0h	Reserved
26-24	AP	R/W	0h	AP[2:0] Privileged permissions Unprivileged permissions Description 000 No access No access All accesses generate a permission fault 001 RW No access Access from privileged software only 010 RW RO Writes by unprivileged software generate a permission fault 011 RW RW Full access 100 Unpredictable Unpredictable Reserved 101 RO No access Reads by privileged software only 110 RO RO Read only, by privileged or unprivileged software 111 RO RO Read only, by privileged or unprivileged software Reset type: CM.SYSRESETn
23-22	RESERVED	R	0h	Reserved

Table 41-187. MPU_RASR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-19	TEX	R/W	0h	TEX C B S Memory type Shareability Other attributes 0b000 0 0 x Strongly-ordered Shareable - 0 1 x Device Shareable - 1 0 0 Normal Not shareable Outer and inner write-through. No write allocate. 1 0 1 Shareable 1 1 0 Normal Not shareable Outer and inner write-back. No write allocate. 1 1 1 Shareable 0b001 0 0 0 Normal Not shareable Outer and inner noncacheable. 0 0 1 Shareable 0 1 x Reserved encoding - 1 0 x Implementation defined attributes. - 1 1 0 Normal Not shareable Outer and inner write-back. Write and read allocate. 1 1 1 Shareable 0b010 0 0 x Device Not shareable Nonshared Device. 0 1 x Reserved encoding - 1 x x Reserved encoding - 0b1BB A A 0 Normal Not shareable Cached memory, BB = outer policy, AA = inner policy. Encoding, AA or BB Corresponding cache policy 00 Non-cacheable 01 Write back, write and read allocate 10 Write through, no write allocate 11 Write back, no write allocate 1 Shareable and BB bits. Reset type: CM.SYSRESETn
18	S	R/W	0h	TEX C B S Memory type Shareability Other attributes 0b000 0 0 x Strongly-ordered Shareable - 0 1 x Device Shareable - 1 0 0 Normal Not shareable Outer and inner write-through. No write allocate. 1 0 1 Shareable 1 1 0 Normal Not shareable Outer and inner write-back. No write allocate. 1 1 1 Shareable 0b001 0 0 0 Normal Not shareable Outer and inner noncacheable. 0 0 1 Shareable 0 1 x Reserved encoding - 1 0 x Implementation defined attributes. - 1 1 0 Normal Not shareable Outer and inner write-back. Write and read allocate. 1 1 1 Shareable 0b010 0 0 x Device Not shareable Nonshared Device. 0 1 x Reserved encoding - 1 x x Reserved encoding - 0b1BB A A 0 Normal Not shareable Cached memory, BB = outer policy, AA = inner policy. Encoding, AA or BB Corresponding cache policy 00 Non-cacheable 01 Write back, write and read allocate 10 Write through, no write allocate 11 Write back, no write allocate 1 Shareable and BB bits. Reset type: CM.SYSRESETn

Table 41-187. MPU_RASR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	C	R/W	0h	TEX C B S Memory type Shareability Other attributes 0b000 0 0 x Strongly-ordered Shareable - 0 1 x Device Shareable - 1 0 0 Normal Not shareable Outer and inner write-through. No write allocate. 1 0 1 Shareable 1 1 0 Normal Not shareable Outer and inner write-back. No write allocate. 1 1 1 Shareable 0b001 0 0 0 Normal Not shareable Outer and inner noncacheable. 0 0 1 Shareable 0 1 x Reserved encoding - 1 0 x Implementation defined attributes. - 1 1 0 Normal Not shareable Outer and inner write-back. Write and read allocate. 1 1 1 Shareable 0b010 0 0 x Device Not shareable Nonshared Device. 0 1 x Reserved encoding - 1 x x Reserved encoding - 0b1BB A A 0 Normal Not shareable Cached memory, BB = outer policy, AA = inner policy. Encoding, AA or BB Corresponding cache policy 00 Non-cacheable 01 Write back, write and read allocate 10 Write through, no write allocate 11 Write back, no write allocate 1 Shareable and BB bits. Reset type: CM.SYSRESETn
16	B	R/W	0h	TEX C B S Memory type Shareability Other attributes 0b000 0 0 x Strongly-ordered Shareable - 0 1 x Device Shareable - 1 0 0 Normal Not shareable Outer and inner write-through. No write allocate. 1 0 1 Shareable 1 1 0 Normal Not shareable Outer and inner write-back. No write allocate. 1 1 1 Shareable 0b001 0 0 0 Normal Not shareable Outer and inner noncacheable. 0 0 1 Shareable 0 1 x Reserved encoding - 1 0 x Implementation defined attributes. - 1 1 0 Normal Not shareable Outer and inner write-back. Write and read allocate. 1 1 1 Shareable 0b010 0 0 x Device Not shareable Nonshared Device. 0 1 x Reserved encoding - 1 x x Reserved encoding - 0b1BB A A 0 Normal Not shareable Cached memory, BB = outer policy, AA = inner policy. Encoding, AA or BB Corresponding cache policy 00 Non-cacheable 01 Write back, write and read allocate 10 Write through, no write allocate 11 Write back, no write allocate 1 Shareable and BB bits. Reset type: CM.SYSRESETn
15-8	SRD	R/W	0h	Subregion disable bits. For each bit in this field: 0 = corresponding sub-region is enabled 1 = corresponding sub-region is disabled Region sizes of 128 bytes and less do not support subregions. When writing the attributes for such a region, write the SRD field as 0x00. Reset type: CM.SYSRESETn
7-6	RESERVED	R	0h	Reserved

Table 41-187. MPU_RASR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-1	SIZE	R/W	0h	Specifies the size of the MPU protection region. The SIZE field defines the size of the MPU memory region specified by the RNR. as follows: (Region size in bytes) = 2(SIZE+1) Example SIZE field values: 0b00100 (4) 32B 5 Minimum permitted size 0b01001 (9) 1KB 10 - 0b10011 (19) 1MB 20 - 0b11101 (29) 1GB 30 - 0b11111 (31) 4GB 32 Maximum possible size Reset type: CM.SYSRESETn
0	ENABLE	R/W	0h	Region enable bit. Reset type: CM.SYSRESETn

41.12.13.6 MPU_RBAR_A1 Register (Offset = DA4h) [Reset = 0h]

MPU_RBAR_A1 is shown in Figure 41-169 and described in Table 41-188.

Return to the Summary Table.

Alias of RBAR

Figure 41-169. MPU_RBAR_A1 Register

31	30	29	28	27	26	25	24
ADDR							
R/W-0h							
23	22	21	20	19	18	17	16
ADDR							
R/W-0h							
15	14	13	12	11	10	9	8
ADDR							
R/W-0h							
7	6	5	4	3	2	1	0
ADDR			VALID	REGION			
R/W-0h			R/W-0h	R/W-0h			

Table 41-188. MPU_RBAR_A1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	ADDR	R/W	0h	Region base address field. The ADDR field is bits[31:N] of the MPU_RBAR. The region size, as specified by the SIZE field in the MPU_RASR, defines the value of N: N = Log2(Region size in bytes), If the region size is configured to 4GB, in the MPU_RASR, there is no valid ADDR field. In this case, the region occupies the complete memory map, and the base address is 0x00000000. The base address is aligned to the size of the region. For example, a 64KB region must be aligned on a multiple of 64KB, for example, at 0x00010000 or 0x00020000. Reset type: CM.SYSRESETn
4	VALID	R/W	0h	MPU Region Number valid bit: Write: 0 = MPU_RNR not changed, and the processor: updates the base address for the region specified in the MPU_RNR ignores the value of the REGION field 1 = the processor: updates the value of the MPU_RNR to the value of the REGION field updates the base address for the region specified in the REGION field. Always reads as zero. Reset type: CM.SYSRESETn
3-0	REGION	R/W	0h	MPU region field: For the behavior on writes, see the description of the VALID field. On reads, returns the current region number, as specified by the RNR. Reset type: CM.SYSRESETn

41.12.13.7 MPU_RASR_A1 Register (Offset = DA8h) [Reset = 0h]

MPU_RASR_A1 is shown in Figure 41-170 and described in Table 41-189.

Return to the Summary Table.

Alias of RASR

Figure 41-170. MPU_RASR_A1 Register

31	30	29	28	27	26	25	24
RESERVED			XN	RESERVED	AP		
R-0h			R/W-0h	R-0h	R/W-0h		
23	22	21	20	19	18	17	16
RESERVED			TEX		S	C	B
R-0h			R/W-0h		R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
SRD							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED			SIZE				ENABLE
R-0h			R/W-0h				R/W-0h

Table 41-189. MPU_RASR_A1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved
28	XN	R/W	0h	Instruction access disable bit: 0 = instruction fetches enabled 1 = instruction fetches disabled. Reset type: CM.SYSRESETn
27	RESERVED	R	0h	Reserved
26-24	AP	R/W	0h	AP[2:0] Privileged permissions Unprivileged permissions Description 000 No access No access All accesses generate a permission fault 001 RW No access Access from privileged software only 010 RW RO Writes by unprivileged software generate a permission fault 011 RW RW Full access 100 Unpredictable Unpredictable Reserved 101 RO No access Reads by privileged software only 110 RO RO Read only, by privileged or unprivileged software 111 RO RO Read only, by privileged or unprivileged software Reset type: CM.SYSRESETn
23-22	RESERVED	R	0h	Reserved

Table 41-189. MPU_RASR_A1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-19	TEX	R/W	0h	TEX C B S Memory type Shareability Other attributes 0b000 0 0 x Strongly-ordered Shareable - 0 1 x Device Shareable - 1 0 0 Normal Not shareable Outer and inner write-through. No write allocate. 1 0 1 Shareable 1 1 0 Normal Not shareable Outer and inner write-back. No write allocate. 1 1 1 Shareable 0b001 0 0 0 Normal Not shareable Outer and inner noncacheable. 0 0 1 Shareable 0 1 x Reserved encoding - 1 0 x Implementation defined attributes. - 1 1 0 Normal Not shareable Outer and inner write-back. Write and read allocate. 1 1 1 Shareable 0b010 0 0 x Device Not shareable Nonshared Device. 0 1 x Reserved encoding - 1 x x Reserved encoding - 0b1BB A A 0 Normal Not shareable Cached memory, BB = outer policy, AA = inner policy. Encoding, AA or BB Corresponding cache policy 00 Non-cacheable 01 Write back, write and read allocate 10 Write through, no write allocate 11 Write back, no write allocate 1 Shareable and BB bits. Reset type: CM.SYSRESETn
18	S	R/W	0h	TEX C B S Memory type Shareability Other attributes 0b000 0 0 x Strongly-ordered Shareable - 0 1 x Device Shareable - 1 0 0 Normal Not shareable Outer and inner write-through. No write allocate. 1 0 1 Shareable 1 1 0 Normal Not shareable Outer and inner write-back. No write allocate. 1 1 1 Shareable 0b001 0 0 0 Normal Not shareable Outer and inner noncacheable. 0 0 1 Shareable 0 1 x Reserved encoding - 1 0 x Implementation defined attributes. - 1 1 0 Normal Not shareable Outer and inner write-back. Write and read allocate. 1 1 1 Shareable 0b010 0 0 x Device Not shareable Nonshared Device. 0 1 x Reserved encoding - 1 x x Reserved encoding - 0b1BB A A 0 Normal Not shareable Cached memory, BB = outer policy, AA = inner policy. Encoding, AA or BB Corresponding cache policy 00 Non-cacheable 01 Write back, write and read allocate 10 Write through, no write allocate 11 Write back, no write allocate 1 Shareable and BB bits. Reset type: CM.SYSRESETn

Table 41-189. MPU_RASR_A1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	C	R/W	0h	TEX C B S Memory type Shareability Other attributes 0b000 0 0 x Strongly-ordered Shareable - 0 1 x Device Shareable - 1 0 0 Normal Not shareable Outer and inner write-through. No write allocate. 1 0 1 Shareable 1 1 0 Normal Not shareable Outer and inner write-back. No write allocate. 1 1 1 Shareable 0b001 0 0 0 Normal Not shareable Outer and inner noncacheable. 0 0 1 Shareable 0 1 x Reserved encoding - 1 0 x Implementation defined attributes. - 1 1 0 Normal Not shareable Outer and inner write-back. Write and read allocate. 1 1 1 Shareable 0b010 0 0 x Device Not shareable Nonshared Device. 0 1 x Reserved encoding - 1 x x Reserved encoding - 0b1BB A A 0 Normal Not shareable Cached memory, BB = outer policy, AA = inner policy. Encoding, AA or BB Corresponding cache policy 00 Non-cacheable 01 Write back, write and read allocate 10 Write through, no write allocate 11 Write back, no write allocate 1 Shareable and BB bits. Reset type: CM.SYSRESETn
16	B	R/W	0h	TEX C B S Memory type Shareability Other attributes 0b000 0 0 x Strongly-ordered Shareable - 0 1 x Device Shareable - 1 0 0 Normal Not shareable Outer and inner write-through. No write allocate. 1 0 1 Shareable 1 1 0 Normal Not shareable Outer and inner write-back. No write allocate. 1 1 1 Shareable 0b001 0 0 0 Normal Not shareable Outer and inner noncacheable. 0 0 1 Shareable 0 1 x Reserved encoding - 1 0 x Implementation defined attributes. - 1 1 0 Normal Not shareable Outer and inner write-back. Write and read allocate. 1 1 1 Shareable 0b010 0 0 x Device Not shareable Nonshared Device. 0 1 x Reserved encoding - 1 x x Reserved encoding - 0b1BB A A 0 Normal Not shareable Cached memory, BB = outer policy, AA = inner policy. Encoding, AA or BB Corresponding cache policy 00 Non-cacheable 01 Write back, write and read allocate 10 Write through, no write allocate 11 Write back, no write allocate 1 Shareable and BB bits. Reset type: CM.SYSRESETn
15-8	SRD	R/W	0h	Subregion disable bits. For each bit in this field: 0 = corresponding sub-region is enabled 1 = corresponding sub-region is disabled Region sizes of 128 bytes and less do not support subregions. When writing the attributes for such a region, write the SRD field as 0x00. Reset type: CM.SYSRESETn
7-6	RESERVED	R	0h	Reserved

Table 41-189. MPU_RASR_A1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-1	SIZE	R/W	0h	Specifies the size of the MPU protection region. The SIZE field defines the size of the MPU memory region specified by the RNR. as follows: (Region size in bytes) = 2(SIZE+1) Example SIZE field values: 0b00100 (4) 32B 5 Minimum permitted size 0b01001 (9) 1KB 10 - 0b10011 (19) 1MB 20 - 0b11101 (29) 1GB 30 - 0b11111 (31) 4GB 32 Maximum possible size Reset type: CM.SYSRESETn
0	ENABLE	R/W	0h	Region enable bit. Reset type: CM.SYSRESETn

41.12.13.8 MPU_RBAR_A2 Register (Offset = DACh) [Reset = 0h]

MPU_RBAR_A2 is shown in Figure 41-171 and described in Table 41-190.

Return to the Summary Table.

Alias of RBAR

Figure 41-171. MPU_RBAR_A2 Register

31	30	29	28	27	26	25	24
ADDR							
R/W-0h							
23	22	21	20	19	18	17	16
ADDR							
R/W-0h							
15	14	13	12	11	10	9	8
ADDR							
R/W-0h							
7	6	5	4	3	2	1	0
ADDR			VALID	REGION			
R/W-0h			R/W-0h	R/W-0h			

Table 41-190. MPU_RBAR_A2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	ADDR	R/W	0h	Region base address field. The ADDR field is bits[31:N] of the MPU_RBAR. The region size, as specified by the SIZE field in the MPU_RASR, defines the value of N: N = Log2(Region size in bytes), If the region size is configured to 4GB, in the MPU_RASR, there is no valid ADDR field. In this case, the region occupies the complete memory map, and the base address is 0x00000000. The base address is aligned to the size of the region. For example, a 64KB region must be aligned on a multiple of 64KB, for example, at 0x00010000 or 0x00020000. Reset type: CM.SYSRESETn
4	VALID	R/W	0h	MPU Region Number valid bit: Write: 0 = MPU_RNR not changed, and the processor: updates the base address for the region specified in the MPU_RNR ignores the value of the REGION field 1 = the processor: updates the value of the MPU_RNR to the value of the REGION field updates the base address for the region specified in the REGION field. Always reads as zero. Reset type: CM.SYSRESETn
3-0	REGION	R/W	0h	MPU region field: For the behavior on writes, see the description of the VALID field. On reads, returns the current region number, as specified by the RNR. Reset type: CM.SYSRESETn

41.12.13.9 MPU_RASR_A2 Register (Offset = DB0h) [Reset = 0h]

MPU_RASR_A2 is shown in Figure 41-172 and described in Table 41-191.

Return to the Summary Table.

Alias of RASR

Figure 41-172. MPU_RASR_A2 Register

31	30	29	28	27	26	25	24
RESERVED			XN	RESERVED	AP		
R-0h			R/W-0h	R-0h	R/W-0h		
23	22	21	20	19	18	17	16
RESERVED			TEX		S	C	B
R-0h			R/W-0h		R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
SRD							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED		SIZE					ENABLE
R-0h		R/W-0h					R/W-0h

Table 41-191. MPU_RASR_A2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved
28	XN	R/W	0h	Instruction access disable bit: 0 = instruction fetches enabled 1 = instruction fetches disabled. Reset type: CM.SYSRESETn
27	RESERVED	R	0h	Reserved
26-24	AP	R/W	0h	AP[2:0] Privileged permissions Unprivileged permissions Description 000 No access No access All accesses generate a permission fault 001 RW No access Access from privileged software only 010 RW RO Writes by unprivileged software generate a permission fault 011 RW RW Full access 100 Unpredictable Unpredictable Reserved 101 RO No access Reads by privileged software only 110 RO RO Read only, by privileged or unprivileged software 111 RO RO Read only, by privileged or unprivileged software Reset type: CM.SYSRESETn
23-22	RESERVED	R	0h	Reserved

Table 41-191. MPU_RASR_A2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-19	TEX	R/W	0h	TEX C B S Memory type Shareability Other attributes 0b000 0 0 x Strongly-ordered Shareable - 0 1 x Device Shareable - 1 0 0 Normal Not shareable Outer and inner write-through. No write allocate. 1 0 1 Shareable 1 1 0 Normal Not shareable Outer and inner write-back. No write allocate. 1 1 1 Shareable 0b001 0 0 0 Normal Not shareable Outer and inner noncacheable. 0 0 1 Shareable 0 1 x Reserved encoding - 1 0 x Implementation defined attributes. - 1 1 0 Normal Not shareable Outer and inner write-back. Write and read allocate. 1 1 1 Shareable 0b010 0 0 x Device Not shareable Nonshared Device. 0 1 x Reserved encoding - 1 x x Reserved encoding - 0b1BB A A 0 Normal Not shareable Cached memory, BB = outer policy, AA = inner policy. Encoding, AA or BB Corresponding cache policy 00 Non-cacheable 01 Write back, write and read allocate 10 Write through, no write allocate 11 Write back, no write allocate 1 Shareable and BB bits. Reset type: CM.SYSRESETn
18	S	R/W	0h	TEX C B S Memory type Shareability Other attributes 0b000 0 0 x Strongly-ordered Shareable - 0 1 x Device Shareable - 1 0 0 Normal Not shareable Outer and inner write-through. No write allocate. 1 0 1 Shareable 1 1 0 Normal Not shareable Outer and inner write-back. No write allocate. 1 1 1 Shareable 0b001 0 0 0 Normal Not shareable Outer and inner noncacheable. 0 0 1 Shareable 0 1 x Reserved encoding - 1 0 x Implementation defined attributes. - 1 1 0 Normal Not shareable Outer and inner write-back. Write and read allocate. 1 1 1 Shareable 0b010 0 0 x Device Not shareable Nonshared Device. 0 1 x Reserved encoding - 1 x x Reserved encoding - 0b1BB A A 0 Normal Not shareable Cached memory, BB = outer policy, AA = inner policy. Encoding, AA or BB Corresponding cache policy 00 Non-cacheable 01 Write back, write and read allocate 10 Write through, no write allocate 11 Write back, no write allocate 1 Shareable and BB bits. Reset type: CM.SYSRESETn

Table 41-191. MPU_RASR_A2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	C	R/W	0h	TEX C B S Memory type Shareability Other attributes 0b000 0 0 x Strongly-ordered Shareable - 0 1 x Device Shareable - 1 0 0 Normal Not shareable Outer and inner write-through. No write allocate. 1 0 1 Shareable 1 1 0 Normal Not shareable Outer and inner write-back. No write allocate. 1 1 1 Shareable 0b001 0 0 0 Normal Not shareable Outer and inner noncacheable. 0 0 1 Shareable 0 1 x Reserved encoding - 1 0 x Implementation defined attributes. - 1 1 0 Normal Not shareable Outer and inner write-back. Write and read allocate. 1 1 1 Shareable 0b010 0 0 x Device Not shareable Nonshared Device. 0 1 x Reserved encoding - 1 x x Reserved encoding - 0b1BB A A 0 Normal Not shareable Cached memory, BB = outer policy, AA = inner policy. Encoding, AA or BB Corresponding cache policy 00 Non-cacheable 01 Write back, write and read allocate 10 Write through, no write allocate 11 Write back, no write allocate 1 Shareable and BB bits. Reset type: CM.SYSRESETn
16	B	R/W	0h	TEX C B S Memory type Shareability Other attributes 0b000 0 0 x Strongly-ordered Shareable - 0 1 x Device Shareable - 1 0 0 Normal Not shareable Outer and inner write-through. No write allocate. 1 0 1 Shareable 1 1 0 Normal Not shareable Outer and inner write-back. No write allocate. 1 1 1 Shareable 0b001 0 0 0 Normal Not shareable Outer and inner noncacheable. 0 0 1 Shareable 0 1 x Reserved encoding - 1 0 x Implementation defined attributes. - 1 1 0 Normal Not shareable Outer and inner write-back. Write and read allocate. 1 1 1 Shareable 0b010 0 0 x Device Not shareable Nonshared Device. 0 1 x Reserved encoding - 1 x x Reserved encoding - 0b1BB A A 0 Normal Not shareable Cached memory, BB = outer policy, AA = inner policy. Encoding, AA or BB Corresponding cache policy 00 Non-cacheable 01 Write back, write and read allocate 10 Write through, no write allocate 11 Write back, no write allocate 1 Shareable and BB bits. Reset type: CM.SYSRESETn
15-8	SRD	R/W	0h	Subregion disable bits. For each bit in this field: 0 = corresponding sub-region is enabled 1 = corresponding sub-region is disabled Region sizes of 128 bytes and less do not support subregions. When writing the attributes for such a region, write the SRD field as 0x00. Reset type: CM.SYSRESETn
7-6	RESERVED	R	0h	Reserved

Table 41-191. MPU_RASR_A2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-1	SIZE	R/W	0h	Specifies the size of the MPU protection region. The SIZE field defines the size of the MPU memory region specified by the RNR. as follows: (Region size in bytes) = 2(SIZE+1) Example SIZE field values: 0b00100 (4) 32B 5 Minimum permitted size 0b01001 (9) 1KB 10 - 0b10011 (19) 1MB 20 - 0b11101 (29) 1GB 30 - 0b11111 (31) 4GB 32 Maximum possible size Reset type: CM.SYSRESETn
0	ENABLE	R/W	0h	Region enable bit. Reset type: CM.SYSRESETn

41.12.13.10 MPU_RBAR_A3 Register (Offset = DB4h) [Reset = 0h]

MPU_RBAR_A3 is shown in Figure 41-173 and described in Table 41-192.

Return to the Summary Table.

Alias of RBAR

Figure 41-173. MPU_RBAR_A3 Register

31	30	29	28	27	26	25	24
ADDR							
R/W-0h							
23	22	21	20	19	18	17	16
ADDR							
R/W-0h							
15	14	13	12	11	10	9	8
ADDR							
R/W-0h							
7	6	5	4	3	2	1	0
ADDR			VALID	REGION			
R/W-0h			R/W-0h	R/W-0h			

Table 41-192. MPU_RBAR_A3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	ADDR	R/W	0h	Region base address field. The ADDR field is bits[31:N] of the MPU_RBAR. The region size, as specified by the SIZE field in the MPU_RASR, defines the value of N: N = Log2(Region size in bytes), If the region size is configured to 4GB, in the MPU_RASR, there is no valid ADDR field. In this case, the region occupies the complete memory map, and the base address is 0x00000000. The base address is aligned to the size of the region. For example, a 64KB region must be aligned on a multiple of 64KB, for example, at 0x00010000 or 0x00020000. Reset type: CM.SYSRESETn
4	VALID	R/W	0h	MPU Region Number valid bit: Write: 0 = MPU_RNR not changed, and the processor: updates the base address for the region specified in the MPU_RNR ignores the value of the REGION field 1 = the processor: updates the value of the MPU_RNR to the value of the REGION field updates the base address for the region specified in the REGION field. Always reads as zero. Reset type: CM.SYSRESETn
3-0	REGION	R/W	0h	MPU region field: For the behavior on writes, see the description of the VALID field. On reads, returns the current region number, as specified by the RNR. Reset type: CM.SYSRESETn

41.12.13.11 MPU_RASR_A3 Register (Offset = DB8h) [Reset = 0h]

MPU_RASR_A3 is shown in Figure 41-174 and described in Table 41-193.

Return to the Summary Table.

Alias of RASR

Figure 41-174. MPU_RASR_A3 Register

31	30	29	28	27	26	25	24
RESERVED			XN	RESERVED	AP		
R-0h			R/W-0h	R-0h	R/W-0h		
23	22	21	20	19	18	17	16
RESERVED			TEX		S	C	B
R-0h			R/W-0h		R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
SRD							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED		SIZE					ENABLE
R-0h		R/W-0h					R/W-0h

Table 41-193. MPU_RASR_A3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved
28	XN	R/W	0h	Instruction access disable bit: 0 = instruction fetches enabled 1 = instruction fetches disabled. Reset type: CM.SYSRESETn
27	RESERVED	R	0h	Reserved
26-24	AP	R/W	0h	AP[2:0] Privileged permissions Unprivileged permissions Description 000 No access No access All accesses generate a permission fault 001 RW No access Access from privileged software only 010 RW RO Writes by unprivileged software generate a permission fault 011 RW RW Full access 100 Unpredictable Unpredictable Reserved 101 RO No access Reads by privileged software only 110 RO RO Read only, by privileged or unprivileged software 111 RO RO Read only, by privileged or unprivileged software Reset type: CM.SYSRESETn
23-22	RESERVED	R	0h	Reserved

Table 41-193. MPU_RASR_A3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-19	TEX	R/W	0h	TEX C B S Memory type Shareability Other attributes 0b000 0 0 x Strongly-ordered Shareable - 0 1 x Device Shareable - 1 0 0 Normal Not shareable Outer and inner write-through. No write allocate. 1 0 1 Shareable 1 1 0 Normal Not shareable Outer and inner write-back. No write allocate. 1 1 1 Shareable 0b001 0 0 0 Normal Not shareable Outer and inner noncacheable. 0 0 1 Shareable 0 1 x Reserved encoding - 1 0 x Implementation defined attributes. - 1 1 0 Normal Not shareable Outer and inner write-back. Write and read allocate. 1 1 1 Shareable 0b010 0 0 x Device Not shareable Nonshared Device. 0 1 x Reserved encoding - 1 x x Reserved encoding - 0b1BB A A 0 Normal Not shareable Cached memory, BB = outer policy, AA = inner policy. Encoding, AA or BB Corresponding cache policy 00 Non-cacheable 01 Write back, write and read allocate 10 Write through, no write allocate 11 Write back, no write allocate 1 Shareable and BB bits. Reset type: CM.SYSRESETn
18	S	R/W	0h	TEX C B S Memory type Shareability Other attributes 0b000 0 0 x Strongly-ordered Shareable - 0 1 x Device Shareable - 1 0 0 Normal Not shareable Outer and inner write-through. No write allocate. 1 0 1 Shareable 1 1 0 Normal Not shareable Outer and inner write-back. No write allocate. 1 1 1 Shareable 0b001 0 0 0 Normal Not shareable Outer and inner noncacheable. 0 0 1 Shareable 0 1 x Reserved encoding - 1 0 x Implementation defined attributes. - 1 1 0 Normal Not shareable Outer and inner write-back. Write and read allocate. 1 1 1 Shareable 0b010 0 0 x Device Not shareable Nonshared Device. 0 1 x Reserved encoding - 1 x x Reserved encoding - 0b1BB A A 0 Normal Not shareable Cached memory, BB = outer policy, AA = inner policy. Encoding, AA or BB Corresponding cache policy 00 Non-cacheable 01 Write back, write and read allocate 10 Write through, no write allocate 11 Write back, no write allocate 1 Shareable and BB bits. Reset type: CM.SYSRESETn

Table 41-193. MPU_RASR_A3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	C	R/W	0h	TEX C B S Memory type Shareability Other attributes 0b000 0 0 x Strongly-ordered Shareable - 0 1 x Device Shareable - 1 0 0 Normal Not shareable Outer and inner write-through. No write allocate. 1 0 1 Shareable 1 1 0 Normal Not shareable Outer and inner write-back. No write allocate. 1 1 1 Shareable 0b001 0 0 0 Normal Not shareable Outer and inner noncacheable. 0 0 1 Shareable 0 1 x Reserved encoding - 1 0 x Implementation defined attributes. - 1 1 0 Normal Not shareable Outer and inner write-back. Write and read allocate. 1 1 1 Shareable 0b010 0 0 x Device Not shareable Nonshared Device. 0 1 x Reserved encoding - 1 x x Reserved encoding - 0b1BB A A 0 Normal Not shareable Cached memory, BB = outer policy, AA = inner policy. Encoding, AA or BB Corresponding cache policy 00 Non-cacheable 01 Write back, write and read allocate 10 Write through, no write allocate 11 Write back, no write allocate 1 Shareable and BB bits. Reset type: CM.SYSRESETn
16	B	R/W	0h	TEX C B S Memory type Shareability Other attributes 0b000 0 0 x Strongly-ordered Shareable - 0 1 x Device Shareable - 1 0 0 Normal Not shareable Outer and inner write-through. No write allocate. 1 0 1 Shareable 1 1 0 Normal Not shareable Outer and inner write-back. No write allocate. 1 1 1 Shareable 0b001 0 0 0 Normal Not shareable Outer and inner noncacheable. 0 0 1 Shareable 0 1 x Reserved encoding - 1 0 x Implementation defined attributes. - 1 1 0 Normal Not shareable Outer and inner write-back. Write and read allocate. 1 1 1 Shareable 0b010 0 0 x Device Not shareable Nonshared Device. 0 1 x Reserved encoding - 1 x x Reserved encoding - 0b1BB A A 0 Normal Not shareable Cached memory, BB = outer policy, AA = inner policy. Encoding, AA or BB Corresponding cache policy 00 Non-cacheable 01 Write back, write and read allocate 10 Write through, no write allocate 11 Write back, no write allocate 1 Shareable and BB bits. Reset type: CM.SYSRESETn
15-8	SRD	R/W	0h	Subregion disable bits. For each bit in this field: 0 = corresponding sub-region is enabled 1 = corresponding sub-region is disabled Region sizes of 128 bytes and less do not support subregions. When writing the attributes for such a region, write the SRD field as 0x00. Reset type: CM.SYSRESETn
7-6	RESERVED	R	0h	Reserved

Table 41-193. MPU_RASR_A3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-1	SIZE	R/W	0h	Specifies the size of the MPU protection region. The SIZE field defines the size of the MPU memory region specified by the RNR. as follows: (Region size in bytes) = 2(SIZE+1) Example SIZE field values: 0b00100 (4) 32B 5 Minimum permitted size 0b01001 (9) 1KB 10 - 0b10011 (19) 1MB 20 - 0b11101 (29) 1GB 30 - 0b11111 (31) 4GB 32 Maximum possible size Reset type: CM.SYSRESETn
0	ENABLE	R/W	0h	Region enable bit. Reset type: CM.SYSRESETn

41.12.14 CM_WD_REGS Registers

Table 41-194 lists the memory-mapped registers for the CM_WD_REGS registers. All register offset addresses not listed in Table 41-194 should be considered as reserved locations and the register contents should not be modified.

Table 41-194. CM_WD_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	SCSR	System Control & Status Register		Go
4h	WDCNTR	Watchdog Counter Register		Go
8h	WDKEY	Watchdog Reset Key Register		Go
Ch	WDCR	Watchdog Control Register		Go
10h	WDWCR	Watchdog Windowed Control Register		Go

Complex bit access types are encoded to fit into small table cells. Table 41-195 shows the codes that are used for access types in this section.

Table 41-195. CM_WD_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

41.12.14.1 SCSR Register (Offset = 0h) [Reset = 3h]

SCSR is shown in Figure 41-175 and described in Table 41-196.

Return to the Summary Table.

System Control & Status Register

Figure 41-175. SCSR Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
rsvd							
R-0-0h							
7	6	5	4	3	2	1	0
rsvd					RESERVED	WDENINT	WDOVERRIDE
R-0-0h					R-0h	R-1h	R/W1S-1h

Table 41-196. SCSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	A write to this register shall only succeed if a value of 0x1234 is written to this field, else the write will be ignored Reset type: CM.RESETn
15-3	rsvd	R-0	0h	Reset type: CM.RESETn
2	RESERVED	R	0h	Reserved
1	WDENINT	R	1h	This bit is 1 at reset and cannot be modified. On every watchdog event (overflow, key sequence write outside the window, incorrect WDCHK value) an NMI will be fired to CM4. Reset type: CM.RESETn
0	WDOVERRIDE	R/W1S	1h	If this bit is set to 1, the user is allowed to change the state of the Watchdog disable (WDDIS) bit in the Watchdog Control (WDCR) register (refer to Watchdog Block section of this spec). If the WDOVERRIDE bit is cleared, by writing a 1 the WDDIS bit cannot be modified by the user. Writing a 0 will have no effect. If this bit is cleared, then it will remain in this state until a reset occurs. The current state of this bit is readable by the user. Reset type: CM.RESETn

41.12.14.2 WDCNTR Register (Offset = 4h) [Reset = 0h]

WDCNTR is shown in Figure 41-176 and described in Table 41-197.

Return to the Summary Table.

Watchdog Counter Register

Figure 41-176. WDCNTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd																								WDCNTR							
R-0-0h																								R-0h							

Table 41-197. WDCNTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	rsvd	R-0	0h	Reset type: CM.RESETn
7-0	WDCNTR	R	0h	These bits contain the current value of the WD counter. The 8-bit counter continually increments at the WDCLK rate. If the counter overflows, a NMI is generated. If the WDKEY register is written with a valid combination within the watchdog window, then the counter is reset to zero. If the WDKEY register is written with a valid combination outside the watchdog window, then a NMI is generated. Reset type: CM.RESETn

41.12.14.3 WDKEY Register (Offset = 8h) [Reset = 0h]

WDKEY is shown in Figure 41-177 and described in Table 41-198.

Return to the Summary Table.

Watchdog Reset Key Register

Figure 41-177. WDKEY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd																								WDKEY							
R-0-0h																								R/W-0h							

Table 41-198. WDKEY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	rsvd	R-0	0h	Reset type: CM.RESETn
7-0	WDKEY	R/W	0h	Writing 0x55 followed by 0xAA will cause the WDCNTR bits to be cleared. Note.. [1] Reads from the WDKEY return the value of WDCR register. Reset type: CM.RESETn

41.12.14.4 WDCR Register (Offset = Ch) [Reset = 40h]

WDCR is shown in Figure 41-178 and described in Table 41-199.

Return to the Summary Table.

Watchdog Control Register

Figure 41-178. WDCR Register

31	30	29	28	27	26	25	24
rsvd							
R-0-0h							
23	22	21	20	19	18	17	16
rsvd							
R-0-0h							
15	14	13	12	11	10	9	8
rsvd				WDPRECLKDIV			
R-0-0h				R/W-0h			
7	6	5	4	3	2	1	0
WDFLG	WDDIS	WDCHK			WDPS		
R/W1S-0h	R/W-1h	R-0/W-0h			R/W-0h		

Table 41-199. WDCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	rsvd	R-0	0h	Reset type: CM.RESETn
11-8	WDPRECLKDIV	R/W	0h	These bits configure the Clock Pre-divider (DIV_N) that feeds clock to prescaler, These bits form upper two bits of the divider 1000 :DIV_N = 2 1001 :DIV_N = 4 1010 :DIV_N = 8 1011 :DIV_N = 16 1100 :DIV_N = 32 1101 :DIV_N = 64 1110 :DIV_N = 128 1111 :DIV_N = 256 0000 :DIV_N = 512 0001 :DIV_N = 1024 0010 :DIV_N = 2048 0011 :DIV_N = 4096 0100 :DIV_N = Reserved 0101 :DIV_N = Reserved 0110 :DIV_N = Reserved 0111 :DIV_N = Reserved All Reserved combinations shall default to DIV_N=512. Reset type: CM.RESETn
7	WDFLG	R/W1S	0h	Watchdog nmi status flag bit. This bit, if set, indicates a NMI was fired to CM4 from WWD. This bit remains latched until the user writes a 1 to clear the condition. Writes of 0 will be ignored. Reset type: CM.RESETn
6	WDDIS	R/W	1h	Writing a 1 to this bit will disable the watchdog module. Writing a 0 will enable the module. This bit can only be modified if the WDOVERRIDE bit in the SCSR register is set to 1. On reset, the watchdog module is disabled. Reset type: CM.RESETn
5-3	WDCHK	R-0/W	0h	The user must ALWAYS write 1,0,1 to these bits whenever a write to this register is performed. Writing any other value will cause an nmi to the CPU (if WD enabled). Reset type: CM.RESETn

Table 41-199. WDCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	WDPS	R/W	0h	These bits configure the watchdog counter clock (WDCLK) rate relative to OSCCLK/512.. 000 WDCLK = OSCCLK/<WDPRECLKDIV>/1 001 WDCLK = OSCCLK/<WDPRECLKDIV>/1 010 WDCLK = OSCCLK/<WDPRECLKDIV>/2 011 WDCLK = OSCCLK/<WDPRECLKDIV>/4 100 WDCLK = OSCCLK/<WDPRECLKDIV>/8 101 WDCLK = OSCCLK/<WDPRECLKDIV>/16 110 WDCLK = OSCCLK/<WDPRECLKDIV>/32 111 WDCLK = OSCCLK/<WDPRECLKDIV>/64 Reset type: CM.RESETn

41.12.14.5 WDWCR Register (Offset = 10h) [Reset = 0h]

WDWCR is shown in Figure 41-179 and described in Table 41-200.

Return to the Summary Table.

Watchdog Windowed Control Register

Figure 41-179. WDWCR Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
rsvd							FIRSTKEY
R-0-0h							R-0h
7	6	5	4	3	2	1	0
MIN							
R/W-0h							

Table 41-200. WDWCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	A write to this register shall only succeed if a value of 0x1234 is written to this field, else the write will be ignored Reset type: CM.RESETn
15-9	rsvd	R-0	0h	Reset type: CM.RESETn
8	FIRSTKEY	R	0h	This bit indicates if the 1st valid WDKEY (0x55 + 0xAA) got detected after MIN was configured to a non-zero value 0.. First Valid Key after non-zero MIN configuration has not happened yet 1.. First Valid key after non-zero MIN configuration got detected Notes.. [1] If MIN = 0, this bit is never set [2] If MIN is written to by software, this bit is auto-cleared [3] This bit is added for debug purposes only Reset type: CM.RESETn
7-0	MIN	R/W	0h	These bits define the lower limt of the Windowed functionality Reset type: CM.RESETn

This page intentionally left blank.

Chapter 42

Advanced Encryption Standard (AES) Accelerator

This section describes the Advanced Encryption Standard (AES) cryptographic hardware-accelerated module.

42.1 Introduction

This section introduces the Advanced Encryption Standard (AES), and describes the AES main functions and connections in the device.

The AES module provides hardware-accelerated data encryption and decryption operations based on a binary key. The AES is a symmetric cipher module that supports a 128-, 192-, or 256-bit key in hardware for encryption and decryption. The AES module is based on a symmetric algorithm, which means that the encryption and decryption keys are identical. To encrypt data means to convert it from plain text to an unintelligible form called cipher text. Decrypting cipher text converts previously encrypted data to its original plain text form. The main features of the AES accelerator are:

AES encrypt and decrypt operations are supported by:

- Galois/Counter mode (GCM), with basic GHASH operation
- Counter mode with CBC-MAC (CCM)
- XTS mode

The following feedback operating modes are available:

- Electronic code book mode (ECB)
- Cipher block chaining mode (CBC)
- Counter mode (CTR)
- Cipher feedback mode (CFB), 128-bit
- F8 mode
- Key sizes: 128, 192, and 256 bits
- Support for CBC_MAC and Fedora 9 (F9) authentication modes
- Basic GHASH operation (when selecting no encryption)
- Key scheduling in hardware
- Support for µDMA transfers
- Fully synchronous design

42.1.1 AES Block Diagram

Figure 42-1 shows the AES block diagram. A single-core or dual-interface architecture is used.

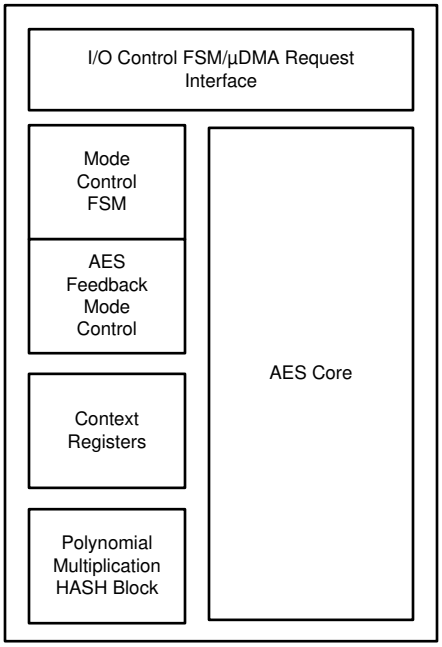


Figure 42-1. AES Block Diagram

AES is an efficient implementation of the Rijndael cipher (the AES algorithm) and a 128-bit polynomial multiplication (referred to here as GHASH, according to the AES-GCM specification). Rijndael is a block cipher in which each data block is 128 bits. The polynomial multiplication multiplies two 128-bit vectors using the smallest 128-bit irreducible polynomial, represented by the following 128-bit string: {0 120}||10000111. The two implementations are combined into the AES wide-bus engine.

Depending on the availability of context and data, the AES wide-bus engine is automatically triggered to process the data. The AES wide-bus engine is directly connected to the context and data registers, so that the AES engine can immediately start processing when all data is available. The AES wide-bus engine also interfaces to the I/O control FSM/µDMA request interface.

AES comprises the following major functional blocks:

- Global control FSM and µDMA interface
- Register interface module
- The AES wide-bus engine

The AES wide-bus engine, which is the major top-level component, comprises the following functional blocks:

- Mode control FSM: Manages the data flow to and from the AES wide-bus engine and starts each encryption or decryption operation
- Feedback modes: The logic that implements the various feedback modes supported by AES.
- GHASH core: The polynomial multiplication algorithm used for AES-GCM
- AES key scheduler: Generates AES encryption and decryption (round) keys
- AES encryption core: The AES encryption algorithm
- AES decryption core: The AES decryption algorithm
- Substitution-boxes (S-Boxes): Contain AES S-Box GF(2^8) implementations

AES encryption requires a specific number of rounds, depending on the key length. The supported key lengths are 128-, 192-, and 256-bit, which require 10, 12, and 14 rounds, respectively, or 32-, 38-, and 44-clock cycles, respectively, because {number of clock cycles} = 2 + 3 × {number of rounds}.

The larger key lengths provide greater encryption strength at the expense of additional rounds, and therefore reduced throughput. The overall throughput of the AES executing polynomial multiplication is adjusted based on the overall cryptographic performance. The AES module contains one ECB core and a dedicated 32-cycle polynomial multiplication module for performing GHASH operations. Polynomial multiplication operates in parallel with the AES core, if data is available for both modules.

Depending on the key size (128, 192, or 256 bits), this core requires 32-, 38-, or 44-clock cycles to process one 128-bit data block. While one data block processes, the next block can be preloaded immediately. When a block is preloaded, the previous block must finish before additional data can be loaded. Therefore, when the pipeline is full, sequential data blocks can be passed every 32-, 38-, or 44-clock cycles.

42.1.1.1 Interfaces

The interface signals to the AES module can be grouped into the following categories:

- Clock enable
- DMA and interrupt interface, used to request new context and packet data or to indicate available result data (encrypted or decrypted data, or authentication result)
- Functional register interface

42.1.1.2 AES Subsystem

The AES subsystem interfaces with the Connectivity Manager µDMA. The three registers include AESDMAINTEN (AES_DMA_Interupt_Enable), AESDMASTATUS (AES_DMA_Interupt_Status), and AESDMASTATUSCLR (AES_DMA_Interupt_Status_Clear). The read-only interrupt status register indicates when a DMA transfer has been completed. These interrupts can be enabled by setting the corresponding bit in the AESDMAINTEN register. When the status register has been read and the interrupt has been serviced, clear the AESDMASTATUS by setting the related AESDMASTATUSCLR bits. The AES subsystem registers are used when initiating DMA transfers. When the DMA is not in use, interrupts are generated.

Table 42-1. AES Subsystem Interrupt Status

Event	Description
AESDMASTATUS[3]: CONTEXT_OUT	Context output interrupt
AESDMASTATUS[2]: DATA_OUT	Data output interrupt
AESDMASTATUS[1]: DATA_IN	Data input interrupt
AESDMASTATUS[0]: CONTEXT_IN	Context input interrupt

42.1.1.3 AES Wide-Bus Engine

The AES wide-bus engine performs the cryptographic operations. The composition of the AES core follows:

AES Key Scheduler

The AES key scheduler generates the round keys. During each round, a new subkey is generated from the input key to be XORed with the data. Round keys are generated arbitrarily and parallel to data processing to minimize register requirements.

For encryption operations, the key sequencer transfers the initial key data to the AES core. For decryption operations, the key scheduler must provide the final subkey to the AES core so the AES can generate the subkeys in reverse order.

AES Encryption Core

The AES encryption core implements the Rijndael algorithm as specified in [FIPS-197]. This core operates on the input block and performs the required substitution, shift, and mix operations. For each round, the encryption core receives the proper round key from the AES key scheduler. A fundamental component of the AES algorithm is the S-Box. The S-Box provides a unique 8-bit output for each 8-bit input. This implementation of the AES encryption core has a 64-bit data path.

AES Decryption Core

The architecture of the AES decryption core is generally the same as the architecture of the encryption core. One difference is that the generation of round keys for decryption requires an initial conversion of the input key (always supplied by the host in the form of an encryption key) to the corresponding decryption key. This conversion is done by performing a dummy encryption operation and storing the final round key as a decryption key. The key scheduler is then reversed to generate the round keys for the decryption operation. Consequently, for each sequence of decryption operations under the same key, a single throughput reduction equal to the time to encrypt a single block occurs. Once a decryption key is generated, subsequent decryption operations with the same key use this generated decryption key directly.

AES Feedback Mode Block

The AES feedback mode block buffers the feedback parameters and controls the various feedback modes. For more information about the ECB, CBC, CTR, and CFB modes of operation, see the NIST-SP800-38A specification.

CTR implements the standard incrementing function, as described in the NIST-SP800-38A specification, with *m* set to 16 or a multiple of 32.

AES-XTS mode requires a polynomial multiplication for initialization vector (IV) generation of the AES operation. This multiplication can be simplified when the first result is available due to the definition and use of the block number within a unit. The input for the polynomial multiplication is not directly j, but α^j, where $\alpha = x^2$ in the GF(2^{128}) domain.

In addition, f8 encryption/decryption mode and f9 and (X)CBC-MAC authentication modes are available.

GHASH Block

The data sequencer manages the data flow to and from the AES core. For data input, the data sequencer monitors the input buffer until a 16-byte block is available. If the AES core is idle, the data sequencer writes this data block to the internal working registers of the AES core, thus clearing the buffer for the next block.

After completing an encryption or decryption operation, the data sequencer writes the AES output to the output buffer. If the output buffer is full at the time of completion, the AES core is held until the buffer clears. Although the data sequencer is designed to support uninterrupted packet encryption, the host must properly manage the input and output packet buffers to achieve designed performance.

42.1.2 AES Algorithm

The AES algorithm generates block ciphers. The AES block size is 16 bytes. The AES keys can be coded on 128, 192, or 256 bits. The larger key sizes provide a higher level of security, but at the cost of a moderate decrease in throughput.

For the AES algorithm:

- The length of the input and output blocks is 128 bits. The block length is represented by Nb = 4, which reflects the number of 32-bit words.
- The length of the cipher key (K) is 128, 192, or 256 bits. The key length is represented by Nk = 4, 6, or 8, which reflects the number of 32-bit words in the cipher key.
- The number of rounds to be performed during the execution of the algorithm depends on the key size. The number of rounds is represented by Nr, where Nr = 10 when Nk = 4 (128-bit key); Nr = 12 when Nk = 6 (192-bit key); and Nr = 14 when Nk = 8 (256-bit key).

Table 42-2 lists the combinations of keys, blocks, and rounds.

Table 42-2. Key-Block-Round Combinations

Key	Key Length (Nk)	Block Size (Nb)	Number of Rounds (Nr)
128 bits	4	4	10
192 bits	6	4	12
256 bits	8	4	14

The AES algorithm for cipher and inverse cipher uses a round function composed of four different byte-oriented transformations:

- Byte substitution using a substitution table (S-Box): This transformation is a nonlinear byte substitution that operates independently on each byte of the state (the state is an intermediate processed block of 128 bits inside the AES; the state is arranged as an array of [4 × Nk] bytes) using an S-Box. This S-Box transformation is reversible.
- Shifting rows of the state array by different offsets: In this transformation, the bytes in the last three rows of the state are cyclically shifted over different numbers of bytes (offsets). The first row ® = 0) is not shifted.
- Mixing the data within each column of the state array: This transformation operates on the state column-by-column, treating each column as a 4-term polynomial. The columns are considered polynomials over GF(2^8) and multiplied modulo $x^4 + 1$ with a fixed polynomial a(x).
- Adding a round key to the state: In this transformation, a round key is added to the state by a simple bitwise XOR operation. Each round key consists of Nb words from the key schedule.

The AES algorithm takes the cipher key (K) and performs a key expansion routine to generate a key schedule. The key expansion generates a total of Nb × (Nr + 1) words: The algorithm requires an initial set of Nb words, and each Nr round requires Nb words of key data. The resulting key schedule consists of a linear array of 4-byte words, denoted [w_I], with I in the range 0 ≤ I ≤ Nb × (Nr + 1).

42.2 AES Operating Modes

42.2.1 GCM Operation

Figure 42-2 shows one round of a GCM operation for encryption and decryption. A 32-bit counter is used as IV (as it is for CTR mode). The data is encrypted in the same way as in CTR mode, by XORing the cryptographic core output with the input. After the encryption/decryption, the ciphertext is XORed with the intermediate authentication result. The XORed result is used as input for the polynomial multiplication to create the next (intermediate) authentication result. For more information about the GCM protocol, see Section 42.3.1.

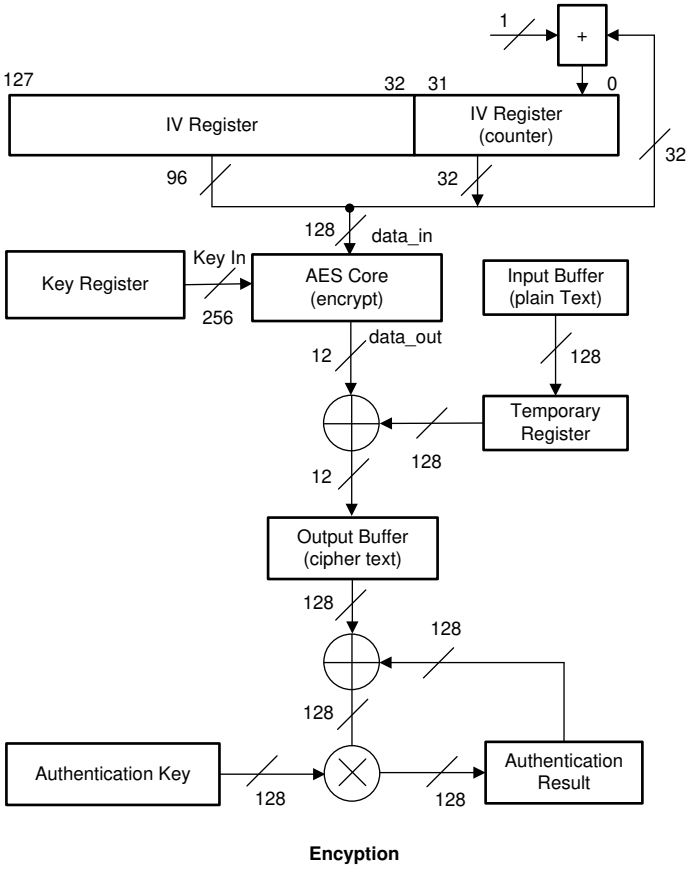


Figure 42-2. AES - GCM Operation

42.2.2 CCM Operation

Figure 42-3 shows one round of a CCM (counter with CBC-MAC) operation for encryption and decryption. A 32-bit counter is used as IV (as it is for CTR mode). The data is encrypted in the same way as in CTR mode, by XORing the cryptographic core output with the input. Immediately after the encryption operation, the plaintext is XORed with the intermediate authentication result. The XOR result is used as input for a second encryption operation to calculate the next (intermediate) authentication result.

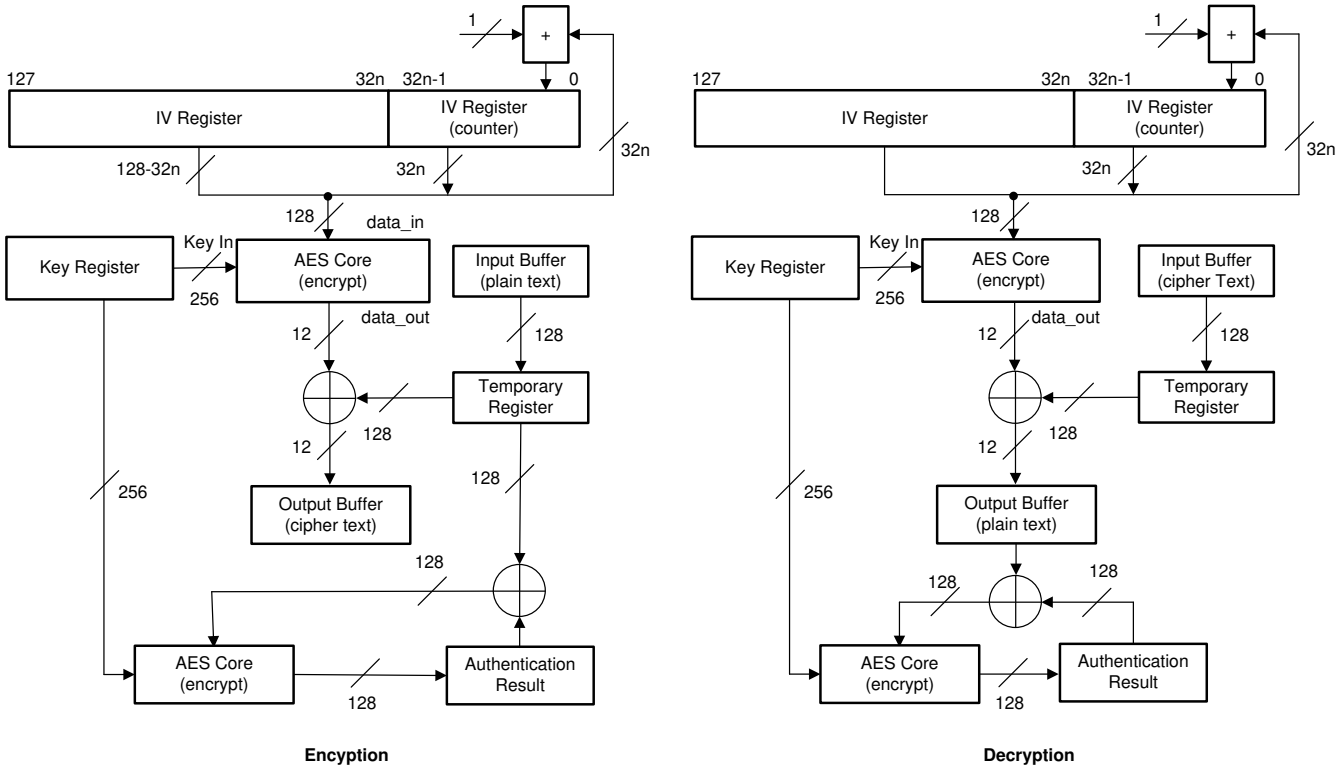


Figure 42-3. AES - CCM Operation

42.2.3 XTS Operation

Figure 42-4 shows the XTS mode of operation for encryption and decryption. The input to the cryptographic core is XORed with the IV; the output of the cryptographic core is XORed with the same IV. For decryption, the cryptographic core operates in reverse, but the XOR operations are the same.

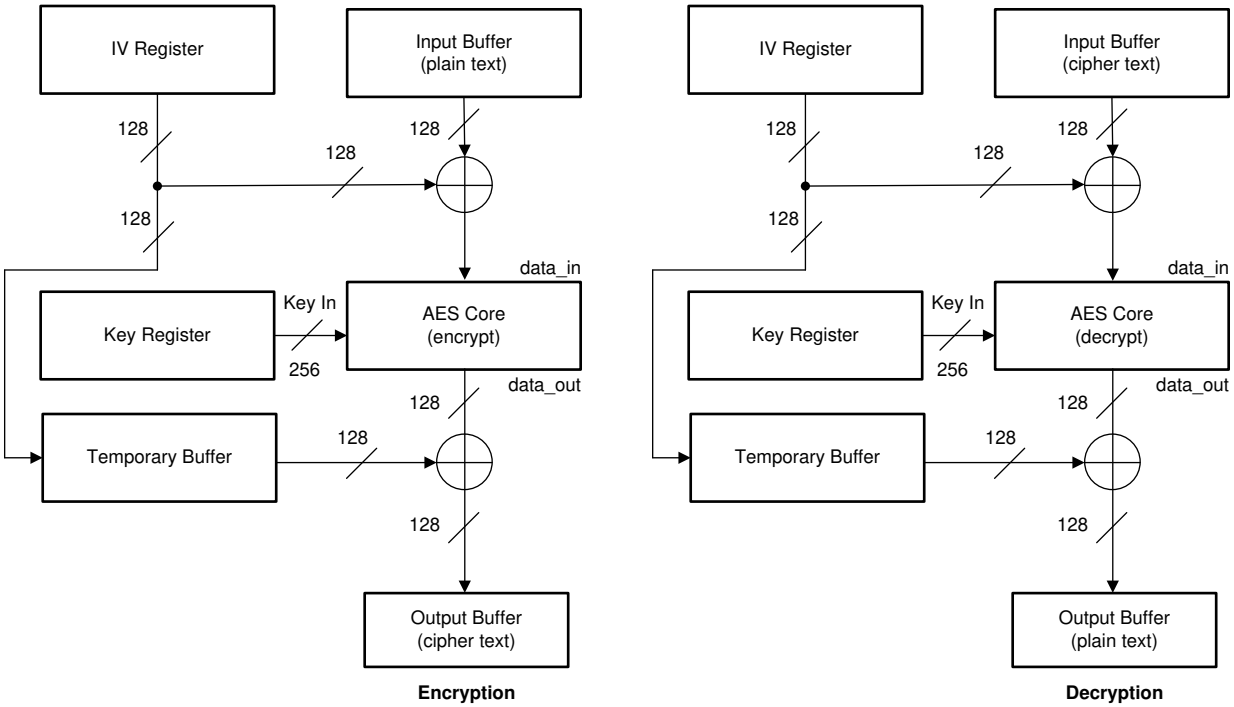


Figure 42-4. AES - XTS Operation

Note

The IV is created with an initial encryption, followed by an LFSR operation for each new block.

42.2.4 ECB Feedback Mode

Figure 42-5 shows the basic ECB feedback mode of operation, where the input data is passed directly to the basic cryptographic core and the output is passed directly to the output buffer. For decryption, the cryptographic core operates in reverse: the decryption data path is used for data processing, whereas encryption uses the encryption data path.

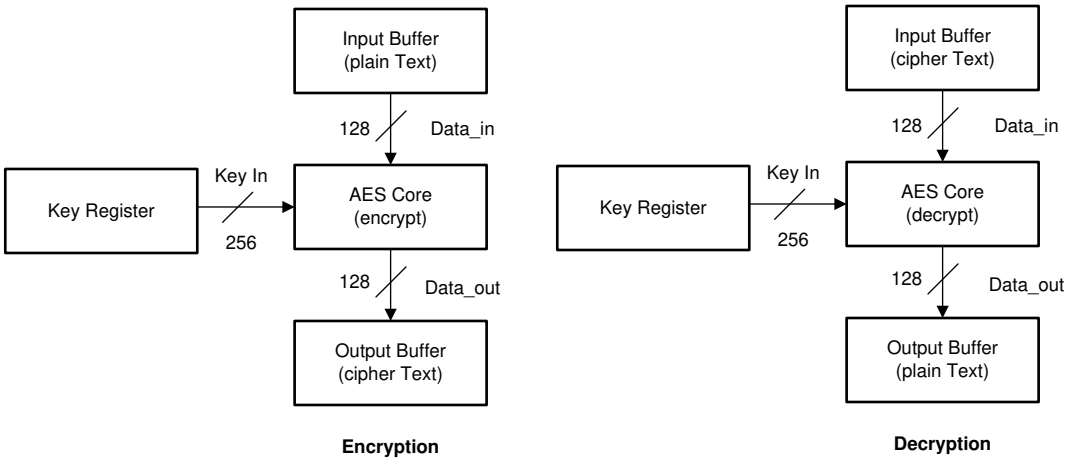


Figure 42-5. AES - ECB Feedback Mode

42.2.5 CBC Feedback Mode

Figure 42-6 shows the CBC feedback mode of operation, where the input data is XORed with the IV before it is passed to the basic cryptographic core. The output of the cryptographic core passes directly to the output buffer and becomes the next IV.

The operation is reversed for decryption, resulting in an XOR at the output of the cryptographic core. The input cipher text of the current operation is the IV for the next operation.

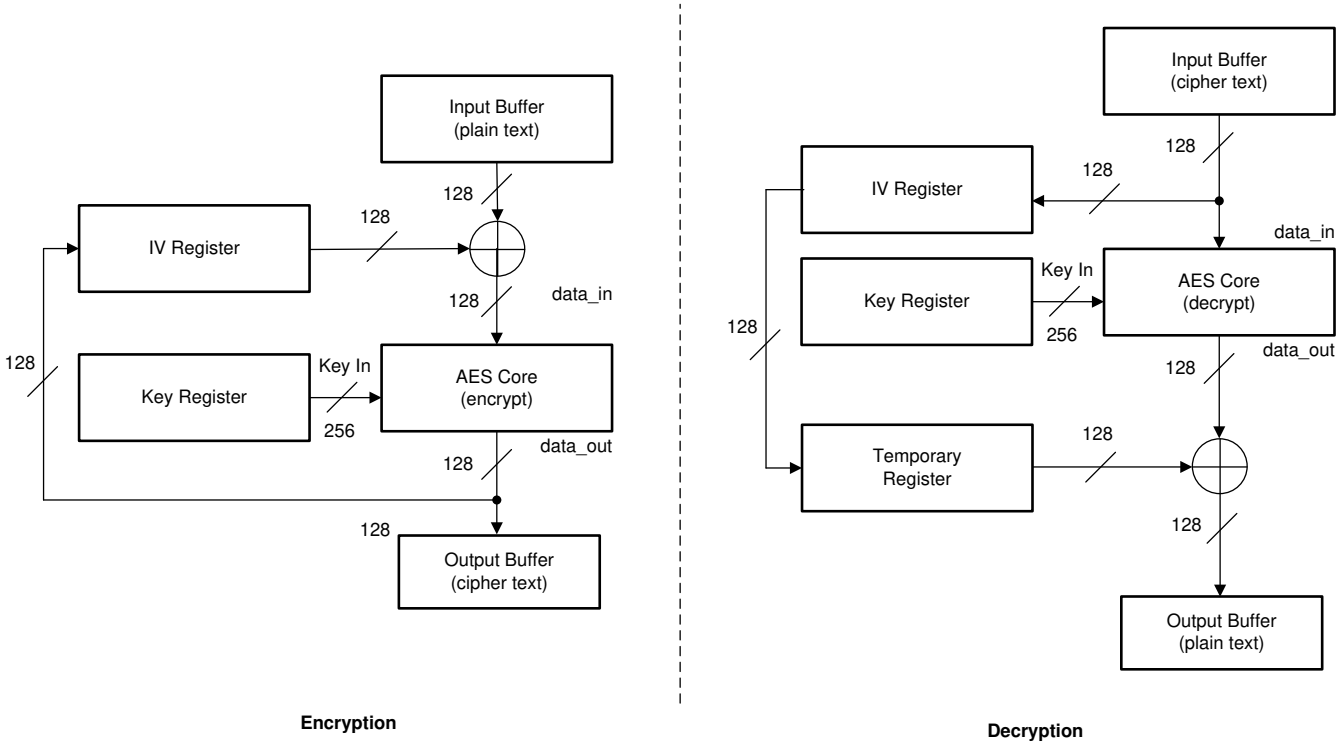


Figure 42-6. AES - CBC Feedback Mode

42.2.6 CTR and ICM Feedback Modes

Figure 42-7 shows the counter feedback (CTR/ICM) mode of operation. This operation encrypts the IV. The output of the cryptographic core (encrypted IV) is XORed with the data, thus creating the output result.

The IV is built out of two components: a fixed part and a counter part. The counter part is incremented with each block. The counter width is selectable per context and can be 16, 32, 64, 96, or 128 bits wide. In this mode, encryption and decryption use the same operation.

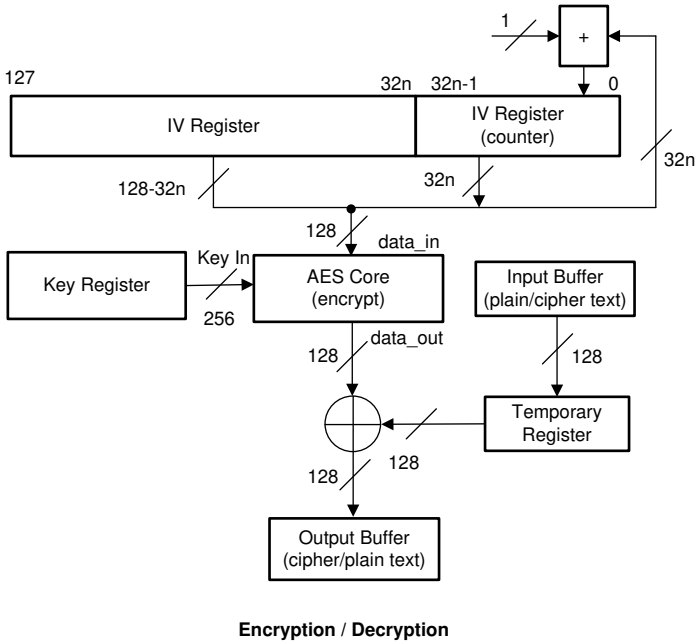


Figure 42-7. AES Encryption With CTR/ICM Mode

Note

The value for n can be 1, 2, 3, or 4 for CTR mode and is ½ for ICM mode.

42.2.7 CFB Mode

Figure 42-8 shows the full block (128 bits) CFB mode of operation for encryption and decryption. The input for the cryptographic core is the IV; the result is XORed with the data. The result is fed back through the IV register as the next input for the cryptographic core. The decryption operation is reversed, but the cryptographic core still performs encryption.

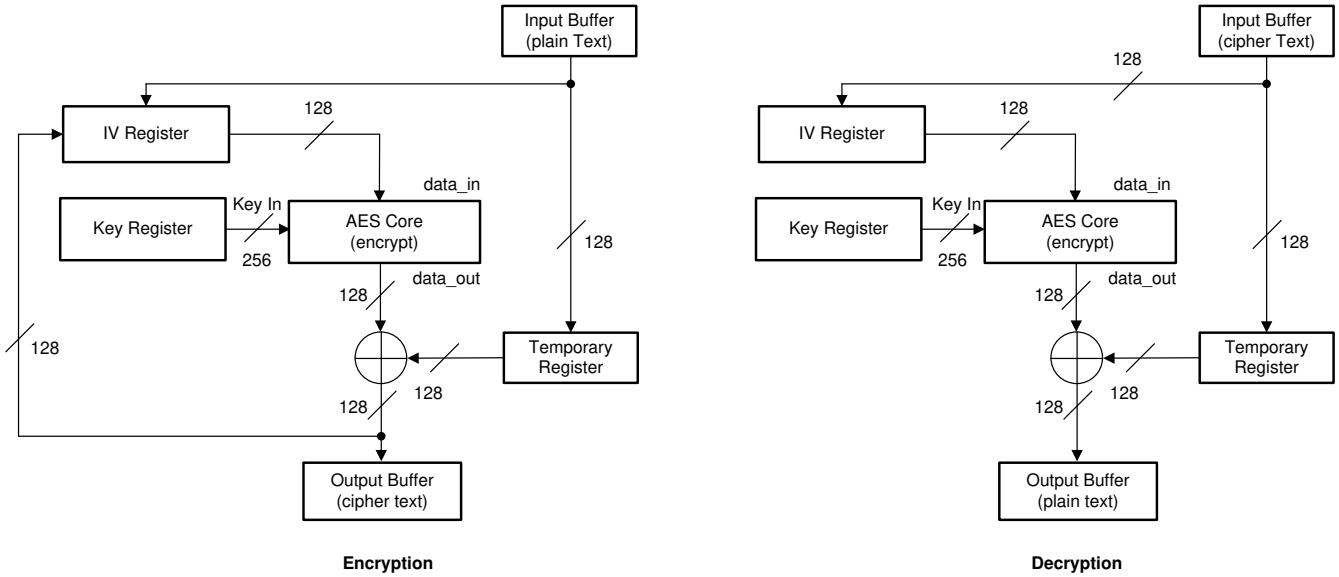


Figure 42-8. AES - CFB Feedback Mode

42.2.8 F8 Mode

Figure 42-9 shows the F8 feedback mode of operation for encryption and decryption. The input to the cryptographic core is the result of the XOR operation of the previous cryptographic core output, a constant IV, and a block counter. The output of the cryptographic core is XORed with the input to create the result. In this mode, encryption and decryption use the same operations.

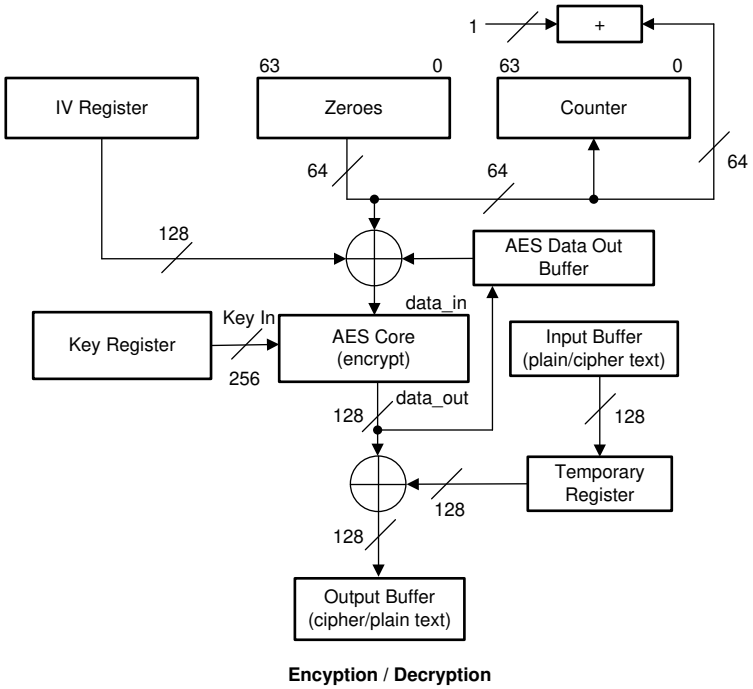


Figure 42-9. AES - F8 Mode

42.2.9 F9 Operation

Figure 42-10 shows the F9 authentication mode of operation, where the input to the cryptographic core is XORed with the IV, and the output is XORed with the previous result to create the next result. The cryptographic core output is fed back as IV for the next block. The result is the output of the last XOR operation of the cryptographic core output.

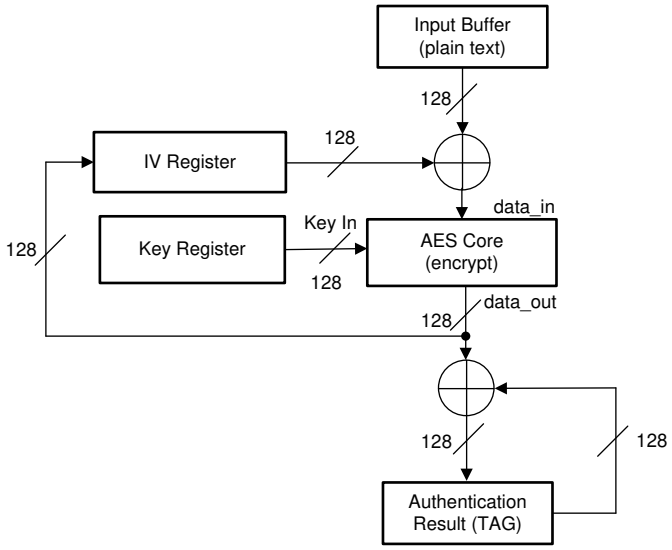


Figure 42-10. AES - F9 Operation

42.2.10 CBC-MAC Operation

Figure 42-11 shows the CBC-MAC authentication mode of operation, where the input to the cryptographic core is XORed with the IV. The cryptographic core output is then fed back as IV for the next block. The last data input block is XORed with an additional input value stored in the temporary buffer; this can be any precalculated value and is dependent on the alignment of the last input block. The result is the cryptographic core output of the last encryption operation.

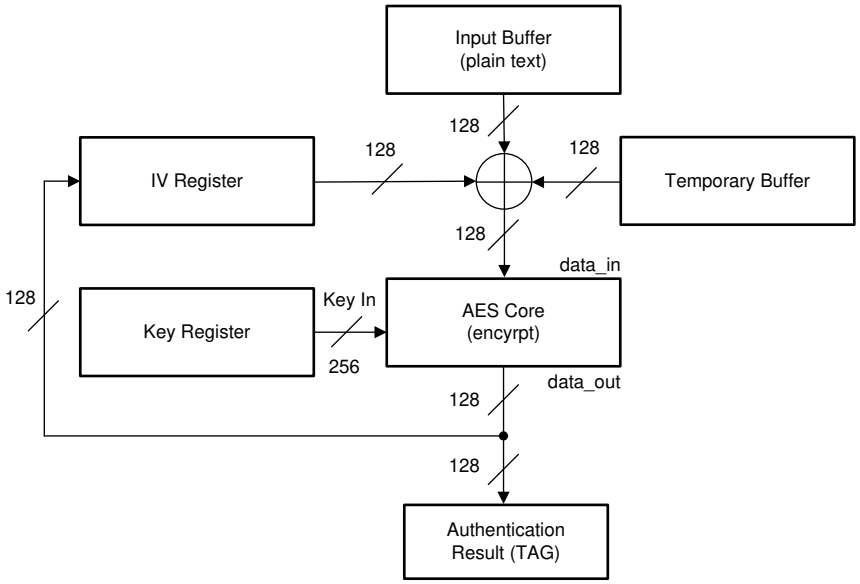


Figure 42-11. AES - CBC-MAC Authentication Mode

42.3 Extended and Combined Modes of Operations

This section describes the protocols (or autonomous precalculations) supported by the AES wide-bus engine.

42.3.1 GCM Protocol Operation

A GCM protocol operation is a combined operation consisting of encryption or decryption, and authentication. A part of the input data stream can be authenticated only, while normally most of the input data is encrypted or decrypted and authenticated. The authentication-only data must always be in front of the data requiring encryption. Within GCM, the authentication-only data is called the additional authentication data (AAD). The AAD is fetched independently of other data.

The intermediate (temporary) result data is used as input to the remaining authentication operation. Because the authentication operation does not require the cryptographic core but only the polynomial multiplication, encryption, decryption, and authentication can be performed in parallel. After encryption of the last data block, additional polynomial multiplication and encryption are required to authenticate a 128-bit-long vector and finally encrypt the authentication result.

42.3.2 CCM Protocol Operation

The CCM protocol operation is a combined operation consisting of encryption or decryption, and authentication. The authentication and encryption or decryption operations use the cryptographic core; these operations are executed sequentially on the AES core. A part of the data stream can require authentication only. The authentication-only data must always be in front of the data requiring encryption.

Authentication starts with the encryption of a predefined block B0. This block consists of flags, nonce, and message length. The next blocks contain the authentication data length concatenated with the authentication-only data. After processing the authentication-only data, the encryption or decryption operations are performed, each followed by the related authentication of the plaintext data block (which equals the input in the case of encryption, and the output in the case of decryption). The final authentication result must be encrypted using the output of the encryption of the IV block A0. This block contains the IV (consisting of flags and nonce) concatenated with the counter, which is zero for A0.

42.3.3 Hardware Requests

The AES module can assert a µDMA request for context in, context out, input data, or output data read. The AES µDMA Interrupt Mask (AES_DMAIM) register can be set to generate interrupts during the following events:

- Context In µDMA request (Cin)
- Context Out µDMA request (Cout)
- Data In µDMA request (Din)
- Data Out µDMA request (Dout)

The AES module can be programmed to assert an interrupt when the µDMA has completed its last transfer.

If context and data transfers are to be handled through software, then the AES Interrupt Enable (AES_IRQENABLE) register can be used to enable interrupt triggering when context out, context in, data in, or data out is ready. The AES Interrupt Status (AES_IRQSTATUS) register indicates when an interrupt is triggered, as listed in Table 42-3.

Table 42-3. Interrupts and Events

Event	Description
AES_IRQSTATUS[3]: CONTEXT_OUT	Context output interrupt
AES_IRQSTATUS[2]: DATA_OUT	Data output interrupt
AES_IRQSTATUS[1]: DATA_IN	Data input interrupt
AES_IRQSTATUS[0]: CONTEXT_IN	Context input interrupt

42.4 AES Module Programming Guide

42.4.1 AES Low-Level Programming Models

This section describes the low-level hardware programming sequences for configuring and using the AES module.

42.4.1.1 Global Initialization

The following list describes the requirements for initializing the AES and associated modules.

1. Configure the AES µDMA channels for Context In, Context Out, Data In, and Data Out by programming the appropriate encoding value in the DMA Channel Map Select n (DMA_CHMAPn) register in the µDMA module. For more information, refer to the *Micro Direct Memory Access (µDMA)* chapter.
2. If the AES channels are configured in the µDMA, enable the required AES DMA requests by programming bits [9:5] of the AES_SYSCONFIG register, in addition to the completion interrupts in the AES DMA Interrupt Mask (DTHE_AES_IM) register.
3. Specify the size of the keys by programming the KEY_SIZE bit field in the AES_CTRL register.
4. Load the AES Key 1 (AES_KEY1_n) register.
5. Load the AES Key 2 (AES_KEY2_n) register if it is used by the configuration mode.
6. Configure the AES for the appropriate encryption or decryption mode (see Section 42.4.1.2).
7. Select encryption or decryption by programming the DIRECTION bit in the AES Control (AES_CTRL) register.

42.4.1.2 AES Operating Modes Configuration

The following sections list the initialization subsequences for the available encryption and decryption modes:

Subsequence: Initialize CCM AES Core Mode

The steps to initialize CCM mode follow:

1. Define the width of the length field and the length of the authentication field by programming the CCM_L and CCM_M bit fields in the AES_CTRL register.
2. Enable counter mode by setting the CTR bit in the AES_CTRL register.
3. Load the authentication data length in the AUTH field of the AES Authentication Data Length (AES_AUTH_LENGTH) register.
4. Select the IV counter by programming the CTR_WIDTH field in the AES_CTRL register.
5. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

Subsequence: Initialize GCM AES Core Mode

The steps to enable GCM mode follow:

1. Enable counter mode by setting the CTR bit in the AES_CTRL register.
2. Load the authentication data length in the AUTH field of the AES Authentication Data Length (AES_AUTH_LENGTH) register.
3. Select the IV counter by programming the CTR_WIDTH field in the AES_CTRL register.
4. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

Subsequence: Initialize CBC-MAC AES Core Mode

The steps to initialize CBC-MAC mode follow:

1. Enable CBC-MAC mode by setting the CBCMAC bit in the AES_CTRL register.
2. Select encryption mode by setting the DIRECTION bit in the AES_CTRL register.
3. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

Subsequence: Initialize F9 AES Core Mode

The steps to configure the AES for F9 mode follow:

1. Enable F9 mode by setting the F9 bit in the AES_CTRL register.
2. Set the key size to 128 bits by programming the KEY_SIZE field to 0x1 in the AES_CTRL register.
3. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

Subsequence: Initialize F8 AES Core Mode

The steps to configure the AES for F8 mode follow:

1. Enable F8 mode by setting the F8 bit in the AES_CTRL register.
2. Select the counter width by programming the CTR_WIDTH field in the AES_CTRL register.
3. Set the key size to 128 bits by setting the KEY_SIZE field to 0x1 in the AES_CTRL register.
4. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

Subsequence: Initialize XTS AES Core Mode

The steps to configure XTS mode follow:

1. Enable XTS mode by configuring the XTS field in the AES_CTRL register.
2. If the XTS field in the AES_CTRL register indicates that the AAD length is required, load the AAD length in the AES_AUTH_LENGTH register.
3. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

Subsequence: Initialize CFB AES Core Mode

The steps to initialize the AES code for CFB mode follow:

1. Enable CFB mode by setting the CFB bit in the AES_CTRL register.
2. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

Subsequence: Initialize ICM AES Core Mode

The steps to initialize the AES code for ICM mode follow:

1. Enable ICM mode by setting the ICM bit in the AES_CTRL register.
2. Configure for a 16-bit counter by programming the CTR_WIDTH field to 0x0 in the AES_CTRL register.
3. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

Subsequence: Initialize CTR AES Core Mode

The steps to initialize CTR mode follow:

1. Enable CTR mode by setting the CTR bit in the AES_CTRL register.
2. Select counter width by programming the CTR_WIDTH in the AES_CTRL register.
3. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

Subsequence: Initialize CBC AES Core Mode

The steps to configure CBC mode follow:

1. Enable CBC mode by setting the MODE bit in the AES_CTRL register.
2. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

42.4.1.3 AES Mode Configurations

AES Polling Mode

Main Sequence: AES Polling Mode – Figure 42-12 shows AES polling mode. The registers used in AES polling mode follow:

- AES Data RW Plaintext/Ciphertext 0 (AES_DATA_IN_OUT_0) registers
- AES Control (AES_CTRL) register
- AES Hash Tag Out 0 (AES_TAG_OUT_0) register

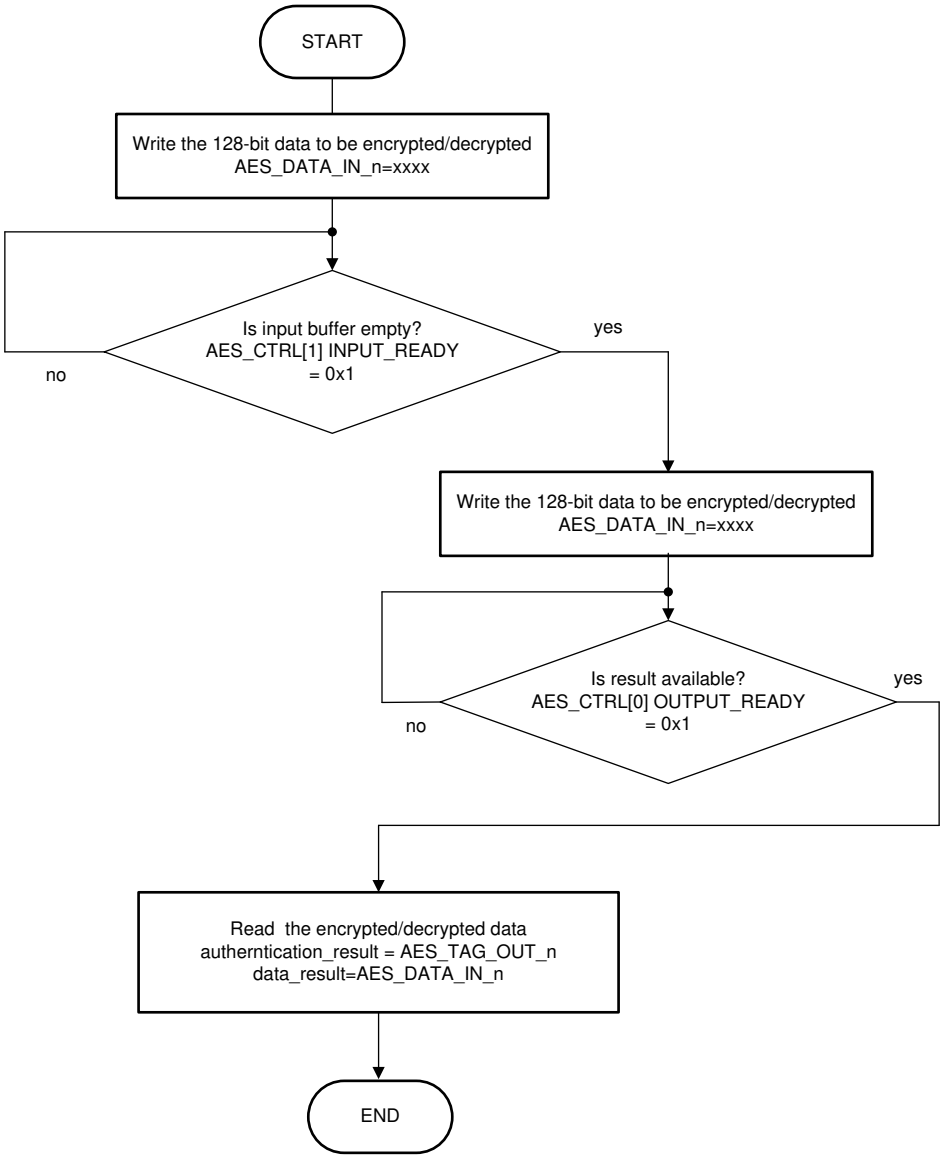


Figure 42-12. AES Polling Mode

AES Interrupt Mode

The application can use software interrupts to control the flow of Context In, Context Out, Data In, and Data Out requests. To enable these interrupts

1. First, initialize the device by following the initialization sequences described in Section 42.4.1.1 and Section 42.4.1.2.
2. When the device has been initialized, the application can enable the AES module interrupts through the AES Interrupt Enable (AES_IRQENABLE) register.
3. Load the input buffers, AES_DATA_IN_OUT_n, with data.

Note

If the application uses interrupt mode, an interrupt is generated for each block of processed data. To support larger data flow, AES µDMA mode must be used and the bits in the AES_IRQENABLE register must be cleared.

AES DMA Mode

When AES DMA mode is enabled, the AES_IRQENABLE register must be cleared. To enable the µDMA to transfer data, follow these steps:

1. When the AES module has been initialized, enable the AES module µDMA channels by programming the DMA Channel Map Select n (DMA_CHMAPn) register in the µDMA module. Refer to Table 42-1 for the channel map associated with AES.
2. Configure the dma_done interrupts by programming the AES DMA Masked Interrupt Status (DTHE_AES_MIS) register.
3. Enable the µDMA channels in the AES by programming the µDMA enable bits in the AES System Configuration (AES_SYSCONFIG) register.

The input buffer registers, AES_DATA_IN_OUT_n, are now loaded.

42.4.1.4 AES Events Servicing

Interrupt Servicing

This section describes the event servicing of the module. Figure 42-13 shows the AES interrupt service. The registers used during event servicing follow:

- AES_IRQSTATUS
- AES_KEY1_n
- AES_KEY2_n
- AES_IV_IN_n
- AES_DATA_IN_OUT_n
- AES_TAG_OUT_n
- AES_IRQENABLE

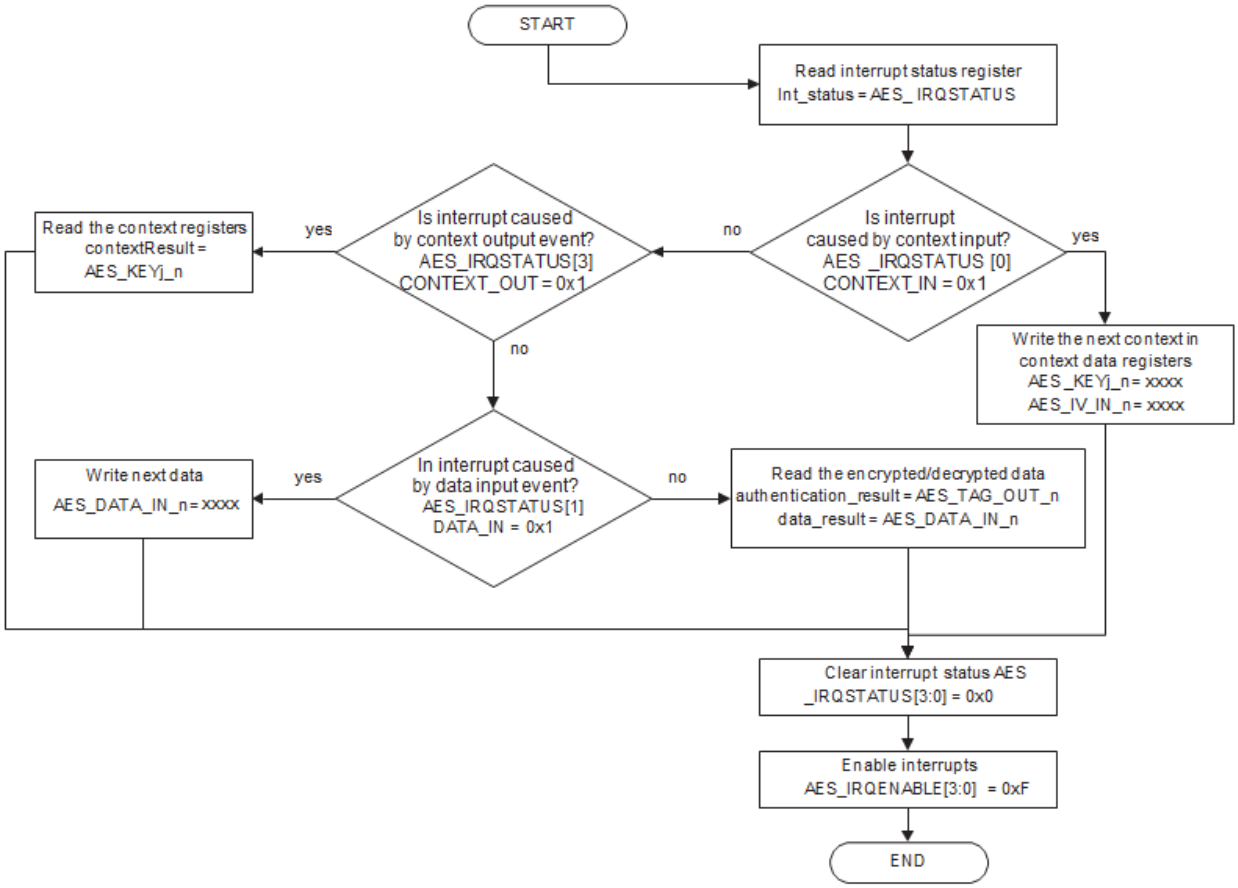


Figure 42-13. AES Interrupt Service

42.5 Software

42.5.1 AES Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/aes

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

42.5.1.1 AES ECB Encryption Example (CM) - CM

FILE: aes_ex1_ecb_encrypt.c

This example encrypts block cipher-text using AES128 in ECB mode. It does the encryption first without uDMA and then with uDMA. The results are checked after each operation.

External Connections
- None

Watch Variables
- *errCountGlobal* - Error Counter. It should be zero.
- *testStatusGlobal* - Test status. It should be equal to PASS.

42.5.1.2 AES ECB De-cryption Example (CM) - CM

FILE: aes_ex2_ecb_decrypt.c

This example de-crypts block cipher-text using AES128 in ECB mode. It does the de-cryption first without uDMA and then with uDMA. The results are checked after each operation.

External Connections
- None

Watch Variables
- *errCountGlobal* - Error Counter. It should be zero.
- *testStatusGlobal* - Test status. It should be equal to PASS.

42.5.1.3 AES GCM Encryption Example (CM) - CM

FILE: aes_ex3_gcm_encrypt.c

This example encrypts block cipher-text using AES128 in GCM mode. It does the encryption first without uDMA and then with uDMA. The results are checked after each operation.

External Connections
- None

Watch Variables
- *errorCountGlobal* - Error Counter. It should be zero.
- *testStatusGlobal* - Test status. It should be equal to PASS.

42.5.1.4 AES GCM Decryption Example (CM) - CM

FILE: aes_ex4_gcm_decrypt.c

This example decrypts block cipher-text using AES128 in GCM mode. It does the decryption first without uDMA and then with uDMA. The results are checked after each operation.

External Connections
- None

Watch Variables
- *errorCountGlobal* - Error Counter. It should be zero.
- *testStatusGlobal* - Test status. It should be equal to PASS.

42.6 AES Registers

The following sections are register descriptions for the AES Subsystem and AES Peripheral Core.

42.6.1 AES Base Addresses

Table 42-4. AES Base Address Table (CM)

DriverLib Name	Base Address	uDMA Access	Ethernet DMA Access
AES_SS_BASE	0x4004_AC00	YES	-
AES_BASE	0x4004_A000	YES	-

42.6.2 AES_SS_REGS Registers

Table 42-5 lists the memory-mapped registers for the AES_SS_REGS registers. All register offset addresses not listed in Table 42-5 should be considered as reserved locations and the register contents should not be modified.

Table 42-5. AES_SS_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	AESDMAINTEN	DMA Done Interrupt enable register		Go
4h	AESDMASTATUS	DMA Done Interrupt status register		Go
8h	AESDMASTATUSCLR	DMA Done Interrupt status clear register		Go

Complex bit access types are encoded to fit into small table cells. Table 42-6 shows the codes that are used for access types in this section.

Table 42-6. AES_SS_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

42.6.2.1 AESDMAINTEN Register (Offset = 0h) [Reset = 0h]

AESDMAINTEN is shown in Figure 42-14 and described in Table 42-7.

Return to the Summary Table.

DMA Done Interrupt enable register

Figure 42-14. AESDMAINTEN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				DMADONECTXOUT	DMADONEDOUT	DMADONEDIN	DMADONECTXIN
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 42-7. AESDMAINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	DMADONECTXOUT	R/W	0h	0: DMADONECTXOUT from uDMA will not be passed on as an interrupt 1:DMADONECTXOUT from uDMA will be passed on as an interrupt. Note: When this bit is 1, an interrupt will be generated on the rising edge of the corresponding DMA done signal Reset type: PER.RESET
2	DMADONEDOUT	R/W	0h	0: DMADONEDOUT from uDMA will not be passed on as an interrupt 1:DMADONEDOUT from uDMA will be passed on as an interrupt. Note: When this bit is 1, an interrupt will be generated on the rising edge of the corresponding DMA done signal Reset type: PER.RESET
1	DMADONEDIN	R/W	0h	0: DMADONEDIN from uDMA will not be passed on as an interrupt 1:DMADONEDIN from uDMA will be passed on as an interrupt. Note: When this bit is 1, an interrupt will be generated on the rising edge of the corresponding DMA done signal Reset type: PER.RESET
0	DMADONECTXIN	R/W	0h	0: DMADONECTXIN from uDMA will not be passed on as an interrupt 1:DMADONECTXIN from uDMA will be passed on as an interrupt. Note: When this bit is 1, an interrupt will be generated on the rising edge of the corresponding DMA done signal Reset type: PER.RESET

42.6.2.2 AESDMASTATUS Register (Offset = 4h) [Reset = 0h]

AESDMASTATUS is shown in Figure 42-15 and described in Table 42-8.

Return to the Summary Table.

DMA Done Interrupt status register

Figure 42-15. AESDMASTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				DMADONECTXOUT	DMADONEDOUT	DMADONEDIN	DMADONECTXIN
R-0h				R-0h	R-0h	R-0h	R-0h

Table 42-8. AESDMASTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	DMADONECTXOUT	R	0h	1: Indicates DMADONECTXOUT from uDMA has occurred. 0: IndicatesDMADONECTXOUT from uDMA has not occured. Reset type: PER.RESET
2	DMADONEDOUT	R	0h	1: Indicates DMADONEDOUT from uDMA has occurred. 0: IndicatesDMADONEDOUT from uDMA has not occured. Reset type: PER.RESET
1	DMADONEDIN	R	0h	1: Indicates DMADONEDIN from uDMA has occurred. 0: IndicatesDMADONEDIN from uDMA has not occured. Reset type: PER.RESET
0	DMADONECTXIN	R	0h	1: Indicates DMADONECTXIN from uDMA has occurred. 0: IndicatesDMADONECTXIN from uDMA has not occured. Reset type: PER.RESET

42.6.2.3 AESDMASTATUSCLR Register (Offset = 8h) [Reset = 0h]

AESDMASTATUSCLR is shown in Figure 42-16 and described in Table 42-9.

Return to the Summary Table.

DMA Done Interrupt status clear register

Figure 42-16. AESDMASTATUSCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				DMADONECTXOUT	DMADONEDOUT	DMADONEDIN	DMADONECTXIN
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 42-9. AESDMASTATUSCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	DMADONECTXOUT	R-0/W1S	0h	1: Clears DMADONECTXOUT bit of AESDMASTATUS. 0: No effect Reset type: PER.RESET
2	DMADONEDOUT	R-0/W1S	0h	1: Clears DMADONEDOUT bit of AESDMASTATUS. 0: No effect Reset type: PER.RESET
1	DMADONEDIN	R-0/W1S	0h	1: Clears DMADONEDIN bit of AESDMASTATUS. 0: No effect Reset type: PER.RESET
0	DMADONECTXIN	R-0/W1S	0h	1: Clears DMADONECTXIN bit of AESDMASTATUS. 0: No effect Reset type: PER.RESET

42.6.3 AES_REGS Registers

Table 42-10 lists the memory-mapped registers for the AES_REGS registers. All register offset addresses not listed in Table 42-10 should be considered as reserved locations and the register contents should not be modified.

Table 42-10. AES_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	AES_KEY2_6	XTS Second Key or CBC-MAC Third Key		Go
4h	AES_KEY2_7	XTS Second Key or CBC-MAC Third Key		Go
8h	AES_KEY2_4	XTS/CCM Second Key or CBC-MAC Third Key		Go
Ch	AES_KEY2_5	XTS Second Key or CBC-MAC Third Key		Go
10h	AES_KEY2_2	XTS/CCM/CBC-MAC Second Key or Hash Key Input		Go
14h	AES_KEY2_3	XTS/CCM/CBC-MAC Second Key or Hash Key Input		Go
18h	AES_KEY2_0	XTS/CCM/CBC-MAC Second Key or Hash Key Input		Go
1Ch	AES_KEY2_1	XTS/CCM/CBC-MAC Second Key or Hash Key Input		Go
20h	AES_KEY1_6	Key		Go
24h	AES_KEY1_7	Key		Go
28h	AES_KEY1_4	Key		Go
2Ch	AES_KEY1_5	Key		Go
30h	AES_KEY1_2	Key		Go
34h	AES_KEY1_3	Key		Go
38h	AES_KEY1_0	Key		Go
3Ch	AES_KEY1_1	Key		Go
40h	AES_IV_IN_OUT_0	Initialization Vector 0		Go
44h	AES_IV_IN_OUT_1	Initialization Vector 1		Go
48h	AES_IV_IN_OUT_2	Initialization Vector 2		Go
4Ch	AES_IV_IN_OUT_3	Initialization Vector 3		Go
50h	AES_CTRL	Input/Output Buffer Control and Mode Selection		Go
54h	AES_C_LENGTH_0	Crypto Data Length 0		Go
58h	AES_C_LENGTH_1	Crypto Data Length 1		Go
5Ch	AES_AUTH_LENGTH	AAD Data Length		Go
60h	AES_DATA_IN_OUT_0	Data Word 0		Go
64h	AES_DATA_IN_OUT_1	Data Word 1		Go
68h	AES_DATA_IN_OUT_2	Data Word 2		Go
6Ch	AES_DATA_IN_OUT_3	Data Word 3		Go
70h	AES_TAG_OUT_0	Hash Result 0		Go
74h	AES_TAG_OUT_1	Hash Result 1		Go
78h	AES_TAG_OUT_2	Hash Result 2		Go
7Ch	AES_TAG_OUT_3	Hash Result 3		Go
80h	AES_REV	Module Revision Number		Go
84h	AES_SYSCONFIG	System Configuration		Go
88h	AES_SYSSTATUS	Reset Status		Go
8Ch	AES_IRQSTATUS	Interrupt Status		Go
90h	AES_IRQENABLE	Interrupt Enable		Go

Table 42-10. AES_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
94h	AES_DIRTY_BITS	Accessed / Dirty Bits		Go

Complex bit access types are encoded to fit into small table cells. Table 42-11 shows the codes that are used for access types in this section.

Table 42-11. AES_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

42.6.3.1 AES_KEY2_6 Register (Offset = 0h) [Reset = 0h]

AES_KEY2_6 is shown in Figure 42-17 and described in Table 42-12.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-17. AES_KEY2_6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 42-12. AES_KEY2_6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	Key Data This register contains the 32-bit key data for the AES module. Initial key for XTS operations For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. The key size equals the AES_KEY1 size. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR Pre-calculated CBC-MAC third key (K3), used to perform a final XOR operation on the last input data block. OR This register is used to store intermediate values and must be initialized with zeroes when writing a new GCM context. OR Used in f8/f9 algorithm Reset type: PER.RESET

42.6.3.2 AES_KEY2_7 Register (Offset = 4h) [Reset = 0h]

AES_KEY2_7 is shown in Figure 42-18 and described in Table 42-13.

Return to the Summary Table.

Secure Host Interface Bank MSW for CBC-MAC and 256-bit XTS

Figure 42-18. AES_KEY2_7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 42-13. AES_KEY2_7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	Key Data This register contains the 32-bit key data for the AES module. Initial key for XTS operations For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. The key size equals the AES_KEY1 size. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR Pre-calculated CBC-MAC third key (K3), used to perform a final XOR operation on the last input data block. Reset type: PER.RESET

42.6.3.3 AES_KEY2_4 Register (Offset = 8h) [Reset = 0h]

AES_KEY2_4 is shown in Figure 42-19 and described in Table 42-14.

Return to the Summary Table.

Secure Host Interface Bank LSW for CBC-MAC

Figure 42-19. AES_KEY2_4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 42-14. AES_KEY2_4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	Key Data This register contains the 32-bit key data for the AES module. Initial key for XTS operations For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. The key size equals the AES_KEY1 size. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR Pre-calculated CBC-MAC third key (K3), used to perform a final XOR operation on the last input data block. Reset type: PER.RESET

42.6.3.4 AES_KEY2_5 Register (Offset = Ch) [Reset = 0h]

AES_KEY2_5 is shown in Figure 42-20 and described in Table 42-15.

Return to the Summary Table.

Secure Host Interface Bank MSW for 192-bit XTS

Figure 42-20. AES_KEY2_5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 42-15. AES_KEY2_5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	Key Data This register contains the 32-bit key data for the AES module. Initial key for XTS operations For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. The key size equals the AES_KEY1 size. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR Pre-calculated CBC-MAC third key (K3), used to perform a final XOR operation on the last input data block. Reset type: PER.RESET

42.6.3.5 AES_KEY2_2 Register (Offset = 10h) [Reset = 0h]

AES_KEY2_2 is shown in Figure 42-21 and described in Table 42-16.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-21. AES_KEY2_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 42-16. AES_KEY2_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	Key Data This register contains the 32-bit key data for the AES module. Initial key for XTS operations For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. The key size equals the AES_KEY1 size. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR Pre-calculated CBC-MAC second key (K2), used to perform a final XOR operation on the last input data block. OR Hash key, can be calculated internal or written via these registers. Only used for GHASH (GCM) modes. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR These 128-bits are used to store the CKKM / IKKM value for f8 OR These 128-bits are used to store the CKKM / IKKM value for f9 Reset type: PER.RESET

42.6.3.6 AES_KEY2_3 Register (Offset = 14h) [Reset = 0h]

AES_KEY2_3 is shown in Figure 42-22 and described in Table 42-17.

Return to the Summary Table.

Secure Host Interface Bank MSW for CCM, CBC-MAC, Hash Key, and 128-bit XTS

Figure 42-22. AES_KEY2_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 42-17. AES_KEY2_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	Key Data This register contains the 32-bit key data for the AES module. Initial key for XTS operations For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. The key size equals the AES_KEY1 size. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR Pre-calculated CBC-MAC second key (K2), used to perform a final XOR operation on the last input data block. OR Hash key, can be calculated internal or written via these registers. Only used for GHASH (GCM) modes. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR These 128-bits are used to store the CKKM / IKKM value for f8 OR These 128-bits are used to store the CKKM / IKKM value for f9 Reset type: PER.RESET

42.6.3.7 AES_KEY2_0 Register (Offset = 18h) [Reset = 0h]

AES_KEY2_0 is shown in Figure 42-23 and described in Table 42-18.

Return to the Summary Table.

Secure Host Interface Bank LSW for XTS, CCM, CBC-MAC, and Hash Key

Figure 42-23. AES_KEY2_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 42-18. AES_KEY2_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	Key Data This register contains the 32-bit key data for the AES module. Initial key for XTS operations For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. The key size equals the AES_KEY1 size. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR Pre-calculated CBC-MAC second key (K2), used to perform a final XOR operation on the last input data block. OR Hash key, can be calculated internal or written via these registers. Only used for GHASH (GCM) modes. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR These 128-bits are used to store the CKKM / IKKM value for f8 OR These 128-bits are used to store the CKKM / IKKM value for f9 Reset type: PER.RESET

42.6.3.8 AES_KEY2_1 Register (Offset = 1Ch) [Reset = 0h]

AES_KEY2_1 is shown in Figure 42-24 and described in Table 42-19.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-24. AES_KEY2_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 42-19. AES_KEY2_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	Key Data This register contains the 32-bit key data for the AES module. Initial key for XTS operations For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. The key size equals the AES_KEY1 size. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR Pre-calculated CBC-MAC second key (K2), used to perform a final XOR operation on the last input data block. OR Hash key, can be calculated internal or written via these registers. Only used for GHASH (GCM) modes. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR These 128-bits are used to store the CKKM / IKKM value for f8 OR These 128-bits are used to store the CKKM / IKKM value for f9 Reset type: PER.RESET

42.6.3.9 AES_KEY1_6 Register (Offset = 20h) [Reset = 0h]

AES_KEY1_6 is shown in Figure 42-25 and described in Table 42-20.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-25. AES_KEY1_6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 42-20. AES_KEY1_6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	AES Key register. For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. For a Host read operation, these registers return all-zeroes. The Host will typically write these registers in order, beginning with the AES_KEY_0 register. The key size (see the AES_MODE register) determines which key registers need to be populated, as indicated in the table below. Reset type: PER.RESET

42.6.3.10 AES_KEY1_7 Register (Offset = 24h) [Reset = 0h]

AES_KEY1_7 is shown in Figure 42-26 and described in Table 42-21.

Return to the Summary Table.

Secure Host Interface Bank MSW for 256-bit key

Figure 42-26. AES_KEY1_7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 42-21. AES_KEY1_7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	AES Key register. For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. For a Host read operation, these registers return all-zeroes. The Host will typically write these registers in order, beginning with the AES_KEY_0 register. The key size (see the AES_MODE register) determines which key registers need to be populated, as indicated in the table below. Reset type: PER.RESET

42.6.3.11 AES_KEY1_4 Register (Offset = 28h) [Reset = 0h]

AES_KEY1_4 is shown in Figure 42-27 and described in Table 42-22.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-27. AES_KEY1_4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 42-22. AES_KEY1_4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	AES Key register. For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. For a Host read operation, these registers return all-zeroes. The Host will typically write these registers in order, beginning with the AES_KEY_0 register. The key size (see the AES_MODE register) determines which key registers need to be populated, as indicated in the table below. Reset type: PER.RESET

42.6.3.12 AES_KEY1_5 Register (Offset = 2Ch) [Reset = 0h]

AES_KEY1_5 is shown in Figure 42-28 and described in Table 42-23.

Return to the Summary Table.

Secure Host Interface Bank MSW for 192-bit key

Figure 42-28. AES_KEY1_5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 42-23. AES_KEY1_5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	AES Key register. For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. For a Host read operation, these registers return all-zeroes. The Host will typically write these registers in order, beginning with the AES_KEY_0 register. The key size (see the AES_MODE register) determines which key registers need to be populated, as indicated in the table below. Reset type: PER.RESET

42.6.3.13 AES_KEY1_2 Register (Offset = 30h) [Reset = 0h]

AES_KEY1_2 is shown in Figure 42-29 and described in Table 42-24.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-29. AES_KEY1_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 42-24. AES_KEY1_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	AES Key register. For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. For a Host read operation, these registers return all-zeroes. The Host will typically write these registers in order, beginning with the AES_KEY_0 register. The key size (see the AES_MODE register) determines which key registers need to be populated, as indicated in the table below. Reset type: PER.RESET

42.6.3.14 AES_KEY1_3 Register (Offset = 34h) [Reset = 0h]

AES_KEY1_3 is shown in Figure 42-30 and described in Table 42-25.

Return to the Summary Table.

Secure Host Interface Bank MSW for 128-bit key

Figure 42-30. AES_KEY1_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 42-25. AES_KEY1_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	AES Key register. For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. For a Host read operation, these registers return all-zeroes. The Host will typically write these registers in order, beginning with the AES_KEY_0 register. The key size (see the AES_MODE register) determines which key registers need to be populated, as indicated in the table below. Reset type: PER.RESET

42.6.3.15 AES_KEY1_0 Register (Offset = 38h) [Reset = 0h]

AES_KEY1_0 is shown in Figure 42-31 and described in Table 42-26.

Return to the Summary Table.

Secure Host Interface Bank LSW

Figure 42-31. AES_KEY1_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 42-26. AES_KEY1_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	AES Key register. For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. For a Host read operation, these registers return all-zeroes. The Host will typically write these registers in order, beginning with the AES_KEY_0 register. The key size (see the AES_MODE register) determines which key registers need to be populated, as indicated in the table below. Reset type: PER.RESET

42.6.3.16 AES_KEY1_1 Register (Offset = 3Ch) [Reset = 0h]

AES_KEY1_1 is shown in Figure 42-32 and described in Table 42-27.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-32. AES_KEY1_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 42-27. AES_KEY1_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	AES Key register. For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. For a Host read operation, these registers return all-zeroes. The Host will typically write these registers in order, beginning with the AES_KEY_0 register. The key size (see the AES_MODE register) determines which key registers need to be populated, as indicated in the table below. Reset type: PER.RESET

42.6.3.17 AES_IV_IN_OUT_0 Register (Offset = 40h) [Reset = 0h]

AES_IV_IN_OUT_0 is shown in Figure 42-33 and described in Table 42-28.

Return to the Summary Table.

Secure Host Interface Bank LSW

Figure 42-33. AES_IV_IN_OUT_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 42-28. AES_IV_IN_OUT_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	For a Host write operation, these registers must be written with the new 128-bit IV to be subsequently transferred to the AES Engine. For a Host read operation, these registers contain the latest 128-bit IV output by the AES Engine. This value is incremented with 0x1, after first use when a new data block is submitted to the engine if CTR/ICM mode is selected. OR For XTS mode, this register must be written with the tweak location ("i") of the data unit or (intermediate) tweak value (Tj), For a Host read operation, the IV_OUT register holds the next tweak value that is used for the next data block provided via the DATA_IN registers. This value can be re-used for continuation with the next block. It must be provide together with a "j" of "0" via the IV_IN registers. OR For f8 this field must be written with zeroes. OR (In case of GCM) For a Host write operation, these registers must be written with the new 128-bit IV to be subsequently transferred to the AES Engine. For a Host read operation, these registers contain the latest 128-bit IV output by the AES Engine. Note that bits [127:96] of the IV represent the initial counter value (which is "1" for GCM) and must therefore be initialized to 0x01000000. This value is incremented with 0x1, after first use when a new data block is submitted to the engine. OR For CCM this field must be written with A0, this value constist of the concatenation of: A0-flags (5-bits of zero and 3-bits "L"), Nonce and counter value. "L" must be a copy from the "L" value of the AES_CTRL register. This "L" indicates the width of the Nonce and counter. The loaded counter must be initialized to zero. The total width of A0 is 128-bit. Reset type: PER.RESET

42.6.3.18 AES_IV_IN_OUT_1 Register (Offset = 44h) [Reset = 0h]

AES_IV_IN_OUT_1 is shown in Figure 42-34 and described in Table 42-29.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-34. AES_IV_IN_OUT_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 42-29. AES_IV_IN_OUT_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	For a Host write operation, these registers must be written with the new 128-bit IV to be subsequently transferred to the AES Engine. For a Host read operation, these registers contain the latest 128-bit IV output by the AES Engine. This value is incremented with 0x1, after first use when a new data block is submitted to the engine if CTR/ICM mode is selected. OR For XTS mode, this register must be written with the tweak location ("i") of the data unit or (intermediate) tweak value (Tj), For a Host read operation, the IV_OUT register holds the next tweak value that is used for the next data block provided via the DATA_IN registers. This value can be re-used for continuation with the next block. It must be provide together with a "j" of "0" via the IV_IN registers. OR For f8 this field must be written with zeroes. OR (In case of GCM) For a Host write operation, these registers must be written with the new 128-bit IV to be subsequently transferred to the AES Engine. For a Host read operation, these registers contain the latest 128-bit IV output by the AES Engine. Note that bits [127:96] of the IV represent the initial counter value (which is "1" for GCM) and must therefore be initialized to 0x01000000. This value is incremented with 0x1, after first use when a new data block is submitted to the engine. OR For CCM this field must be written with A0, this value constist of the concatenation of: A0-flags (5-bits of zero and 3-bits "L"), Nonce and counter value. "L" must be a copy from the "L" value of the AES_CTRL register. This "L" indicates the width of the Nonce and counter. The loaded counter must be initialized to zero. The total width of A0 is 128-bit. Reset type: PER.RESET

42.6.3.19 AES_IV_IN_OUT_2 Register (Offset = 48h) [Reset = 0h]

AES_IV_IN_OUT_2 is shown in Figure 42-35 and described in Table 42-30.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-35. AES_IV_IN_OUT_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 42-30. AES_IV_IN_OUT_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	For a Host write operation, these registers must be written with the new 128-bit IV to be subsequently transferred to the AES Engine. For a Host read operation, these registers contain the latest 128-bit IV output by the AES Engine. This value is incremented with 0x1, after first use when a new data block is submitted to the engine if CTR/ICM mode is selected. OR For XTS mode, this register must be written with the tweak location ("i") of the data unit or (intermediate) tweak value (Tj), For a Host read operation, the IV_OUT register holds the next tweak value that is used for the next data block provided via the DATA_IN registers. This value can be re-used for continuation with the next block. It must be provide together with a "j" of "0" via the IV_IN registers. OR For f8 this field must be written with zeroes. OR (In case of GCM) For a Host write operation, these registers must be written with the new 128-bit IV to be subsequently transferred to the AES Engine. For a Host read operation, these registers contain the latest 128-bit IV output by the AES Engine. Note that bits [127:96] of the IV represent the initial counter value (which is "1" for GCM) and must therefore be initialized to 0x01000000. This value is incremented with 0x1, after first use when a new data block is submitted to the engine. OR For CCM this field must be written with A0, this value constist of the concatenation of: A0-flags (5-bits of zero and 3-bits "L"), Nonce and counter value. "L" must be a copy from the "L" value of the AES_CTRL register. This "L" indicates the width of the Nonce and counter. The loaded counter must be initialized to zero. The total width of A0 is 128-bit. Reset type: PER.RESET

42.6.3.20 AES_IV_IN_OUT_3 Register (Offset = 4Ch) [Reset = 0h]

AES_IV_IN_OUT_3 is shown in Figure 42-36 and described in Table 42-31.

Return to the Summary Table.

Secure Host Interface Bank MSW

Figure 42-36. AES_IV_IN_OUT_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 42-31. AES_IV_IN_OUT_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	For a Host write operation, these registers must be written with the new 128-bit IV to be subsequently transferred to the AES Engine. For a Host read operation, these registers contain the latest 128-bit IV output by the AES Engine. This value is incremented with 0x1, after first use when a new data block is submitted to the engine if CTR/ICM mode is selected. OR For XTS mode, this register must be written with the tweak location ("i") of the data unit or (intermediate) tweak value (Tj), For a Host read operation, the IV_OUT register holds the next tweak value that is used for the next data block provided via the DATA_IN registers. This value can be re-used for continuation with the next block. It must be provide together with a "j" of "0" via the IV_IN registers. OR For f8 this field must be written with zeroes. OR (In case of GCM) For a Host write operation, these registers must be written with the new 128-bit IV to be subsequently transferred to the AES Engine. For a Host read operation, these registers contain the latest 128-bit IV output by the AES Engine. Note that bits [127:96] of the IV represent the initial counter value (which is "1" for GCM) and must therefore be initialized to 0x01000000. This value is incremented with 0x1, after first use when a new data block is submitted to the engine. OR For CCM this field must be written with A0, this value constist of the concatenation of: A0-flags (5-bits of zero and 3-bits "L"), Nonce and counter value. "L" must be a copy from the "L" value of the AES_CTRL register. This "L" indicates the width of the Nonce and counter. The loaded counter must be initialized to zero. The total width of A0 is 128-bit. Reset type: PER.RESET

42.6.3.21 AES_CTRL Register (Offset = 50h) [Reset = 80000000h]

AES_CTRL is shown in Figure 42-37 and described in Table 42-32.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-37. AES_CTRL Register

31	30	29	28	27	26	25	24
CTXTRDY	SVCTXTRDY	SAVE_CONTEXT	RESERVED				CCM_M
R-1h	R-0h	R/W-0h	R-0h				R/W-0h
23	22	21	20	19	18	17	16
CCM_M			CCM_L		CCM	GCM	
R/W-0h			R/W-0h		R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
CBCMAC	F9	F8	XTS		CFB	ICM	CTR_WIDTH
R/W-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CTR_WIDTH	CTR	MODE	KEY_SIZE		DIRECTION	INPUT_READY	OUTPUT_READY
R/W-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h	R-0h	R-0h

Table 42-32. AES_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CTXTRDY	R	1h	Context Data Registers Ready Value Description 0 The context data registers are not ready to be overwritten. 1 The context data registers can be overwritten and the host is permitted to write the next context. Reset type: PER.RESET
30	SVCTXTRDY	R	0h	AES TAG/IV Block(s) Ready This bit is only asserted if the SAVE_CONTEXT bit is set to 1. This bit is mutual exclusive with the CTXTRDY bit. Value Description 0 AES authentication TAG and/or IV block(s) is/are not available. 1 Indicates the AES authentication TAG and /or IV block(s) is/are available for the host to retrieve. Reset type: PER.RESET
29	SAVE_CONTEXT	R/W	0h	TAG or Result IV Save If this bit is set, the CONTEXT_OUT interrupt bit is set in the AES_IRQSTATUS register if the operation is finished and related signals are enabled. Value Description 0 No effect. 1 Indicates an authentication TAG of result IV needs to be stored as a result context. Reset type: PER.RESET
28-25	RESERVED	R	0h	Reserved

Table 42-32. AES_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24-22	CCM_M	R/W	0h	Counter with CBC-MAC (CCM)Defines M which indicates the length of the authentication field for CCM operations the authentication field length equals two times the sum of CCM-M plus one. The AES Engine always returns a 128-bit authentication field, of which the M least significant bytes are valid. All values are supported. Reset type: PER.RESET
21-19	CCM_L	R/W	0h	Indicates the width of the length field for CCM operations the length field in bytes equals the value of CMM-L plus one. Supported values for L are: Value Description 0x0 width = 0 0x1 width = 2 0x2 reserved 0x3 width = 4 0x4 - 0x6 reserved 0x7 width = 8 Reset type: PER.RESET
18	CCM	R/W	0h	AES-CCM Mode Enable Value Description 0 AES-CCM mode is not enabled. 1 AES-CCM mode enabled. This is a combined mode, using AES for both authentication and encryption. No additional mode selection is required. Reset type: PER.RESET
17-16	GCM	R/W	0h	AES-GCM Mode Enable This is a combined mode, using the Galois field-multiplier GF(2^{128}) for authentication and AES-CTR mode for encryption the bits specify the GCM mode. Value Description 0x0 No operation 0x1 GHASH with H loaded and Y0-encrypted forced to zero 0x2 GHASH with H loaded and Y0-encrypted calculated internally 0x3 Autonomous GHASH (both H and Y0-encrypted calculated internally) Reset type: PER.RESET
15	CBCMAC	R/W	0h	AES-CBC MAC Enable The DIRECTION bit must be set to 1 for this mode. Value Description 0 AES-CBC MAC mode is not enabled. 1 AES-CBC MAC mode enabled. Reset type: PER.RESET
14	F9	R/W	0h	AES f9 Mode Enable The AES key size must be set to 128-bit for this mode. Value Description 0 f9 mode is not enabled 1 f9 mode is enabled. Reset type: PER.RESET
13	F8	R/W	0h	AES f8 Mode Enable, The KEY_SIZE must be set to 128-bit for this mode. Value Description 0 AES f8 mode is not enabled. 1 AES f8 mode is enabled. Reset type: PER.RESET

Table 42-32. AES_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12-11	XTS	R/W	0h	AES-XTS Operation Enable The bits specify the XTS mode. Value Description 0x0 No operation 0x1 Previous/intermediate tweak value and j loaded (value is loaded via IV, j is loaded via the AAD length register) 0x2 Key2, n and j are loaded (n is loaded via IV, j is loaded via the AAD length register) 0x3 Key2 and n are loaded j=0 (n is loaded via IV) Reset type: PER.RESET
10	CFB	R/W	0h	Full block AES cipher feedback mode (CFB128) Enable Value Description 0 AES-CFB mode is not enabled. 1 AES-CFB mode is enabled. Reset type: PER.RESET
9	ICM	R/W	0h	AES Integer Counter Mode (ICM) Enable This is a counter mode with a 16-bit wide counter. Value Description 0 AES-ICM mode is not enabled. 1 AES-ICM mode is enabled. Reset type: PER.RESET
8-7	CTR_WIDTH	R/W	0h	AES-CTR Mode Counter Width Value Description 0x0 Counter is 32 bits 0x1 Counter is 64 bits 0x2 Counter is 96 bits 0x3 Counter is 128 bits Reset type: PER.RESET
6	CTR	R/W	0h	Counter Mode This bit must also be set for GCM and CCM mode, when encryption/decryption is required. Value Description 0 Counter mode is not enabled. 1 Counter mode is enabled. Reset type: PER.RESET
5	MODE	R/W	0h	ECB/CBC Mode Value Description 0 ECB mode 1 CBC mode Reset type: PER.RESET
4-3	KEY_SIZE	R/W	0h	Key Size Value Description 0x0 reserved 0x1 Key is 128 bits 0x2 Key is 192 bits 0x3 Key is 256 bits Reset type: PER.RESET
2	DIRECTION	R/W	0h	Encryption/Decryption Selection If set to =1, an encrypt operation is performed. If set to 0, a decrypt operation is performed. DIRECTION Value Description 0 Decryption is selected. 1 Encryption is selected. Reset type: PER.RESET

Table 42-32. AES_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	INPUT_READY	R	0h	Input Ready Status Value Description 0 Input buffer is not empty. 1 Indicates that the 16-byte input buffer is empty, and the host is permitted to write the next block of data. Reset type: PER.RESET
0	OUTPUT_READY	R	0h	Output Ready Status Value Description 0 No AES output block is available. 1 An AES output block is available for the host to retrieve. Reset type: PER.RESET

42.6.3.22 AES_C_LENGTH_0 Register (Offset = 54h) [Reset = 0h]

AES_C_LENGTH_0 is shown in Figure 42-38 and described in Table 42-33.

Return to the Summary Table.

Secure Host Interface Bank LSW

Figure 42-38. AES_C_LENGTH_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LENGTH																															
R-0/W-0h																															

Table 42-33. AES_C_LENGTH_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	LENGTH	R-0/W	0h	Bits [60:0] of the crypto length registers (LSW and MSW) store the cryptographic data length in bytes for all modes. Once processing with this context is started, this length decrements to zero. Data lengths up to (2^{61} - 1) bytes are allowed. For GCM, any value up to 2^{36} - 32 bytes can be used. This is because a 32-bit counter mode is used the maximum number of 128-bit blocks is 2^{32} - 2, resulting in a maximum number of bytes of 2^{36} - 32. A write to this register triggers the engine to start using this context. This is valid for all modes except GCM and CCM. Note that for the combined modes, this length does not include the authentication only data the authentication length is specified in the AES_AUTH_LENGTH register below. All modes must have a length > 0. For the combined modes, it is allowed to have one of the lengths equal to zero. For the basic encryption modes (ECB/CBC/CTR/ICM/CFB128) it is allowed to program zero to the length field in that case the length is assumed infinite. All data must be byte (8-bit) aligned for stream cipher modes bit aligned data streams are not supported by the AES Engine. For block cipher modes, the data length must be programmed in multiples of the block cipher size, 16 bytes. For a Host read operation, these registers return all-zeroes. Reset type: PER.RESET

42.6.3.23 AES_C_LENGTH_1 Register (Offset = 58h) [Reset = 0h]

AES_C_LENGTH_1 is shown in Figure 42-39 and described in Table 42-34.

Return to the Summary Table.

Secure Host Interface Bank MSW

Figure 42-39. AES_C_LENGTH_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LENGTH																															
R-0/W-0h																															

Table 42-34. AES_C_LENGTH_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	LENGTH	R-0/W	0h	Bits [60:0] of the crypto length registers (LSW and MSW) store the cryptographic data length in bytes for all modes. Once processing with this context is started, this length decrements to zero. Data lengths up to (2^{61} - 1) bytes are allowed. For GCM, any value up to 2^{36} - 32 bytes can be used. This is because a 32-bit counter mode is used the maximum number of 128-bit blocks is 2^{32} - 2, resulting in a maximum number of bytes of 2^{36} - 32. A write to this register triggers the engine to start using this context. This is valid for all modes except GCM and CCM. Note that for the combined modes, this length does not include the authentication only data the authentication length is specified in the AES_AUTH_LENGTH register below. All modes must have a length > 0. For the combined modes, it is allowed to have one of the lengths equal to zero. For the basic encryption modes (ECB/CBC/CTR/ICM/CFB128) it is allowed to program zero to the length field in that case the length is assumed infinite. All data must be byte (8-bit) aligned for stream cipher modes bit aligned data streams are not supported by the AES Engine. For block cipher modes, the data length must be programmed in multiples of the block cipher size, 16 bytes. For a Host read operation, these registers return all-zeroes. Reset type: PER.RESET

42.6.3.24 AES_AUTH_LENGTH Register (Offset = 5Ch) [Reset = 0h]

AES_AUTH_LENGTH is shown in Figure 42-40 and described in Table 42-35.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-40. AES_AUTH_LENGTH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AUTH																															
R-0/W-0h																															

Table 42-35. AES_AUTH_LENGTH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	AUTH	R-0/W	0h	Bits [31:0] of the authentication length register store the authentication data length in bytes for combined modes only (GCM or CCM) Supported AAD-lengths for CCM are from 0 to (2^{16} - 2^{8}) bytes. For GCM any value up to (2^{32} - 1) bytes can be used. Once processing with this context is started, this length decrements to zero. A write to this register triggers the engine to start using this context for GCM and CCM. For XTS this register is optionally used to load "j". Loading of "j" is only required if "j" != 0. "j" is a 28-bit value and must be written to bits [31-4] of this register. "j" represents the sequential number of the 128-bit block inside the data unit. For the first block in a unit, this value is zero. It is not required to provide a "j" for each new data block within a unit. Note that it is possible to start with a "j" unequal to zero refer to Table 4 for more details. For a Host read operation, these registers return all-zeroes. Reset type: PER.RESET

42.6.3.25 AES_DATA_IN_OUT_0 Register (Offset = 60h) [Reset = 0h]

AES_DATA_IN_OUT_0 is shown in Figure 42-41 and described in Table 42-36.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-41. AES_DATA_IN_OUT_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 42-36. AES_DATA_IN_OUT_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	The Data Input/Output Registers buffer the input/output data blocks to/from the AES Engine. Notice that the data input buffer (AES_DATA_IN_n) and data output buffer (AES_DATA_OUT_n) are mapped to the same address locations. Writes to these addresses load the Input Buffer while reads pull from the Output Buffer. Therefore, for write access, the data input buffer is written for read access, the data output buffer is read. The data input buffer must be written prior to starting an operation. The data output buffer contains valid data on completion of an operation. All writes from, and reads to, these registers are tracked independently per direction and HIB. Therefore, any 128-bit data block can be split over multiple 32-bit word transfers, which can be mixed with other transfers over the external interface. Reset type: PER.RESET

42.6.3.26 AES_DATA_IN_OUT_1 Register (Offset = 64h) [Reset = 0h]

AES_DATA_IN_OUT_1 is shown in Figure 42-42 and described in Table 42-37.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-42. AES_DATA_IN_OUT_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 42-37. AES_DATA_IN_OUT_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	The Data Input/Output Registers buffer the input/output data blocks to/from the AES Engine. Notice that the data input buffer (AES_DATA_IN_n) and data output buffer (AES_DATA_OUT_n) are mapped to the same address locations. Writes to these addresses load the Input Buffer while reads pull from the Output Buffer. Therefore, for write access, the data input buffer is written for read access, the data output buffer is read. The data input buffer must be written prior to starting an operation. The data output buffer contains valid data on completion of an operation. All writes from, and reads to, these registers are tracked independently per direction and HIB. Therefore, any 128-bit data block can be split over multiple 32-bit word transfers, which can be mixed with other transfers over the external interface. Reset type: PER.RESET

42.6.3.27 AES_DATA_IN_OUT_2 Register (Offset = 68h) [Reset = 0h]

AES_DATA_IN_OUT_2 is shown in Figure 42-43 and described in Table 42-38.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-43. AES_DATA_IN_OUT_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 42-38. AES_DATA_IN_OUT_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	The Data Input/Output Registers buffer the input/output data blocks to/from the AES Engine. Notice that the data input buffer (AES_DATA_IN_n) and data output buffer (AES_DATA_OUT_n) are mapped to the same address locations. Writes to these addresses load the Input Buffer while reads pull from the Output Buffer. Therefore, for write access, the data input buffer is written for read access, the data output buffer is read. The data input buffer must be written prior to starting an operation. The data output buffer contains valid data on completion of an operation. All writes from, and reads to, these registers are tracked independently per direction and HIB. Therefore, any 128-bit data block can be split over multiple 32-bit word transfers, which can be mixed with other transfers over the external interface. Reset type: PER.RESET

42.6.3.28 AES_DATA_IN_OUT_3 Register (Offset = 6Ch) [Reset = 0h]

AES_DATA_IN_OUT_3 is shown in Figure 42-44 and described in Table 42-39.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-44. AES_DATA_IN_OUT_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 42-39. AES_DATA_IN_OUT_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	The Data Input/Output Registers buffer the input/output data blocks to/from the AES Engine. Notice that the data input buffer (AES_DATA_IN_n) and data output buffer (AES_DATA_OUT_n) are mapped to the same address locations. Writes to these addresses load the Input Buffer while reads pull from the Output Buffer. Therefore, for write access, the data input buffer is written for read access, the data output buffer is read. The data input buffer must be written prior to starting an operation. The data output buffer contains valid data on completion of an operation. All writes from, and reads to, these registers are tracked independently per direction and HIB. Therefore, any 128-bit data block can be split over multiple 32-bit word transfers, which can be mixed with other transfers over the external interface. Reset type: PER.RESET

42.6.3.29 AES_TAG_OUT_0 Register (Offset = 70h) [Reset = 0h]

AES_TAG_OUT_0 is shown in Figure 42-45 and described in Table 42-40.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-45. AES_TAG_OUT_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HASH																															
R-0h																															

Table 42-40. AES_TAG_OUT_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HASH	R	0h	Bits [31:0] of the AES TAG registers store the authentication value for the combined and authentication only modes. For a Host read operation, these registers contain the last 128-bit TAG output of the AES Engine the TAG is available until the next context is written. This register will only contain valid data if the TAG is available, when the "store_ready" bit from AES_CTRL register is set. During processing or for operations/modes that do not return a TAG, reads from this register returns data from the IV register. For operations that do return a TAG in the IV register (e.g. XTS), the IV register must be accessed directly. Reset type: PER.RESET

42.6.3.30 AES_TAG_OUT_1 Register (Offset = 74h) [Reset = 0h]

AES_TAG_OUT_1 is shown in Figure 42-46 and described in Table 42-41.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-46. AES_TAG_OUT_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HASH																															
R-0h																															

Table 42-41. AES_TAG_OUT_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HASH	R	0h	Bits [31:0] of the AES TAG registers store the authentication value for the combined and authentication only modes. For a Host read operation, these registers contain the last 128-bit TAG output of the AES Engine the TAG is available until the next context is written. This register will only contain valid data if the TAG is available, when the "store_ready" bit from AES_CTRL register is set. During processing or for operations/modes that do not return a TAG, reads from this register returns data from the IV register. For operations that do return a TAG in the IV register (e.g. XTS), the IV register must be accessed directly. Reset type: PER.RESET

42.6.3.31 AES_TAG_OUT_2 Register (Offset = 78h) [Reset = 0h]

AES_TAG_OUT_2 is shown in Figure 42-47 and described in Table 42-42.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-47. AES_TAG_OUT_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HASH																															
R-0h																															

Table 42-42. AES_TAG_OUT_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HASH	R	0h	Bits [31:0] of the AES TAG registers store the authentication value for the combined and authentication only modes. For a Host read operation, these registers contain the last 128-bit TAG output of the AES Engine the TAG is available until the next context is written. This register will only contain valid data if the TAG is available, when the "store_ready" bit from AES_CTRL register is set. During processing or for operations/modes that do not return a TAG, reads from this register returns data from the IV register. For operations that do return a TAG in the IV register (e.g. XTS), the IV register must be accessed directly. Reset type: PER.RESET

42.6.3.32 AES_TAG_OUT_3 Register (Offset = 7Ch) [Reset = 0h]

AES_TAG_OUT_3 is shown in Figure 42-48 and described in Table 42-43.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-48. AES_TAG_OUT_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HASH																															
R-0h																															

Table 42-43. AES_TAG_OUT_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HASH	R	0h	Bits [31:0] of the AES TAG registers store the authentication value for the combined and authentication only modes. For a Host read operation, these registers contain the last 128-bit TAG output of the AES Engine the TAG is available until the next context is written. This register will only contain valid data if the TAG is available, when the "store_ready" bit from AES_CTRL register is set. During processing or for operations/modes that do not return a TAG, reads from this register returns data from the IV register. For operations that do return a TAG in the IV register (e.g. XTS), the IV register must be accessed directly. Reset type: PER.RESET

42.6.3.33 AES_REV Register (Offset = 80h) [Reset = 40000B02h]

AES_REV is shown in Figure 42-49 and described in Table 42-44.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-49. AES_REV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REVISION																															
R-40000B02h																															

Table 42-44. AES_REV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REVISION	R	40000B02h	Revision number: 31-30:SCHEME = 0b01 29-28:Reserved=0b00 27-16:FUNC = 0x000 15-11:RTL Revision = 0b00001 10-8:MAJOR = 0b011 7-2:CUSTOM = 0b000000 1-0:MINOR = 0b10 Reset type: PER.RESET

42.6.3.34 AES_SYSCONFIG Register (Offset = 84h) [Reset = 1h]

AES_SYSCONFIG is shown in Figure 42-50 and described in Table 42-45.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-50. AES_SYSCONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			RESERVED	RESERVED	RESERVED	MAP_CONTEXT_OUT_ON_DATA_OUT	DMA_REQ_CONTEXT_OUT_EN
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DMA_REQ_CONTEXT_IN_EN	DMA_REQ_DATA_OUT_EN	DMA_REQ_DATA_IN_EN	RESERVED	SIDLE		SOFTRESET	AUTOIDLE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-1h

Table 42-45. AES_SYSCONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	MAP_CONTEXT_OUT_ON_DATA_OUT	R/W	0h	If set to '1' the two context out requests (dma_req_context_out_en, Bit [8] above, and context_out interrupt enable, Bit [3] of AES_IRQENABLE register) are mapped on the corresponding data output request bit. In this case, the original "context out" bit values are ignored. Therefore, when this bit is enabled and the dma_req_data_out_en (Bit [6]) is enabled, this DMA request is used for the context output and data output. Similarly if the data_out interrupt enable (Bit [2] of AES_IRQENABLE register) is enabled, this interrupt is used for context output and data output. Note that when context and data output request or interrupt are mapped on each other, the context output is always requested after the last data output. Reset type: PER.RESET
8	DMA_REQ_CONTEXT_OUT_EN	R/W	0h	DMA Request Context Out Enable If set to 1, the DMA context output request is enabled (for context data out, for example, TAG for authentication modes). Value Description 0 DMA disabled for context output request. 1 DMA enabled for context output request. Reset type: PER.RESET
7	DMA_REQ_CONTEXT_IN_EN	R/W	0h	DMA Request Context In Enable Value Description 0 DMA disabled for context input request. 1 DMA enabled for context input request. Reset type: PER.RESET

Table 42-45. AES_SYSCONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	DMA_REQ_DATA_OUT_EN	R/W	0h	DMA Request Data Out Enable Value Description 0 DMA disabled for data output request. 1 DMA enabled for data output request. Reset type: PER.RESET
5	DMA_REQ_DATA_IN_EN	R/W	0h	DMA Request Data In Enable Value Description 0 DMA disabled for data input request. 1 DMA enabled for data input request. Reset type: PER.RESET
4	RESERVED	R/W	0h	Reserved
3-2	SIDLE	R/W	0h	Slave Idle Mode Value Description 0x0 Force-idle mode 0x1 No-idle 0x2 Smart-idle 0x3 reserved Reset type: PER.RESET
1	SOFTRESET	R/W	0h	Soft reset Value Description 0 No operation 1 Start soft reset sequence Reset type: PER.RESET
0	AUTOIDLE	R/W	1h	If set to 1, the internal clocks are switched off when there is no processing to be done. This bit is only available on the sHIB. Reset type: PER.RESET

42.6.3.35 AES_SYSSTATUS Register (Offset = 88h) [Reset = 1h]

AES_SYSSTATUS is shown in Figure 42-51 and described in Table 42-46.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-51. AES_SYSSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							RESETDONE
R-0h							R-1h

Table 42-46. AES_SYSSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	RESETDONE	R	1h	Reset Done Value Description 0 Reset is not complete. 1 Reset is has completed. Reset type: PER.RESET

42.6.3.36 AES_IRQSTATUS Register (Offset = 8Ch) [Reset = 0h]

AES_IRQSTATUS is shown in Figure 42-52 and described in Table 42-47.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-52. AES_IRQSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CONTEXT_OUT	DATA_OUT	DATA_IN	CONTEXT_IN
R-0h				R-0h	R-0h	R-0h	R-0h

Table 42-47. AES_IRQSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	CONTEXT_OUT	R	0h	Context Output Interrupt Status Value Description 0 Authentication tag (and IV) interrupt(s) is/are not active. 1 Authentication tag (and IV) interrupt(s) is/are active and the interrupt output has been triggered. Reset type: PER.RESET
2	DATA_OUT	R	0h	Data Out Interrupt Status Value Description 0 The data out interrupt is not active. 1 The data out interrupt is active and the interrupt output has been triggered. Reset type: PER.RESET
1	DATA_IN	R	0h	Data In Interrupt Status Value Description 0 The data in interrupt is not active. 1 The data in interrupt is active and the interrupt output has been triggered. Reset type: PER.RESET
0	CONTEXT_IN	R	0h	Context In Interrupt Status Value Description 0 The context in interrupt is not active. 1 The context in interrupt is active and the interrupt output has been triggered. Reset type: PER.RESET

42.6.3.37 AES_IRQENABLE Register (Offset = 90h) [Reset = 0h]

AES_IRQENABLE is shown in Figure 42-53 and described in Table 42-48.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-53. AES_IRQENABLE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CONTEXT_OUT	DATA_OUT	DATA_IN	CONTEXT_IN
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 42-48. AES_IRQENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	CONTEXT_OUT	R/W	0h	Context Out Interrupt Enable Value Description 0 Authentication tag (and IV) interrupt(s) is/are disabled. 1 Authentication tag (and IV) interrupt(s) is/are enabled. Reset type: PER.RESET
2	DATA_OUT	R/W	0h	Data Out Interrupt Enable Value Description 0 The data out interrupt is disabled. 1 The data out interrupt is enabled. Reset type: PER.RESET
1	DATA_IN	R/W	0h	Data In Interrupt Enable Value Description 0 The data in interrupt is disabled. 1 The data in interrupt is enabled. Reset type: PER.RESET
0	CONTEXT_IN	R/W	0h	Context In Interrupt Enable Value Description 0 The context in interrupt is disabled. 1 The context in interrupt is enabled. Reset type: PER.RESET

42.6.3.38 AES_DIRTY_BITS Register (Offset = 94h) [Reset = 0h]

AES_DIRTY_BITS is shown in Figure 42-54 and described in Table 42-49.

Return to the Summary Table.

Secure Host Interface Bank

Figure 42-54. AES_DIRTY_BITS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	S_DIRTY	S_ACCESS
R-0h				R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

Table 42-49. AES_DIRTY_BITS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W1S	0h	Reserved
2	RESERVED	R/W1S	0h	Reserved
1	S_DIRTY	R/W1S	0h	AES Dirty Bit This bit must be written to a 1 to clear. Value Description 0 No AES registers have been written. 1 Indicates when any of the AES_x registers have been written (except for the AES_DIRTYBITS register). Reset type: PER.RESET
0	S_ACCESS	R/W1S	0h	AES Access Bit This bit must be written to a 1 to clear. Value Description 0 No AES registers have been read. 1 Indicates when any of the AES_x registers have been read (except for the AES_DIRTYBITS register). Reset type: PER.RESET

This page intentionally left blank.

Chapter 43

Ethernet Media Access Controller (EMAC)

This chapter describes the Ethernet module and provides a functional description of the module.

43.1 Introduction

The Ethernet module enables a host to transmit and receive data over the Ethernet in compliance with IEEE 802.3-2015.

43.1.1 Standard Compliance

In addition to the default interfaces defined in the IEEE 802.3 specifications, the Ethernet module supports several industry standard interfaces to the PHY. The Ethernet module is compliant with the following standards:

The EBC unit has the following capabilities:

- IEEE 802.3-2015 for Ethernet MAC, Media Independent Interface (MII)
- IEEE 1588-2008 for precision networked clock synchronization
- IEEE 802.3az-2010 for Energy Efficient Ethernet (EEE)
- RMII specification version 1.2 from RMII consortium
- Reverse Media Independent Interface (RevMII) by Dmitriy Gusev

43.1.2 MAC Features

The Ethernet controller supports a number of Tx and Rx MAC features. The MAC includes the following feature groups:

- MAC Tx and Rx features
- MAC Tx features
- MAC Rx features

43.1.2.1 MAC Tx and Rx Features

The combined features for Tx and Rx are as follows:

- Separate transmission, reception, and control interfaces to the application
- Little-endian mode for Transmit and Receive paths
- 10, 100 data transfer rates with the following PHY interfaces:
  - IEEE 802.3-compliant MII (default) interface to communicate with an external Ethernet PHY
  - RMII interface to communicate with an external Fast Ethernet PHY
  - RevMII interface to directly communicate with a remote MAC
- Half-duplex operation:
  - CSMA/CD Protocol support
  - Flow control using back pressure support (based on implementation-specific white papers and UNH Ethernet Clause 4 MAC Test Suite - Annex D)
- Standard IEEE 802.3az-2010 for Energy Efficient Ethernet in MII PHYs.
- Full-duplex flow control operations (IEEE 802.3x Pause packets and Priority flow control)
- Network statistics with RMON or MIB Counters (RFC2819/RFC2665)
- Support Ethernet packet timestamping as described in IEEE 1588-2002 and IEEE 1588-2008 (64-bit timestamps given in the Tx or Rx status of PTP packet). Both one-step and two-step timestamping is supported in TX direction.
- Flexibility to control the Pulse-Per-Second (PPS) output signal
- MDIO (Clause 22 and Clause 45) master interface for PHY device configuration and management

43.1.2.2 MAC Tx Features

The MAC Tx features are as follows:

- Preamble and start of packet data (SFD) insertion
- Separate 32-bit status for each packet transmitted from the application
- Automatic CRC and pad generation controllable on a per-packet basis
- Programmable packet length to support Standard or Jumbo Ethernet packets with up to 16 KB of size
- Programmable Inter Packet Gap (40–96 bit times in steps of 8)
- IEEE 802.3x Flow Control automatic transmission of zero-quanta Pause packet when flow control input transitions from assertion to de-assertion (in full-duplex mode)
- Source Address field insertion or replacement, and VLAN insertion, replacement, and deletion in transmitted packets with per-packet or static-global control
- Insertion, replacement, or deletion of up to two VLAN tags
- Option to transmit packets with reduced preamble size in full-duplex mode
- Insert, replace, or delete queue/channel-based VLAN tags

43.1.2.3 MAC Rx Features

The MAC Rx features are as follows:

- Flexible address filtering modes:
  - 8 48-bit perfect Destination Address/Source address filters with masks for each byte
  - 64-bit Hash filter for multicast and unicast (DA) addresses
  - Option to pass all multi-cast addressed packets
  - Promiscuous mode to pass all packets without any filtering for network monitoring
  - Pass all incoming packets (as per filter) with a status report
- Additional packet filtering:
  - VLAN tag-based: Perfect match and Hash-based filtering. Filtering based on either outer or inner VLAN tag is possible.
  - Layer 3 and Layer 4-based: TCP or UDP over IPv4 or IPv6
  - Extended VLAN tag based filtering 4-filter selection
- IEEE 802.1Q VLAN tag detection and option to delete the VLAN tags in received packets
- Module to detect remote wake-up packets and AMD magic packets
- Forwarding of received Pause packets to the application (in full-duplex mode)
- Receive module for Layer 3/Layer 4 checksum offload for received packets
- Stripping of up to two VLAN Tags and providing the tags in the status.

43.2 System Level Integration

This section provides a listing and description of the ethernet signal connections.

43.2.1 Ethernet Signal Connection and Description

The different types of PHY interfaces supported by the Ethernet Controller and their corresponding signal connection and description is given below.

43.2.1.1 MII Interface Signals

The MII interface signals and descriptions are shown in Table 43-1 and Figure 43-1.

Table 43-1. MII Interface Signals

Signal	Type	Description
ENET_MII_TX_CLK	I	The transmit clock is a continuous clock that provides the timing reference for transmit operations. The ENET_MII_TX_DATA and ENET_MII_TXEN signals are connected to this clock. The clock is generated by the PHY and is 2.5 MHz at 10 Mbps operation and 25 MHz at 100 Mbps operation.
ENET_MII_TX_DATA[3:0]	O	The transmit data pins are a collection of 4 data signals, ENET_MII_TX_DATA[3:0], comprising 4 bits of data. ENET_MII_TX_DATA[0] is the least-significant bit. The signals are synchronized by ENET_MII_TX_CLK and valid only when ENET_MII_TX_EN is asserted.
ENET_MII_TX_EN	O	The transmit enable signal indicates that the ENET_MII_TXDATA[3:0] pins are generating 4-bit data for use by the PHY. This signal is driven synchronously by ENET_MII_TX_CLK.
ENET_MII_TX_ERR	O	The Transmit Error signal.
ENET_MII_COL	I	In full-duplex operation, the ENET_MII_COL pin is used for hardware transmit flow control. Asserting the ENET_MII_COL pin stops packet transmissions; packets transmitting when ENET_MII_COL is asserted completes transmission. The ENET_MII_COL pin should be held low, if hardware transmit flow control is not used.
ENET_MII_CRS	I	In half-duplex operation, the ENET_MII_CRS pin is asserted by the PHY when the network is not idle in either transmit or receive. The pin is deasserted when both transmit and receive are idle. This signal is not necessarily synchronous to ENET_MII_TX_CLK or ENET_MII_RX_CLK. In full-duplex operation, the ENET_MII_CRS pin should be held low.
ENET_MII_RX_CLK	I	The receive clock is a continuous clock that provides the timing reference for receive operations. The ENET_MII_RX_DATA and ENET_MII_RX_DV signals are connected to this clock. The clock is generated by the PHY and is 2.5 MHz at 10 Mbps operation and 25 MHz at 100 Mbps operation.
ENET_MII_RX_DATA[3:0]	I	The receive data pins are a collection of 4 data signals comprising 4 bits of data. ENET_MII_RX_DATA[0] is the least-significant bit. The signals are synchronized by ENET_MII_RX_CLK and valid only when ENET_MII_RX_DV is asserted.
ENET_MII_RX_DV	I	The receive data valid signal indicates that the ENET_MII_RX_DATA pins are generating nibble data for use by the MAC. This signal is driven synchronous to ENET_MII_RX_CLK.
ENET_MII_RX_ER	I	Receive Error. The ENET_MII_RX_ER signal is asserted to indicate that an error is detected in received frame.
ENET_MII_INTR	I	Interrupt input from the physical layer chip outside the device.
ENET_MDIO_CLK	O	Management data clock. The MDIO data clock is sourced by the MDIO module on the system. The clock is used to synchronize MDIO data access operations done on the ENET_MDIO_DATA pin.
ENET_MDIO_DATA	I/O	The ENET_MDIO_DATA pin drives PHY management data into and out of the PHY by way of an access frame consisting of start of frame, read/write indication, PHY address, register address, and data bit cycles. The ENET_MDIO_DATA pin acts as an output for all but the data bit cycles at which time the pin is an input for read operations

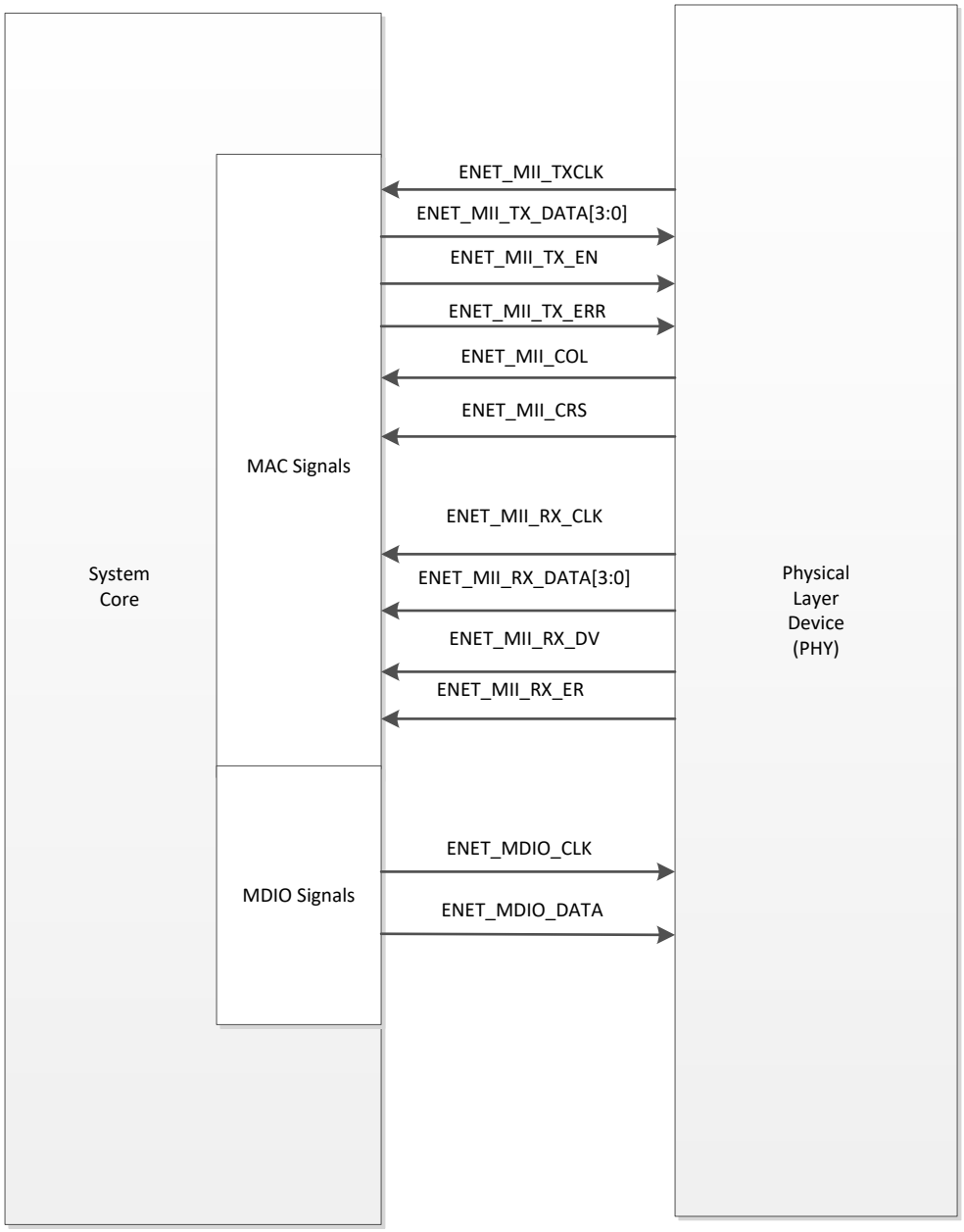


Figure 43-1. MII Mode Signals

43.2.1.2 RMII Interface Signals

The RMII interface signals and descriptions are shown in Table 43-2 and Figure 43-2.

Table 43-2. RMII Interface Signals

Signal	Type	Description
ENET_RMII_CLK	I/O	The reference clock is a continuous clock that provides the timing reference for transmit,receive operations. The ENET_MII_TX_DATA and ENET_MII_TX_EN signals are connected to this clock. The clock is generated by the PHY and is 5.0 MHz at 10 Mbps operation and 50 MHz at 100 Mbps operation. If External Clocking option is selected, this signal is input from the PHY/ external device; if internal clock is selected, this signal is output.

Table 43-2. RMII Interface Signals (continued)

Signal	Type	Description
ENET_MII_TX_DATA[1:0]	O	The transmit data pins are a collection of 2 data signals, ENET_MII_TX_DATA[1:0], comprising 2 bits of data. ENET_MII_TX_DATA[0] is the least-significant bit. The signals are synchronized by ENET_RMII_CLK and valid only when ENET_MII_TX_EN is asserted.
ENET_MII_TX_EN	O	The transmit enable signal indicates that the ENET_MII_TX_DATA[1:0] pins are generating 2-bit data for use by the PHY. This signal is driven synchronously by ENET_RMII_CLK.
ENET_MII_RX_DV	I	Multiplexed signal between Carrier Sense and Receive Data Valid.
ENET_MII_RX_DATA[1:0]	I	The receive data pins are a collection of 2 data signals comprising 2 bits of data. ENET_MII_RX_DATA[0] is the least-significant bit. The signals are synchronized by ENET_RMII_CLK and valid only when ENET_MII_RX_DV is asserted.
ENET_MII_RX_ERR	I	Receive Error. The ENET_MII_RX_ERR signal is asserted to indicate that an error is detected in received frame.
ENET_MDIO_CLK	O	Management data clock. The MDIO data clock is sourced by the MDIO module on the system. The clock is used to synchronize MDIO data access operations done on the ENET_MDIO_DATA pin.
ENET_MDIO_DATA	I/O	The ENET_MDIO_DATA pin drives PHY management data into and out of the PHY by way of an access frame consisting of start of frame, read/write indication, PHY address, register address, and data bit cycles. The ENET_MDIO_DATA pin acts as an output for all but the data bit cycles at which time the pin is an input for read operations.

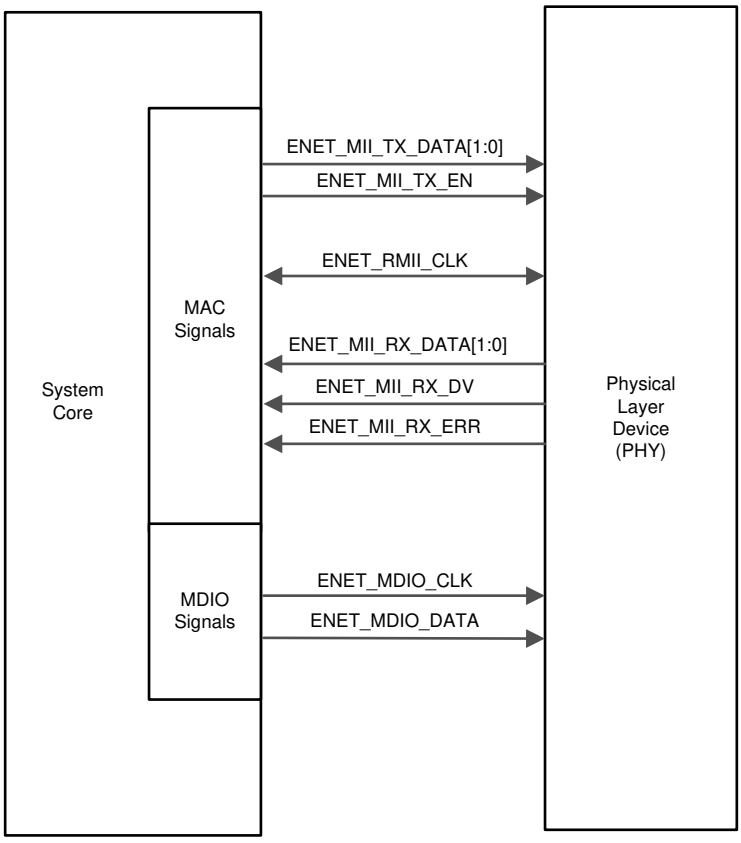


Figure 43-2. RMII Mode Signals

43.2.1.3 RevMII Interface Signals

The RevMII interface signals and descriptions are shown in Table 43-3 and Figure 43-3.

Table 43-3. RevMII Interface Signals

Signal	Type	Description
ENET_MII_TX_CLK	O	The transmit clock is a continuous clock that provides the timing reference for transmit operations. The ENET_MII_TX_DATA and ENET_MII_TX_EN signals are connected to this clock. The clock is generated from the device and is 2.5 MHz at 10 Mbps operation and 25 MHz at 100 Mbps operation.
ENET_MII_TX_DATA[3:0]	O	The transmit data pins are a collection of 4 data signals, ENET_MII_TX_DATA[3:0], comprising 4 bits of data. ENET_MII_TX_DATA[0] is the least-significant bit. The signals are synchronized by ENET_MII_TX_CLK and valid only when ENET_MII_TX_EN is asserted.
ENET_MII_TX_EN	O	The transmit enable signal indicates that the ENET_MII_TX_DATA[3:0] pins are generating 4-bit data for use by the PHY. This signal is driven synchronously by ENET_MII_TX_CLK.
ENET_MII_TX_ERR	O	EMAC MII transmit error.
ENET_MII_COL	O	In full-duplex operation, the ENET_MII_COL pin is used for hardware transmit flow control. Asserting the ENET_MII_COL pin stops packet transmissions; packets transmitting when ENET_MII_COL is asserted completes transmission. The ENET_MII_COL pin should be held low, if hardware transmit flow control is not used.
ENET_MII_CRS	O	In half-duplex operation, the ENET_MII_CRS pin is asserted by the PHY when the network is not idle in either transmit or receive. The pin is deasserted when both transmit and receive are idle. This signal is not necessarily synchronous to ENET_MII_TX_CLK or ENET_MII_RX_CLK. In full-duplex operation, the ENET_MII_CRS pin should be held low.
ENET_MII_RX_CLK	O	The receive clock is a continuous clock that provides the timing reference for receive operations. The ENET_MII_RX_DATA and ENET_MII_RX_DV signals are connected to this clock. The clock is generated from the device and is 2.5 MHz at 10 Mbps operation and 25 MHz at 100 Mbps operation.
ENET_MII_RX_DATA[3:0]	I	The receive data pins are a collection of 4 data signals comprising 4 bits of data. ENET_MII_RX_DATA[0] is the least-significant bit. The signals are synchronized by ENET_MII_RX_CLK and valid only when ENET_MII_RX_DV is asserted.
ENET_MII_RX_DV	I	The receive data valid signal indicates that the ENET_MII_RX_DATA pins are generating nibble data for use by the MAC. This signal is driven synchronous to ENET_MII_RX_CLK.
ENET_MII_RX_ERR	I	EMAC MII / RMII receive error.
ENET_MII_INTR	O	RevMII mode PHY interrupt out pin.
ENET_MDIO_CLK	I	The ENET_MDIO_DATA pin drives PHY management data into and out of the PHY by way of an access frame consisting of start of frame, read/write indication, PHY address, register address, and data bit cycles. The ENET_MDIO_DATA pin acts as an output for all but the data bit cycles at which time the pin is an input for read operations.
ENET_MDIO_DATA	I/O	The ENET_MDIO_DATA pin drives PHY management data into and out of the PHY by way of an access frame consisting of start of frame, read/write indication, PHY address, register address, and data bit cycles. The ENET_MDIO_DATA pin acts as an output for all but the data bit cycles at which time the pin is an input for read operations.
ENET_REVMII_MDIO_RST	I	RevMII PHY reset input.

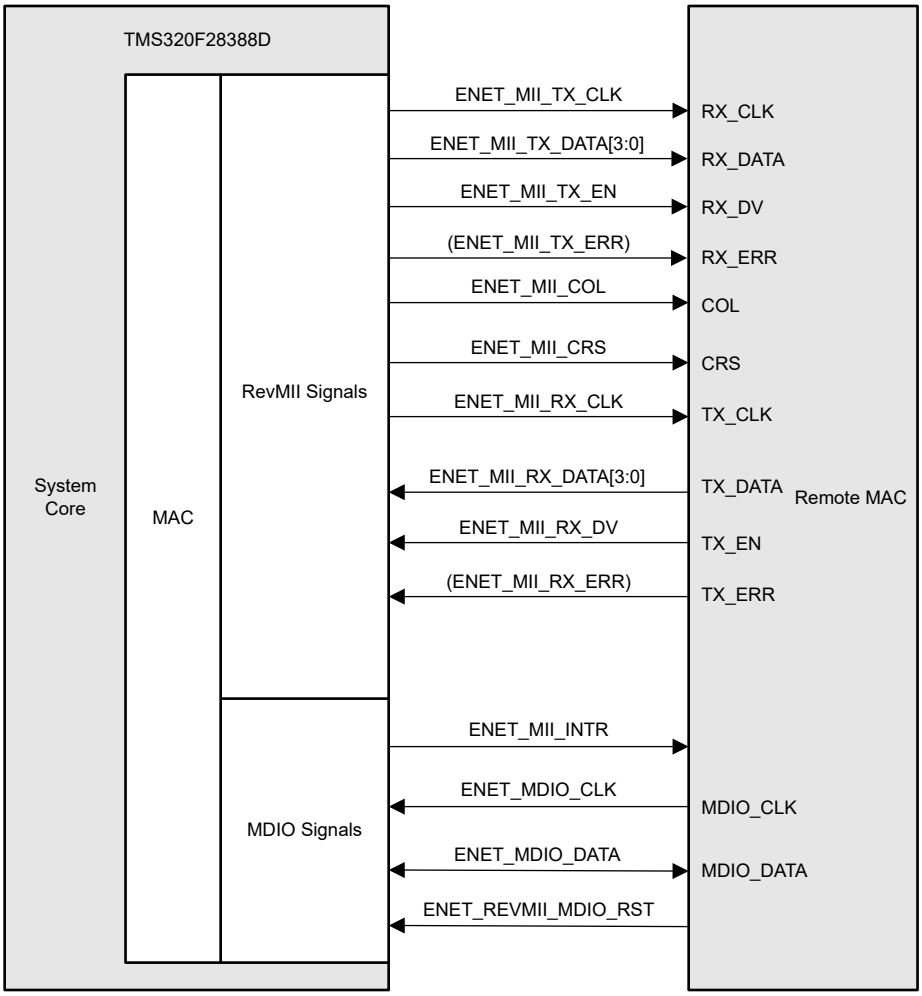


Figure 43-3. RevMII Mode Signals

43.2.1.4 Pulse Per Second Signals

The following two signals listed in Table 43-4 are Pulse Per Second Signals output from the Ethernet module.

Table 43-4. Pulse Per Second Signals

Signal	Type	Description
ENET_PPS0	O	Pulse Per Second 0. Fixed/Flexibile Pulse per Second Output.
ENET_PPS1	O	Pulse Per Second 1. Fixed/Flexibile Pulse per Second Output.

43.2.2 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value from the C28x CPU.

Some IO functionality is defined by GPIO register settings independent of this peripheral. For input signals, the GPIO input qualification should be set to asynchronous mode by setting the appropriate GPxQSELn register bits to 11b. The internal pullups can be configured in the GPyPUD register from the C28x CPU.

See the *General-Purpose Input/Output (GPIO)* chapter for more details on GPIO mux and settings.

43.2.3 MAC Interface Selection

The MAC supports the following interfaces:

- MII
- RMII
- RevMII

The required interface can be selected by programming the PHY_INTF_SEL field in the EMACSS_CTRLSTS register.

43.2.4 Clocks for Ethernet Module

The Ethernet module needs the following clocks:

- Functional clock: This clock uses the Communication manager functional clock CMCLK.
- PTP/Time stamping logic requires a 100-MHz clock that can be sourced from AUXPLLCLKRAW or SYSPLLCLK by programming the ETHDIVSRCSEL and ETHDIV of the System Control Module. Program these fields to get a 100-MHz clock output for PTP block. There is an internal divider of /2 that generates a 50-MHz clock.
- RMII mode clock: The 50-MHz internal clock for RMII mode is generated from the above clock.

Figure 43-4 summarizes the clock options. Refer to the Clock Control module for further details of PLL programmation.

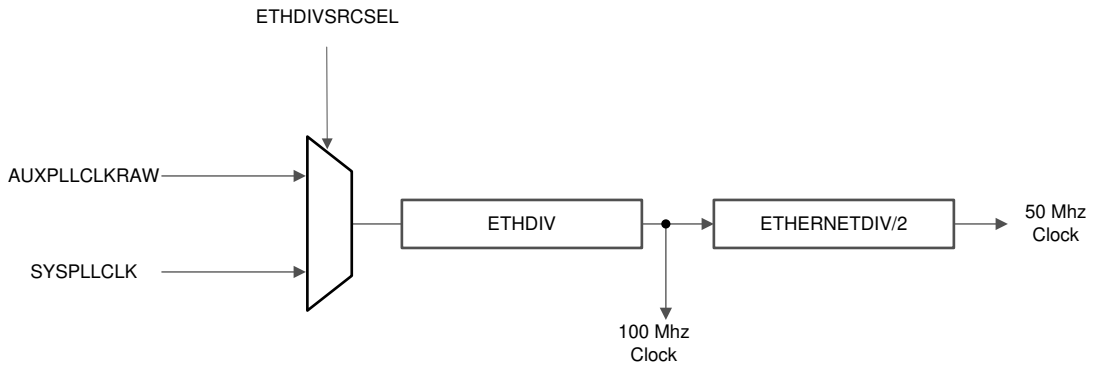


Figure 43-4. Ethernet Clocking

43.2.5 RMII Mode Clocking

For RMII mode operation, the module needs a 50-MHz clock. It can be sourced internally (from the module) or can be sourced externally (from any external component through the RMII_CLK pin input). The CLK_SRC_SEL field of EMACSS_CTRLSTS register programs the source of the RMII clock, as shown in Figure 43-5.

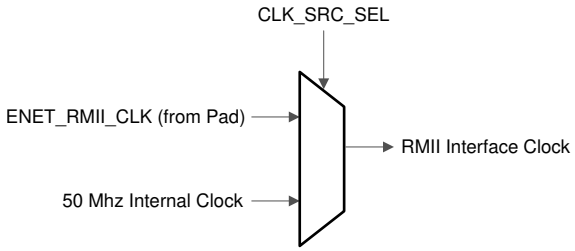


Figure 43-5. RMII Clocking

If the external clock is selected, the ENET_RMII_CLK pin of the device should be provided with the required 50-MHz clock.

If the internal mode is selected, the 50-MHz clock generated internally shall be used to clock the RMII module. Ensure that the clock source and divider are set accordingly to provide a 50-MHz clock.

43.2.6 RevMII Mode Clocking

For RevMII mode operation, the module needs a 25-MHz or 2.5-MHz clock. It can be sourced internally (from the internal 50-MHz clock) or can be sourced externally (from the pad ENET_RMII_CLK). The CLK_SRC_SEL field of the EMACSS_CTRLSTS register programs the source of the RevMII clock, as shown in Figure 43-6.

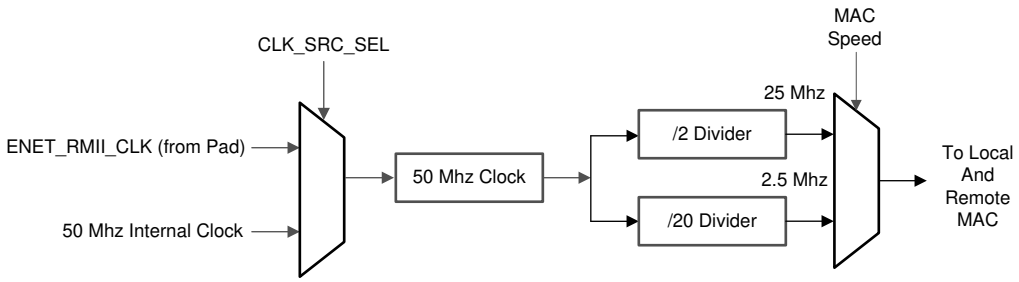


Figure 43-6. RevMII Clocking

If the external clock is selected, the ENET_RMII_CLK pin of the device should be provided with a 50-MHz clock.

If the internal mode is selected, the 50-MHz clock generated internally shall be used to clock the RMII module. Ensure that the clock source and divider are set accordingly to provide a 50-MHz clock.

The internal dividers shall derive the 25-MHz clock (for 100 mbps link) or 2.5-MHz clock (for 10 mbps link) from the internal dividers and also drive the respective clock on the ENET_MII_TXCLK and ENET_MII_RXCLK, respectively.

43.2.7 Configuring Trigger Sources for Time Stamping

The PTP Submodule of Ethernet can also be used to measure the auxiliary timestamps for Software events (triggered through PTP_AUX_TS_SW_TRIGx) and hardware events (through the EPWM cross bar outputs) Auxiliary Trigger 0 and Trigger 1.

The PTP_AUX_TS_TRIG_SELx register can be programmed to select the trigger source as listed in Table 43-5 and shown in Figure 43-7.

Table 43-5. Auxiliary Trigger Sources

PTP_AUX_TS_TRIG_SELx	PTP Auxiliary Trigger Source
0	PTP_AUX_TS_SW_TRIGx
1	EPWM XBAR Output 7
2	EPWM XBAR Output 8
3-31	Reserved

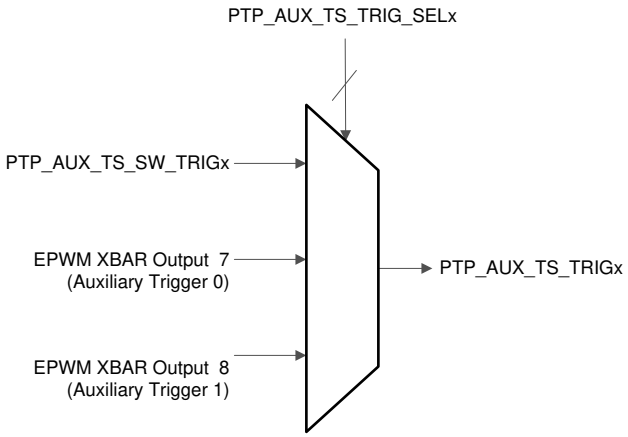


Figure 43-7. Trigger Sources for Auxiliary Timestamping

43.2.7.1 Software Trigger for Time Stamping

The PTP_AUX_TS_SW_TRIGx field of EMACSS_PTPTSSWTRIGx register enables the software trigger for capturing the auxiliary timestamps. This results in a timestamp value pushed into the FIFO for storing timestamps. The Time stamp value can be read from the MAC Auxiliary TimeStamp registers. Corresponding snapshot trigger id can be read from MAC_TIMESTAMP_STATUS register.

43.2.8 Ethernet Interrupts

The Ethernet module provides five interrupts:

- The sbd_interrupt that is a combination of various events generated in the module
- Channel 0,1 Transmit Completion interrupts
- Channel 0,1 Receive Completion interrupts

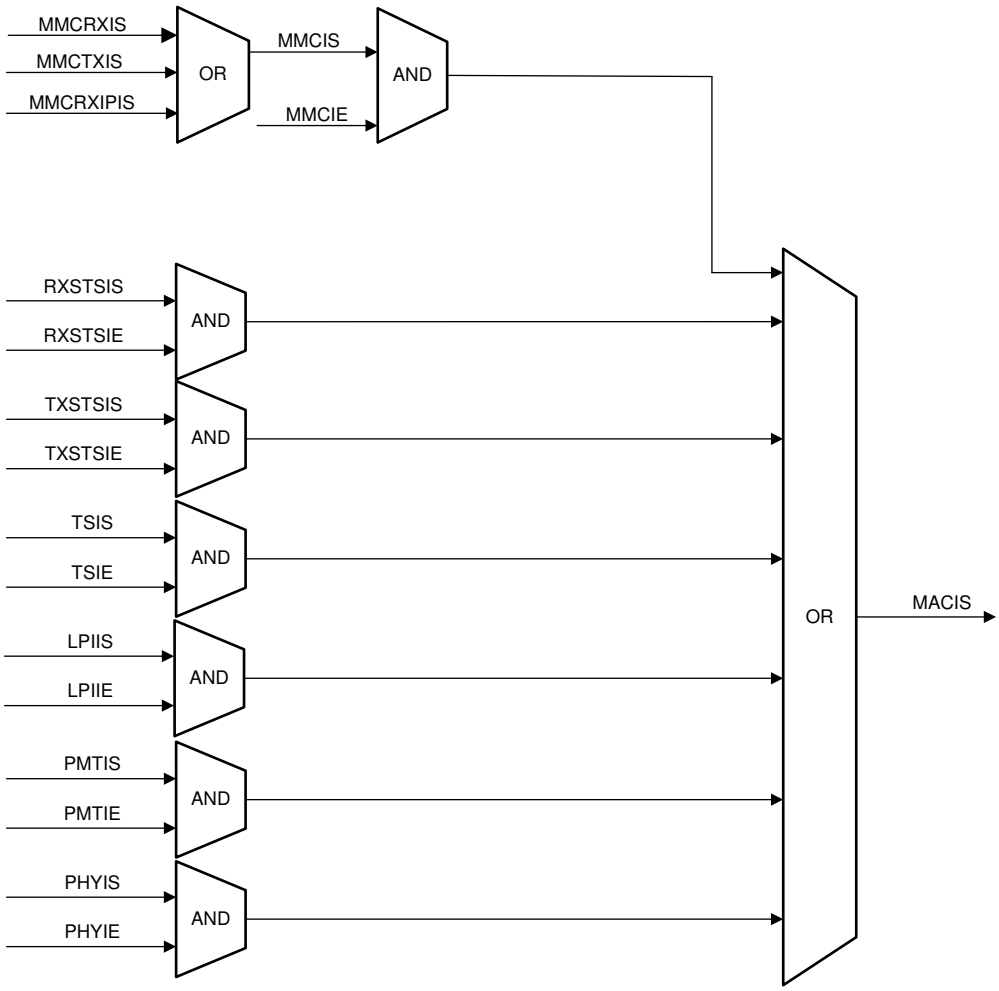


Figure 43-8. MAC Interrupt Sources

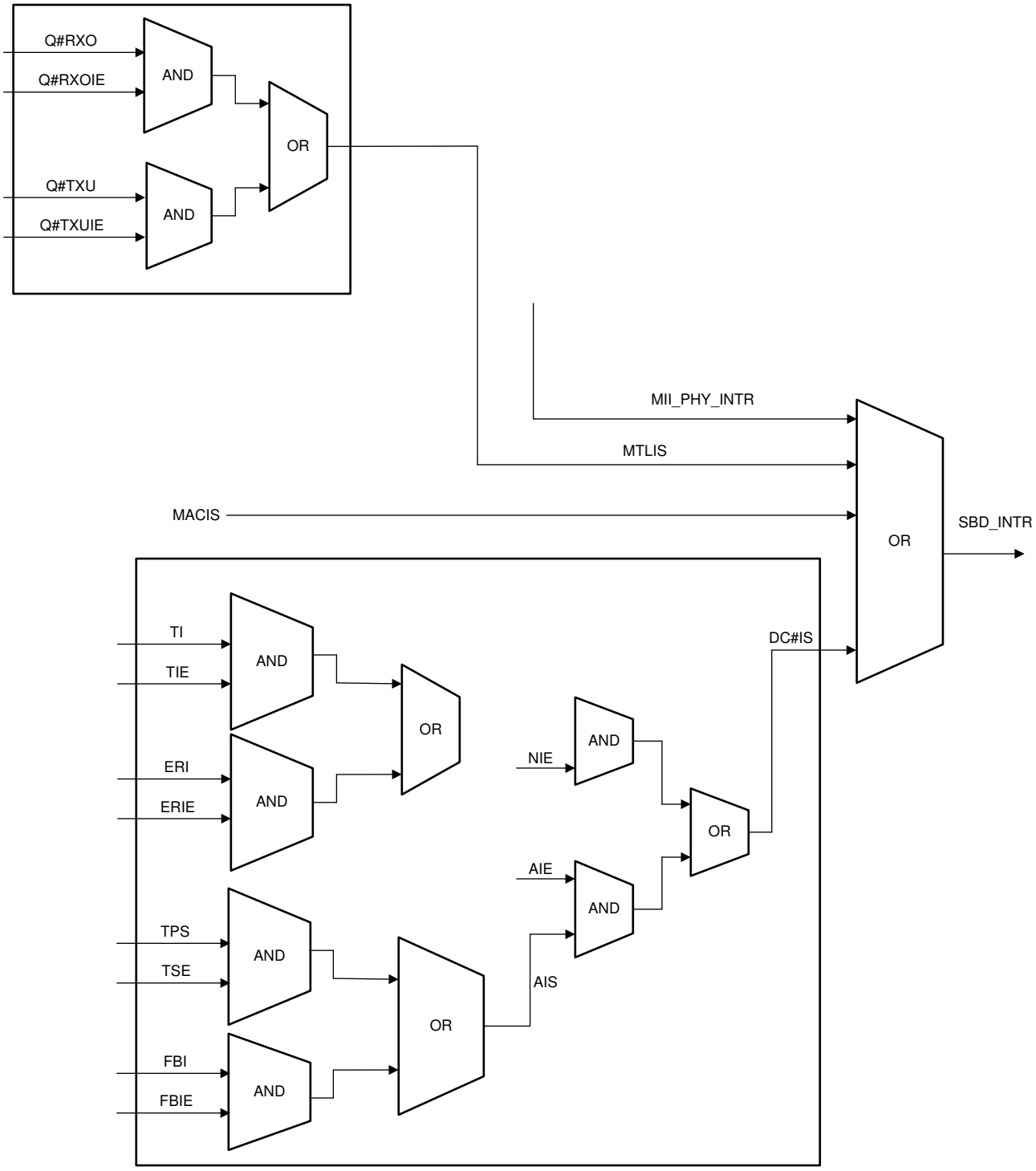


Figure 43-9. Combined SBD_Intr Sources

43.3 Features

The following sections cover the features of the Ethernet module.

43.3.1 Multiple Channels and Queues Support

The Ethernet module supports two queues and channels on Tx and Rx Paths.

Note: In multi queue/channel configurations, enable only Q0/CH0 on Tx and Rx for half-duplex operation. If you want to enable single queue/channel in full-duplex operation, any queue/channel can be enabled.

43.3.1.1 Multiple Queues and Channels in Transmit Path

The Ethernet module supports two Transmit queues. The number of Tx channels is equal to the number of Tx queues. The different scheduling algorithms are detailed below. The algorithm is chosen by the SCHALG field setting of the MTL_Operation mode register.

Fixed Priority Scheme: The fixed priority scheme is the default priority scheme for the DMA channels. In fixed priority scheme, the channel with highest priority always wins the arbitration when it requests the bus.

Table 43-6. Fixed Priority Scheme for DMA Channels

Priority Level	Channel
0 (low)	Channel 0
1	Channel 1

Weighted Strict Priority Scheme

In Weighted Strict Priority (WSP), the weight corresponds to the number of burst transfers given to a channel in one arbitration cycle. The unused burst transfers of one or more channels are reallocated based on the priority. The channel with highest priority gets the unused burst transfer time before it is allocated to a channel with next highest priority.

Weighted Round Robin Scheme (defaulted by design)

The channel priorities are fixed. Channel 1 has the highest priority and Channel 0 has the lowest priority. When a channel uses the allocated burst transfers, the channel with next lower priority is processed. After processing the allocated bandwidth of all channels that had packets to transmit, any unused burst transfer time is allocated to the channel of the highest priority (if required), and then next highest priority (if required), and so on. It is suggested to program the TCW field of the DMA_CHn_Tx_Control Register with nonzero weights, if this scheme is chosen.

Table 43-7. Weight for DMA Channels

TCW Field	Transmit Channel Weight
000	1
001	2
010	3
011	4
100	5
101	6
110	7
111	8

43.3.1.2 Multiple Queues and Channels in Receive Path

The Ethernet module supports two Receive channels and two Rx Queues. In the Rx direction, the MTL Rx Controller selects the Rx DMA for which Rx DMA is transferring or reading the data from the Rx FIFO memory. This scheduling is based on the programming done in the respective MTL_RxQ[x]_Control register.

Each Rx DMA indicates when the Rx DMA is ready to transfer data and the size of the burst-length (number of beats) that the Rx DMA has to transfer. The scheduler checks whether sufficient data (of requested burst length) is available to be transferred to these DMAs and then selects the Rx DMA that gets serviced using the programmed priorities.

Priority Scheme for Tx DMA and Rx DMA

The DWC_ether_qos DMA arbiter supports two types of arbitration: fixed priority and weighted round-robin.

The DMA arbiter performs the arbitration between the Tx and Rx paths of DMA channels for accessing descriptors and data buffers. The DMA arbiter supports two types of arbitration: fixed priority and weighted round-robin. The DA bit of the DMA_Mode register specifies the arbitration scheme (fixed or weighted round-robin) between the Tx and Rx DMA of a channel.

If you have enabled the Tx DMA and Rx DMA of a channel, you can specify which DMA gets the bus when the channel gets the control of the bus. You can set the priority between the corresponding Tx DMA and Rx DMA by using the TXPR field of the DMA_Mode register. For round-robin arbitration, you can use the PR field of the DMA_Mode register to specify the weighted priority between the Tx DMA and Rx DMA.

Table 43-8. Priority Scheme for Tx DMA and Rx DMA

Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Priority Schemes
x	x	x	0	1	Rx always has priority over Tx
0	0	0	0	0	Tx and Rx have equal priority. Rx gets the access first on simultaneous requests
0	0	1	0	0	Rx has priority over Tx in ratio 2:1
0	1	0	0	0	Rx has priority over Tx in ratio 3:1
0	1	1	0	0	Rx has priority over Tx in ratio 4:1
1	0	0	0	0	Rx has priority over Tx in ratio 5:1
1	0	1	0	0	Rx has priority over Tx in ratio 6:1
1	1	0	0	0	Rx has priority over Tx in ratio 7:1
1	1	1	0	0	Rx has priority over Tx in ratio 8:1
x	x	x	1	1	Tx always has priority over Rx
0	0	0	1	0	Tx and Rx have equal priority. Tx gets the access first on simultaneous requests
0	0	1	1	0	Tx has priority over Rx in ratio 2:1
0	1	0	1	0	Tx has priority over Rx in ratio 3:1
0	1	1	1	0	Tx has priority over Rx in ratio 4:1
1	0	0	1	0	Tx has priority over Rx in ratio 5:1
1	0	1	1	0	Tx has priority over Rx in ratio 6:1
1	1	0	1	0	Tx has priority over Rx in ratio 7:1
1	1	1	1	0	Tx has priority over Rx in ratio 8:1

Static Mapping

In Static Mapping mode, all packets of an Rx Queue are connected to a specific DMA channel.

In this mode, all of the packets of an Rx Queue are connected to a specific DMA channel. For example, all the packets from Rx Queue 0 can be routed to a DMA channel by programming Q0MDMACH (bit[3:0]) and Q0DDMACH (bit[7] = 0) of the MTL_RxQ_DMA_Map0 Register.

Similarly, packets from other Rx Queues can be routed to any DMA channel by programming register fields corresponding to each Queue.

Dynamic (Per Packet) Mapping

In Dynamic (per packet) Mapping mode, the destination DMA channel is decide by the MAC core receiver for each packet.

In this mode, the destination DMA channel of a packet being read from a Rx Queue is not constant but decided independently for each packet. For example, if you set the Q1DDMACH bit of the MTL_RxQ_DMA_Map0 register, the static mapping is disabled for Rx Queue 1 and the value in Q1MDMACH is ignored. The destination DMA channel is decided by the MAC receiver for each packet, depending on the following in decreasing order of priority:

1. L3-L4 filter Based DMA Selection
   The TCP/UDP and IP header fields of the received packet are matched against the corresponding values programmed and enabled for comparison in the MAC_L3_L4_Address_Control register. If the match is successful, the DMA channel number programmed in the DMCHN field of the MAC_L3_L4_Address_Control register is selected as the destination DMA channel number provided DMCHEN bit of the same register is set.
   If none of the L3-L4 Registers give a comparison match, then DWC_ether_qos proceeds to the next step.
2. Extended VLAN Based DMA Selection
   Extended routing is applicable only if the VLAN Filter has passed. Routing is only based on Perfect Filter Result. Each perfect filter has a DMA Channel Enable and a DMA Channel Number field that can be programmed. Routing is applicable for a filter, only if the DMA Channel Enable bit is set.
   The frame is routed to the smallest Matching Filter's DMA Channel provided it is enabled. If that filter's DMA Channel number is not enabled, the frame gets routed to Channel 0. For example, if a frame's VLAN tag matches filters 7, 3, and 1. Then the MAC checks if Filter 1's DMA Channel Number is enabled through programming. If yes, the frame gets routed to the programmed value; otherwise it gets routed to DMA. When the inverse filter is enabled; is routed to the least mismatched filter's DMA channel number provided it is enabled. If the DMA Channel enable bit is not set, then the frame is routed based on DA based addressing or to Channel 0.
   If Hash Filter is also enabled, it is used to determine the Filter result only. Routing still depends on the Perfect Filters enabled. If none of the perfect filters are enabled or if all of them are bypassed, then the VLAN Filter based routing does not take place. The frame is routed using DA Based addressing or to Channel 0. If all the perfect filters give a fail result and the Hash Filter has passed, then the VLAN Filter result is a pass but routing is based on DA Based Addressing or to Channel 0. Similar behavior is seen when inverse Filtering is enabled as well.
3. Ethernet DA-Based DMA Selection
   The DA address of the received packet is compared against the programmed DA values in MAC Address Registers. If the address matches any of the programmed values, the corresponding DCS field (when enabled) determines the destination DMA channel number.

If none of the previous operations is able to make a successful match/decision, then the packet is routed to DMA Channel 0 by default.

43.3.1.3 RxQueue to DMA Mapping

The packets in the MTL Rx Queues can be routed to any one of the multiple DMA channels by programming the MTL_RxQ_DMA_Map0 Register (for queues 0, 1, 2, and 3):

The following types of Rx Queue to DMA mapping is possible through programming:

- Static Mapping
- Dynamic (Per Packet) Mapping

43.3.1.4 Selection of Tag Priorities Assigned to Tx and Rx Queues

The DWC_ether_qos controller supports assigning tag priorities to Tx and Rx Queues through programming registers.

The VLAN Tag priorities can be assigned to Tx Queues by programming the corresponding PSTQ# field in the MAC_TxQ_Prty_Map0 register.

The bit corresponding to the VLAN Tag priority can be set in the PSTQ# field for assigning that priority to the Tx Queue. For example, if you want to assign VLAN Tag priority of 3 to Tx Queue 0, set bit [3] in PSTQ0 field.

The same VLAN Tag priority can be set for multiple Tx queues. The Tx packet association to particular Tx Queue is governed by the application, so the MAC does not route the packet based on Tx Queue priority mapping; it is only used for Pause Flow Control (PFC). The settings in the PSTQ# field determines the Tx Queues blocked for transmission when the PFC packet is received with corresponding VLAN Tag priorities enabled.

The VLAN Tag priorities can be assigned to Rx Queues by programming the PSRQ field in the corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers. The bit corresponding to the VLAN Tag priority can be set in the PSRQ field for assigning that priority to the Rx Queue. For example, if you want to assign VLAN Tag priority of 3 to Rx Queue 0, set bit [3] in PSRQ field of MAC_RxQ_Ctrl2 register. The VLAN Tag priority assigned to particular Rx Queue must be unique, that is, more than one Rx Queue cannot be assigned the same VLAN Tag priority. However, more than one VLAN Tag priorities can be assigned to same Rx Queue.

The settings in the PSRQ field is used for VLAN Tagged Rx packet routing to Rx Queues as well as for PFC based Tx flow control. The received VLAN Tagged Rx packet is routed to Rx Queue that has the VLAN Tag priority match. In PFC based Tx flow control, PSRQ field corresponding to a particular Rx Queue is used for enabling VLAN Tag priorities in the PFC packet transmitted when corresponding Rx Queue threshold levels are reached.

43.3.1.5 Rx Side Routing from MAC to Queues

The DWC_ether_qos supports Rx Side Routing from the MAC to Queues.

The MAC routes the Rx packets to the Rx Queues based on following packet types in that order

- Unicast/Multicast destination address packets that fail the destination address filter
- Multicast/Broadcast destination address packets that pass the destination address filter
- VLAN Tag Priority field in VLAN Tagged AV data packets
- VLAN Tagged IEEE 1588 PTP over Ethernet packets
- Untagged IEEE1588 PTP over Ethernet packets
- VLAN Tag Priority field in VLAN Tagged packets
- Untagged packets

The VLAN Tagged Rx packets can be routed based on the priorities assigned to Rx Queues through PSRQ field in corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers and the corresponding Rx Queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0 register.

The untagged IEEE 1588 PTP over Ethernet Rx packets can be routed based on the Rx Queue number specified in the PTPQ field in MAC_RxQ_Ctrl1 register and the corresponding Rx Queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0 register. The VLAN tagged IEEE 1588 PTP over Ethernet Rx packets can be routed based on the priorities assigned to Rx Queues through PSRQ field in corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers or the Rx Queue number specified in the PTPQ field in the MAC_RxQ_Ctrl1 register and the corresponding Rx Queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0 register. This is determined by programming in TPQC field of MAC_RxQ_Ctrl1 register. This type of Rx packet routing is available when IEEE 1588 Timestamp feature support and Multiple Rx The VLAN Tagged IEEE 1588 PTP over Ethernet Rx packets are detected only when 802.1AS mode is disabled (AV8021ASMEN bit in MAC_Timestamp_Control register is set to 0), otherwise this type of Rx packets are routed as generic VLAN Tagged Rx packets.

The multicast/broadcast destination address Rx packets that pass the destination address filter can be routed based on the Rx Queue number specified in the MCBCQ field of MAC_RxQ_Ctrl1 register when enabled through MCBCQEN bit of MAC_RxQ_Ctrl1 register and the corresponding Rx Queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0 register.

The untagged Rx packets can be routed based on the Rx Queue number specified in the UPQ field of MAC_RxQ_Ctrl1 register and the corresponding Rx Queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0 register.

The unicast destination address Rx packets that fail the destination address filter can be routed based on the Rx Queue number specified in the UFFQ field of MAC_RxQ_Routing_Ctrl register when enabled through UFFQE bit

of MAC_RxQ_Routing_Ctrl register, RA bit of MAC_Packet_Filter register is set to 1 and the corresponding Rx Queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0 register.

The multicast destination address Rx packets that fail the destination address filter can be routed based on the Rx Queue number specified in the MFFQ field of MAC_RxQ_Routing_Ctrl register when enabled through MFFQE bit of MAC_RxQ_Routing_Ctrl register, RA bit of MAC_Packet_Filter register is set to 1 and the corresponding Rx Queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0 register.

If the Rx packet cannot be classified in any of the defined packet type for routing, it will be routed through Rx Queue 0.

43.3.2 IEEE 1588 Timestamp Support

The IEEE 1588 defines a Precision Time Protocol (PTP) enables precise synchronization of time in measurement and control systems. This protocol enables heterogeneous systems that include clocks of varying inherent precision, resolution, and stability to synchronize. The protocol supports system-wide synchronization accuracy in the sub-microsecond range with minimal network and local clock computing resources.

The Ethernet module supports the IEEE 1588-2002 (version 1) and IEEE 1588-2008 (version 2). The IEEE 1588-2002 supports PTP transported over UDP/IP and IEEE 1588-2008 supports PTP transported over Ethernet. The Ethernet module provides programmable support for both standards.

The controller supports the following features:

- Provides an option to take snapshot of all packets or only PTP type packets
- Provides an option to take snapshot of only event messages
- Provides an option to take the snapshot based on the clock type: ordinary, boundary, end-to-end transparent, and peer-to-peer transparent
- Provides an option to take the snapshot based on the clock type: ordinary, boundary, end-to-end transparent, and peer-to-peer transparent
- Provides an option to select the node to be a master or slave for ordinary and boundary clock
- Identifies the PTP message type, version, and PTP payload in packets sent directly over Ethernet and sends the status
- Provides an option to measure sub-second time in digital or binary format

43.3.2.1 Feature Description

The different aspects of Time stamping related features is explained in following subsections.

43.3.2.1.1 Clock Types

Ethernet module supports different clock types defined in IEEE 1588-2008.

The Ethernet module supports the following clock types:

- Ordinary Clock
- Boundary Clock
- End-to-End Transparent Clock
- Peer-to-Peer Transparent Clock

Ordinary Clock

The ordinary clock has a single PTP state and a single physical port. In a domain, an ordinary clock supports a single copy of the protocol.

The ordinary clock in a domain supports a single copy of the protocol. The ordinary clock has a single PTP state and a single physical port. In typical industrial automation applications, an ordinary clock is associated with an application device such as a sensor or an actuator. In telecom applications, the ordinary clock can be associated with a timing demarcation device.

The ordinary clock can be a grandmaster or a slave clock. It supports the following features:

- Sends and receives PTP messages. The timestamp snapshot can be controlled as described in MAC_-Timestamp_Control.
- Maintains the data sets such as timestamp values

Table 43-9 shows the messages for which you can take the timestamp snapshot on the receive side for master and slave nodes.

Table 43-9. Ordinary Clock PTM Messages for Snapshot

Master	Slave
Delay_Reg	SYNC

For an ordinary clock, you can take the snapshot of either of the following PTP message types: version 1 or version 2. You cannot take the snapshots for both PTP message types. You can take the snapshot by setting the TSVER2ENA bit and selecting the snapshot mode in MAC_Timestamp_Control.

Boundary Clock

The boundary clock typically has several physical ports communicating with the network. The messages related to synchronization, master-slave hierarchy, and signaling terminate in the protocol engine of the boundary clock and such messages are not forwarded. The PTP message type status given by the MAC helps you to identify the type of message and take appropriate action.

The boundary clock is similar to the ordinary clock except for the following features:

- The clock data sets are common to all ports of the boundary clock.
- The local clock is common to all ports of the boundary clock.

End-to-End Transparent Clock

The end-to-end transparent clock supports the end-to-end delay measurement mechanism between the slave clocks and the master clock. The end-to-end transparent clock forwards all messages like normal bridge, router, or repeater. The residence time of a PTP packet is the time taken by the PTP packet from the Ingress port to the Egress port.

The residence time of a SYNC packet inside the end-to-end transparent clock is updated in the correction field of the associated Follow_Up PTP packet before it is transmitted. Similarly, the residence time of a Delay_Req packet, inside the end-to-end transparent clock, is updated in the correction field of the associated Delay_Resp PTP packet before it is transmitted. Therefore, the snapshot must be taken at both Ingress and Egress ports only for the messages mentioned in Table 43-10. You can take the snapshot by setting the SNAPTYPSEL bits to 10 in the MAC_Timestamp_Control register.

Table 43-10. End to End Transparent Clock: PTP Messages for Snapshot

PTP Messages
SYNC
Delay_Reg

Peer-to-Peer Transparent Clock

In the peer-to-peer transparent clock, the computation of the link delay is based on an exchange of Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up messages with the link peer.

The peer-to-peer transparent clock differs from the end-to-end transparent clock in the way it corrects and handles the PTP timing messages. In all other aspects, it is identical to the end-to-end transparent clock.

In the peer-to-peer transparent clock, the computation of the link delay is based on an exchange of Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up messages with the link peer. The residence time of the Pdelay_Req and the associated Pdelay_Resp packets is added and inserted into the correction field of the associated Pdely_Resp_Followup packet. Therefore, support for taking snapshot for the event messages related to Pdelay is added as shown in Table 43-11.

Table 43-11. Peer to Peer Transparent Clock: PTP Messages for Snapshot

PTP Messages
SYNC
PDelay_reg
Pdelay_Resp

You can take the snapshot by setting the SNAPTYPESEL bit to 11 in MAC_Timestamp_Control register.

Delay Request-Response Mechanism

The system or network is classified into the master and slave nodes for distributing the timing and clock information.

The system or network is classified into the master and slave nodes for distributing the timing and clock information. Figure 43-10 shows the process that PTP uses for synchronizing a slave node to a master node by exchanging PTP messages.

The PTP process is shown in Figure 43-10.

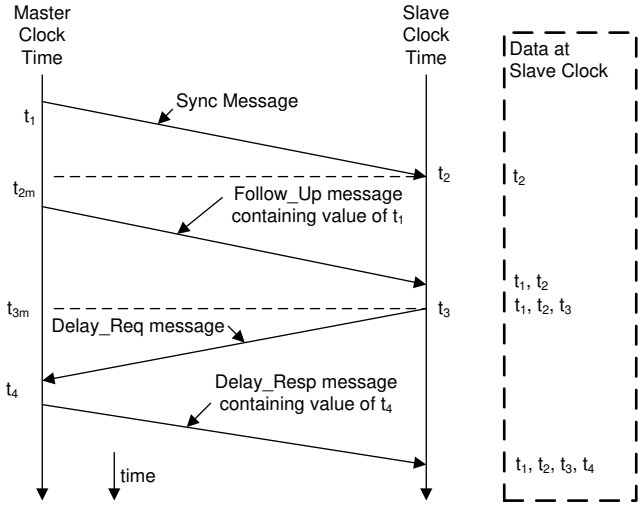


Figure 43-10. Networked Time Synchronization

1. The master broadcasts the PTP Sync messages to all its nodes. The Sync message contains the reference time information of the master. This message leaves the system of the master at t1. This time must be captured for Ethernet ports at MII.
2. The slave receives the Sync message and also captures the exact time, t2, using its timing reference.
3. The master sends a Follow_up message to the slave, which contains t1 information for later use.
4. The slave sends a Delay_Req message to the master and notes the exact time, t3, at which this packet leaves the MII interface.
5. The master receives the message, capturing the exact time t4, at which the message enters its system.
6. The master sends the t4 information to the slave in the Delay_Resp message.
7. The slave uses the four values of t1, t2, t3, and t4 to synchronize its local timing reference to the timing reference of the master.

Most of the PTP implementation is done in the software above the Ethernet/UDP layer. However, the hardware support is required to capture the exact time when specific PTP packets enter or leave the Ethernet port at the MII interface. This timing information must be captured and returned to the software for proper implementation of PTP with high accuracy.

43.3.2.1.1.1 Peer-to-Peer Transparent Clock (P2PTC) Message Support

The IEEE 1588-2008 supports peer-to-peer PTP (Pdelay) message in addition to the SYNC, Delay Request, Follow-up, and Delay Response messages.

The IEEE 1588-2008 supports peer-to-peer PTP (Pdelay) message in addition to the SYNC, Delay Request, Follow-up, and Delay Response messages. Figure 43-11 shows the method to calculate the propagation delay in clocks supporting peer-to-peer path correction.

1. Port 1 issues a Pdelay_Req message and generates a timestamp (t1) for the Pdelay_Req message.
2. Port 2 receives the Pdelay_Req message and generates a timestamp (t2) for this message.
3. Port 2 returns a Pdelay_Resp message and generates a timestamp (t3) for this message. To minimize errors because of any frequency offset between the two ports, Port 2 returns the Pdelay_Resp message as quickly as possible after the receipt of the Pdelay_Req message. Port 2 returns any one of the following:
   - Difference between the timestamps t2 and t3 in the Pdelay_Resp message
   - Difference between the timestamps t2 and t3 in the Pdelay_Resp_Follow_Up message
   - Timestamps t2 and t3 in the Pdelay_Resp and Pdelay_Resp_Follow_Up messages, respectively
4. Port 1 generates a timestamp (t4) on receiving the Pdelay_Resp message.
5. Port 1 uses all four timestamps to compute the mean link delay.

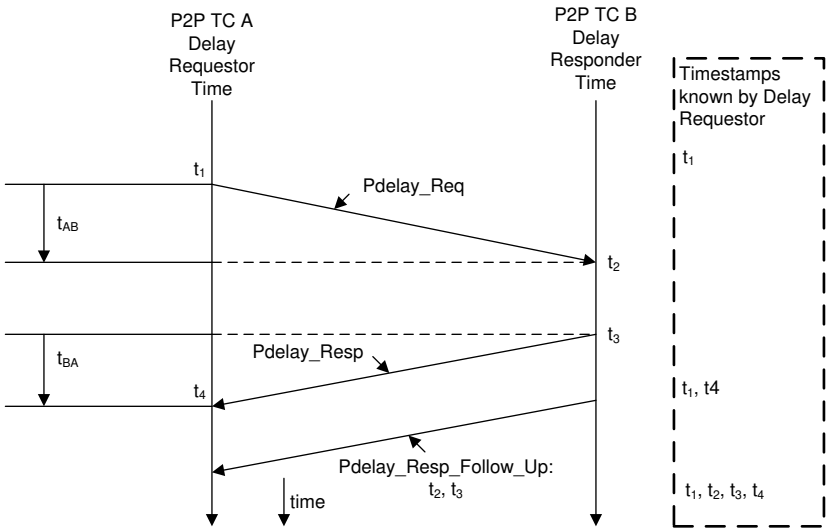


Figure 43-11. Propagation Delay Calculation in Clocks Supporting Peer-to-Peer Path Correction

43.3.2.1.1.2 Timestamp Correction

According to the IEEE 1588 specification, a timestamp must be captured when the PTP message timestamp point (leading edge of the first bit of the octet immediately following the Start Frame Delimiter octet) crosses the boundary between the node and the network. As the MAC takes the timestamp at an internal point far from the actual boundary of the node and network, this captured timestamp is corrected/updated for the ingress/egress path latency (including the delay in the PHY layers). Further correction is done for the inaccuracies/errors introduced due to the clock (MII Tx, Rx clock) being different at the capture point as compared to the PTP clock (clk_ptp_ref_i) that is used to generate the time. The resultant CDC (Clock Domain Crossing) circuits add error depending on the clock period of the MII and PTP clocks.

43.3.2.1.1.3 Ingress Correction

In the Receive side the timestamp captured at the internal snapshot point is delayed (later in time) as compared to the time at which that packet's SFD bit is received at the port's boundary. Therefore, the captured timestamp must be reduced by the ingress latency and the errors in CDC sampling. This correction value must be determined/calculated by the software and written into the MAC_Timestamp_Ingress_Corr_* registers.

The correction value consists of the following three components:

- External latency in the PHY layer between boundary point and the input of the core.
  If the PHY is compliant to the IEEE 802.3 Clause 45 MMD registers, it has a register indicating the maximum and minimum ingress latency. The software can read these registers and determine the average ingress latency in the PHY. Alternatively (if the PHY does not support these registers), the ingress latency must be determined from its datasheet or timing characteristics.
- Internal latency from the input of the core to the internal capture point.
  The internal ingress latency can be read from the MAC_Ingress_Timestamp_Latency register. This is a read-only register and gives the latency in scaledNanoseconds format defined in IEEE 1588 Clause 5.3.2. The latency differs based on the active PHY interface (RMII, so on) and the operating speed. Therefore, the software must read this register after any speed change in the MAC to determine the current internal latency.
- CDC Synchronization.
  The CDC synchronization error is almost equal to two times the clock-period of the PTP clock (clk_ptp_ref_i).

The values determined from these three components should be added by the software and must be written into the TSIC and TSICSNS fields of the MAC_Timestamp_Ingress_Corr_* registers.

Note

The value written to the register must be negative (two's complement as explained below), because it has to be subtracted from the captured timestamp. The MAC Receiver adds the value in this register to the captured timestamp and then gives the resultant value as the timestamp of the received packet.

When TSCTRLSSR bit in MAC_Timestamp_Control register is set, the nanoseconds field of the captured timestamp is in decimal format with a granularity of 1ns. So the Bit31 of TSIC must be set to 1 (for negative value) and bits[30:0] must be written with "10^9 - total ingress_correction_value[nanosecond part]" represented in binary. For example, if the required correction value is -5 ns, then the value is 0xBB9A_C9FB.

When TSCTRLSSR bit in MAC_Timestamp_Control register is reset, the nanoseconds field of the captured timestamp is in binary format with a granularity of ~0.466ns. Therefore, bits[30:0] must be written with "2^31 - total ingress_correction_value" represented in binary with bit[31] = 1.

43.3.2.1.1.4 Egress Correction

In the Transmit side the timestamp captured at the internal snapshot point is earlier (advanced in time) as compared to the time at which that packet's SFD bit is output at the port's boundary. Therefore, the captured timestamp must be compensated by the egress latency and the errors in CDC sampling. This correction value must be determined/calculated by the software and written into the MAC_Timestamp_Egress_Corr_* registers.

The correction value consists of the following three components:

1. External latency in the PHY layer between the output of the core and the boundary of the port and the network
   If the PHY is compliant to the IEEE 802.3 Clause 45 MMD registers, it has a register indicating the maximum and minimum egress latency. The software can read these registers and determine the average egress latency in the PHY. Alternatively (if the PHY does not support these registers), the egress latency must be determined from its datasheet or timing characteristics .
2. Internal latency from the internal capture point and the output of the core
   This internal egress latency can be read from the MAC_Egress_Timestamp_Latency register. This is a read-only register and gives the latency in scaledNanoseconds format defined in IEEE 1588 Clause 5.3.2. The latency will differ based on the active PHY interface (RMII, so on) and the operating speed. Hence the software must read this register after any speed change in the MAC to determine the current internal latency.
3. CDC synchronization error The CDC synchronization error value differs depending on the One-step timestamping mode. When One-step timestamping is enabled, the value = (1 * period of clk_ptp_ref_i + 4 * period of clk_tx_i).
   Otherwise (Two-step timestamping mode), the value = -(2 * period of clk_ptp_ref_i).

43.3.2.1.1.5 Frequency Range of Reference Timing Clock

The timestamp information is transferred across asynchronous clock domains, from the MAC clock domain to the application clock domain. Therefore, a minimum delay is required between two consecutive timestamp captures. This delay is 4 clock cycles of MII and 3 clock cycles of PTP clocks. If the delay between two timestamp captures is less than this delay, the MAC does not take a timestamp snapshot for the second packet.

43.3.2.1.2 Maximum PTP Clock Frequency

The maximum PTP clock frequency is limited by the maximum resolution of the reference time (1 ns resulting in 1 GHz) and the timing constraints achievable for logic operating on the PTP clock, whichever is lesser. In addition, the resolution or granularity of the reference time source determines the accuracy of the synchronization. Therefore, a higher PTP clock frequency gives better system performance.

43.3.2.1.3 Minimum PTP Clock Frequency

The minimum PTP clock frequency depends on the time required between two consecutive SFD bytes and the time taken for synchronizing the time to the MII clock domain. This relationship is given in the following equation:

3 * PTP clock period + 4 * MII clock period <+ Minimum gap between two SFDs

The MII clock frequency is fixed by IEEE specification. Therefore, the minimum PTP clock frequency required for proper operation depends on the operating mode and operating speed of the MAC as shown in Table 43-12.

Note

It is recommended that you use a clock that has constant frequency, preferably a divided application clock

When IEEE 1588 timestamp feature is enabled with internal timestamp, use a PTP clock frequency which is greater than 5 MHz. This is because the 8-bit MAC_Sub_Second_Incre- ment register limits the minimum PTP frequency that can be used to ~4 MHz.

Table 43-12. Minimum PTP Clock Frequency Example

Mode	Minimum Gap Between Two SFDs	Minimum PTP Frequency with External Timestamp Input	Minimum PTP Frequency with Internal Timestamp
10 Mbps full duplex	168 MII clocks (128 clocks for a 64-byte packet + 24 clomcks of in IFG + 16 clocks of preamble)	~45 KHz	5 MHz
10 Mbps half duplex	48 MII clocks (8 clocks for a JAM pattern sent just after SFD because of collision + 24 IFG + 16 preamble)	170 KHz	5 MHz
100 Mbps full duplex	168 MII clocks (128 clocks for a 64-byte packet + 24 clocks of min IFG + 16 clocks of preamble)	~0.5 MHz	5 MHz
100 Mbps half duplex	48 MII clocks (8 clocks for a JAM pattern sent just after SFD because of collision + 24 IFG + 16 preamble)	4.55 MHz	5 MHz

43.3.2.1.4 PTP Processing and Control

Table 43-13 shows the common message header for the PTP messages. This format is taken from the IEEE 1588-2008.

Table 43-13. Message Format Defined in IEEE 1588-2008

Bits									
7	6	5	4	3	2	1	0	Octet	Offset
transportSpecific				messageType[1]				1	0
Reserved				versionPTP				1	1
messageLength								2	2
domainNumber								1	4
Reserved								1	5
flagField								2	6
correctionField								8	8
Reserved								4	16
sourcePortIdentity								10	20
sequenceId								2	30
controlField[1]								1	32
logMessageInterval								1	33

(1) In version 1, controlField is used. In version 2, messageType field is used for detecting different message types.

There are some fields in the Ethernet payload that you can use to detect the PTP packet type and control the snapshot to be taken. These fields are different for the following PTP packets:

- PTP Packets Over IPv4
- PTP Frames Over IPv6
- PTP Packets Over Ethernet

43.3.2.1.5 PTP Packets Over IPv4

Table 43-14 provides information about the fields that are matched to control snapshot for the PTP packets sent over UDP over IPv4 for IEEE 1588 version 1 and 2. The octet positions for the tagged packets are offset by 4. This is based on the IEEE 1588-2008, Annex D and the message format defined in Table 43-13.

Table 43-14. IPv4-UDP PTP Packet Fields Required for Control and Status

Field Matched	Octet Position	Matched Value	Description
MAC Packet Type	12, 13	0x0800	IPv4 datagram
IP version and Header Length	14	0x45	IP version is IPv4
Layer 4 Protocol	23	0x11	UDP
IP Multicast Address (IEEE 1588 version 1)	30, 31, 32, 33	0xE0, 0x00, 0x01, 0x81 (or 0x82, 0x83, 0x84)	Multicast IPv4 addresses allowed:
			■ 224.0.1.129
			■ 224.0.1.130
			■ 224.0.1.131
			■ 224.0.1.132
IP Multicast Address (IEEE 1588 version 2)	30, 31, 32, 33	0xE0, 0x00, 0x01, 0x81 (Hex)	■ PTP Primary multicast address: 224.0.1.129
		0xE0, 0x00, 0x00, 0x6B (Hex)	■ PTP Pdelay multicast address: 224.0.0.107
UDP Destination Port	36, 37	0x013F, 0x0140	■ 0x013F: PTP event messages
			■ 0x0140: PTP general messages
PTP Control Field (IEEE 1588 version 1)	74	0x00, 0x01, 0x02, 0x03, 0x04	■ 0x00: SYNC
			■ 0x01: Delay_Req
			■ 0x02: Follow_Up
			■ 0x03: Delay_Resp
			■ 0x04: Management
PTP Message Type Field (IEEE 1588 version 2)	42 (nibble)	0x9, 0xB, 0xC, 0xD 0x0, 0x1, 0x2, 0x3, 0x8,	■ 0x0: SYNC
			■ 0x1: Delay_Req
			■ 0x2: Pdelay_Req
			■ 0x3: Pdelay_Resp
			■ 0x8: Follow_Up
			■ 0x9: Delay_Resp
			■ 0xA: Pdelay_Resp_Follow_Up
			■ 0xB: Announce
			■ 0xC: Signaling
			■ 0xD: Management
PTP Version	43 (nibble)	0x1, 0x2	■ 0x1: Supports PTP version 1
			■ 0x2: Supports PTP version 2

PTP event messages are SYNC, Delay_Req (IEEE 1588 version 1 and 2) or Pdelay_Req, Pdelay_Resp (IEEE 1588 version 2 only)

43.3.2.1.6 PTP Frames Over IPv6

Table 43-15 provides information about the fields that are matched to control the snapshots for the PTP packets sent over UDP over IPv6 for IEEE 1588 version 1 and 2. The octet positions for the tagged packets are offset by 4. This is based on the IEEE 1588-2008, Annex D and the message format defined in Table 43-13.

Table 43-15. IPv6-UDP PTP Packet Fields Required for Control and Status

Field Matched	Octet Position	Matched Value	Description
MAC Packet Type	12, 13	0x86DD	IP datagram
IP Version	14 (Bits [7:4])	0x6	IP version is IPv6
Layer 4 Protocol	20a (*)	0x11	UDP
PTP Multicast Address	38 – 53	FF0x:0:0:0:0:0:0:181 (Hex) FF02:0:0:0:0:0:0:6B (Hex)	■ PTP Primary multicast address: FF0x:0:0:0:0:0:0:0:181 (Hex)
			■ PTP Pdelay multicast address: FF02:0:0:0:0:0:0:0:6B (Hex)
UDP Destination Port	56, 57a	0x013F, 0x140	■ 0x013F: PTP event message
			■ 0x0140: PTP general messages
PTP Control Field (IEEE1588 version 1)	94a	0x00, 0x01, 0x02, 0x03, 0x04	■ 0x00: SYNC
			■ 0x01: Delay_Req
			■ 0x02: Follow_Up
			■ 0x03: Delay_Resp
			■ 0x04: Management (version1)
PTP Message Type Field (IEEE 1588 version 2)	62a (nibble)	0x0, 0x1, 0x2, 0x3, 0x8, 0x9, 0xB, 0xC, 0xD	■ 0x0: SYNC
			■ 0x1: Delay_Req
			■ 0x2: Pdelay_Req
			■ 0x3: Pdelay_Resp
			■ 0x8: Follow_Up
			■ 0x9: Delay_Resp
			■ 0xA: Pdelay_Resp_Follow_Up
			■ 0xB: Announce
			■ 0xC: Signaling
			■ 0xD: Management
PTP Version	63 (nibble)	0x1, 0x2	■ 0x1: Supports PTP version 1
			■ 0x2: Supports PTP version 2

43.3.2.1.7 PTP Packets Over Ethernet

Table 43-16 provides information about the fields that are matched to control the snapshots for the PTP packets sent over Ethernet for IEEE 1588 version 1 and 2. The octet positions for the tagged packets are offset by 4. This is based on the IEEE 1588-2008, Annex D and the message format defined in Table 43-13.

Table 43-16. PTP Packets Over Ethernet

Field Matched	Octet Position	Matched Value	Description
MAC Destination Multicast Address(1)	0–5	01-1B-19-00-00-00	All PTP messages can use any of the following multicast addresses(2)
		01-80-C2-00-00-0E	■ 01-1B-19-00-00-00
			■ 01-80-C2-00-00-0E(3)
MAC Packet Type	12, 13	0x88F7	PTP Ethernet packet
PTP Control Field (IEEE 1588 version 1)	46	0x00, 0x01, 0x02, 0x03, 0x04	■ 0x00: SYNC
			■ 0x01: Delay_Req
			■ 0x02: Follow_Up
			■ 0x03: Delay_Resp
			■ 0x04: Management
PTP Message Type Field (IEEE 1588 version 2)	14 (nibble)	0x0, 0x1, 0x2, 0x3, 0x8, 0x9, 0xB, 0xC, 0xD	■ 0x0: SYNC
			■ 0x1: Delay_Req
			■ 0x2: Pdelay_Req
			■ 0x3: Pdelay_Resp
			■ 0x8: Follow_Up
			■ 0x9: Delay_Resp
			■ 0xA: Pdelay_Resp_Follow_Up
			■ 0xB: Announce
			■ 0xC: Signaling
			■ 0xD: Management
PTP Version	15 (nibble)	0x1, 0x2	■ 0x1: Supports PTP version 1
			■ 0x2: Supports PTP version 2

(1) The unicast address match of destination addresses (DA), programmed in MAC address 0 to 31, is used if the TSENMACADDR bit of MAC_Timestamp_Control register is set.
(2) IEEE 1588-2008, Annex F
(3) The MAC does not consider the PTP version 1 messages with Peer delay multicast address (01-80-C2-00-00-0E) as valid PTP messages.

43.3.2.1.8 Transmit Path Functions

The MAC captures a timestamp when the Start Packet Delimiter (SFD) of a packet is sent on the MII interface. The packets, for which the timestamps has to be captured can be controlled on per-packet basis. Each Transmit packet can be marked to indicate whether a timestamp should be captured for it.

The MAC does not process the transmitted packets to identify the PTP packets. You need to specify the packets for which you want to capture timestamps.

You can specify the packets by using the control bits in the Transmit descriptor. The MAC returns the timestamp to the software inside the corresponding Transmit descriptor, thus connecting the time- stamp automatically to the specific PTP packet. The 64-bit timestamp information is written to the TDES0 and TDES1 fields. The TDES0 field holds the 32 least significant bits of the timestamp.

43.3.2.1.9 Receive Path Functions

The MAC can be programmed to capture the timestamp of all packets received on the MII interface or to process packets to identify the valid PTP messages. Use the following options of the MAC_Timestamp_Control register to control the snapshot of the time to be sent to the application:

- Enable snapshot for all packets
- Enable snapshot for IEEE 1588 version 1 or version 2 timestamp
- Enable snapshot for PTP packets transmitted directly over Ethernet or UDP-IP-Ethernet
- Enable timestamp snapshot for the received packet for IPv4 or IPv6
- Enable timestamp snapshot only for EVENT messages (SYNC, DELAY_REQ, PDELAY_REQ, or PDELAY_RESP)
- Enable the node to be a master or slave and select the snapshot type This feature controls the type of messages for which snapshots are taken.

Note

The Ethernet module also supports the PTP messages over VLAN packets.

Table 43-17. Timestamp Snapshot Dependency on Register Bits

MAC_Timestamp_Control Register Bit			PTP Messages
SNAPTYPSEL	TSMSTRENA	TSEVNTENA	
00	x	0	SYNC, Follow_Up, Delay_Req, Delay_Resp
00	0	1	SYNC
00	1	1	Delay_Req
01	x	0	SYNC, Follow_Up, Delay_Req, Delay_Resp, Pdelay_Req, Pdelay_Resp, Pdelay_Resp_Follow_Up
01	0	1	SYNC, Pdelay_Req, Pdelay_Resp
01	1	1	Delay_Req, Pdelay_Req, Pdelay_Resp
10	x	x	SYNC, Delay_Req
11	x	x	Pdelay_Req, Pdelay_Resp

The DMA returns the timestamp to the software inside the corresponding Receive Descriptor. The extended status, containing the timestamp message status and the IPC status, is written in normal descriptor RDES1 and the snapshot of the timestamp is written in RDES0 and RDES1 fields of context descriptor. The RDES0 field holds the 32 least-significant bits of the timestamp.

43.3.2.2 IEEE 1588 System Time Source

To get a snapshot of the time, the MAC requires a reference time in 64-bit format as defined in the IEEE 1588-2002 (80-bit format as defined in the IEEE 1588-2008). The Ethernet module provides the following options for using the reference timing source in a node:

43.3.2.2.1 External Timestamp Input

This option takes an external 64-bit timing reference and its clock as input. The clock input is used to synchronize the timing reference to the MAC clock domain.

The 64-bit timing reference is split into the following two 32-bit signals that can be provided by using the TSWH and TSWL registers of the Ethernet Subsystem:

- Upper 32-bits providing the time in seconds
- Lower 32-bits providing the time in nanoseconds

This timing reference is used to timestamp the packets.

43.3.2.2.2 Internal Reference Time

This option takes only the reference clock input and uses it to internally generate the Reference time (also called the system time) and capture timestamps.

The timestamp has the following fields:

- UInteger48 seconds Field. The seconds field is the integer portion of the timestamp in units of seconds. It is 48-bits wide. For example, 2.000000001 seconds are represented as seconds Field = 0x0000_0000_0002.
- UInteger32 nano seconds Field. The nanoseconds field is the fractional portion of the timestamp in units of nanoseconds. For example, 2.000000001 seconds are represented as nanoSeconds = 0x0000_0001. The nanoseconds field supports the following two modes:
  - Digital rollover mode: In this mode, the maximum value in the nanoseconds field is 0x3B9A_C9FF, that is, (10e9-1) nanoseconds.
  - Binary rollover mode: In this mode, the nanoseconds field rolls over and increments the seconds field after value 0x7FFF_FFFF. Accuracy is ~0.466 ns per bit.

43.3.2.2.3 System Time Register Module

The system time generator module is an optional module. It is not available if external time updating is enabled. The 80-bit time is maintained in this module and updated using the input reference clock (clk_ptp_ref_i). This time is the source for taking snapshots (timestamps) of Ethernet packets being transmitted or received at the MII interface.

The system time counter can be initialized or corrected using the coarse correction method. In this method, the initial value or the offset value is written to the Timestamp Update register. For initialization, the system time counter is written with the value in the Timestamp Update register. For system time correction, the offset value is added to or subtracted from the system time.

In the fine correction method, the frequency offset and/or frequency drift of a slave clock (clk_ptp_ref_i) with respect to the master clock (as defined in IEEE 1588-2002) is corrected over a period of time instead of in one clock, as in coarse correction. This helps maintain linear time and does not introduce drastic changes (or a large jitter) in the reference time between PTP Sync message intervals. In this method, an accumulator sums up the contents of the Addend register, as shown in Figure 43-12. The arithmetic carry that the accumulator generates is used as a pulse to increment the system time counter. The accumulator and the addend are 32-bit registers. The accumulator acts as a high-precision frequency multiplier or divider.

Note

You must connect a PTP clock with a frequency higher than the frequency required for the specified accuracy.

This algorithm is shown in Figure 43-12.

The system time update logic requires a 50-MHz clock frequency to achieve 20-ns accuracy. The frequency division is the ratio of the reference clock frequency to the required clock frequency. For example, if the reference clock (clk_ptp_ref_i) is 66 MHz, this ratio is calculated as 66 MHz / 50 MHz = 1.32. Therefore, the default addend value to be set in the register is 2^{32}/ 1.32, 0xC1F07C1F.

If the reference clock drifts lower, for example, to 65 MHz, the ratio is 65 / 50, or 1.3 and the value to set in the addend register is 2^{32} / 1.30, or 0xC4EC4EC4. If the clock drifts higher, for example, to 67 MHz, the addend register must be set to 0xBF0B7672. When the clock drift is nil, the default addend value of 0xC1F07C1F (2^{32} / 1.32) must be programmed.

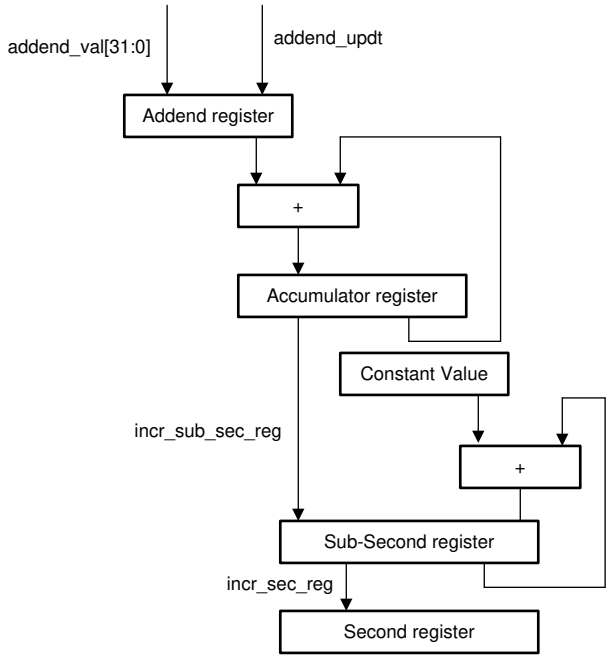


Figure 43-12. System Time Update Using Fine Method

In Figure 43-12, the constant value used to accumulate the sub-second register is decimal 43, which achieves an accuracy of 20 ns in the system time (in other words, it is incremented in 20 ns steps). When External Time Update is enabled, the optional System Time module is not available.

The software must calculate the drift in frequency based on the Sync messages and accordingly update the Addend register.

Initially, the slave clock is set with FreqCompensationValue0 in the Addend register. This value is:

$FreqCompensationValue0 = 2^{32} / FreqDivisionRatio$

The algorithm is:

- At time $MasterSyncTime_n$, the master sends the slave clock a Sync message. The slave receives this message when its local clock is $SlaveClockTime_n$ and computes $MasterClockTime_n$ as $MasterClockTime_n = MasterSyncTime_n + MasterToSlaveDelay_n$
- The master clock count for current Sync cycle, $MasterClockCount_n$, is $MasterClockCount_n = MasterClockTime_n – MasterClockTime_n – 1$ (assuming that MasterToSlaveDelay is the same for Sync cycles n and n – 1)
- The slave clock count for current Sync cycle, $SlaveClockCount_n$, is $SlaveClockCount_n = SlaveClockTime_n – SlaveClockTime_n – 1$
- The difference between master and slave clock counts for current Sync cycle, $ClockDiffCount_n$, is $ClockDiffCount_n = MasterClockTime_n – SlaveClockTime_n$
- The frequency-scaling factor for slave clock, $FreqScaleFactor_n$, is $FreqScaleFactor_n = (MasterClockCount_n + ClockDiffCount_n) / SlaveClockCount_n$
- The frequency compensation value for Addend register, $FreqCompensationValue_n$, is $FreqCompensationValue_n = FreqScaleFactor_n * FreqCompensationValue_n – 1$

Note

Modes of Internal Reference time can be set through the TSCTRLSSR bit in the MAC_Timestamp_Control_Register.

43.3.2.3 IEEE 1588 Higher Word Register

The timestamp maintained in the MAC is 64-bit wide. The overflow to upper 16-bits of seconds register happens once in 130 years. The values of the upper 16-bits of the seconds field can be read from the CSR register.

This is provided to use an additional Higher Word Register.

Note

This higher word register can be used only when the System Time Source is Internal.

43.3.2.4 IEEE 1588 Auxillary Snapshot

The auxiliary snapshot feature allows you to store a snapshot of the system time based on an external event. This feature is independent of whether the system time is generated internally or given as input (through the Subsystem register).

Two auxiliary snapshot input time stamps can be captured on a single common auxiliary snapshot FIFO. The snapshots taken for any input are stored in a common FIFO. The application can read the MAC_Time-stamp_Status register to know the timestamp of which input is available for reading at the top of this FIFO.

The MAC stores these snapshots in a FIFO of depth 4. Only 64-bits of the timestamp are stored in the FIFO. You can read the upper 16-bits of seconds from the MAC_System_Time_Higher_Word_Seconds register when it is present. When a snapshot is stored, the MAC indicates this to the application with an interrupt. The value of the snapshot is read through a FIFO register access. If the FIFO becomes full and an external trigger to take the snapshot is asserted, a snapshot trigger-missed status (ATSSTM) is set in the MAC_Timestamp_Status register. This indicates that the latest auxiliary snapshot of the timestamp is not stored in the FIFO. The latest snapshot is not written to the FIFO when it is full.

When an application reads the 64-bit timestamp from the FIFO, the space becomes available to store the next snapshot. You can clear a FIFO by setting the ATSFC bit in MAC_Auxiliary_Control register. When multiple snapshots are present in the FIFO, the count is indicated in Bits[27:25] of MAC_Timestamp_Status register.

43.3.2.5 Flexible Pulse-Per-Second Output

The Ethernet module provides the flexibility to program the start or stop time, width, and interval of the pulse generated on the ptp_pps_o output.

Note

By default, the module is in the “Fixed Pulse-Per-Second Output” mode and indicated 1 second arrival. The frequency of the PPS output can be changed by setting the PPSCTRL0 field in the MAC_PPS_Control Register.

The Ethernet module provides features such as programming the start or stop time, with the flexible PPS output option.

The Ethernet module supports the following features with the flexible PPS output option:

- Programming the start or stop time in terms of system time.
- Programming the start point of the single pulse and start and stop points of the pulse train in terms of 64-bit system time. The Target Time registers are used to program the start and stop time.
- Programming the stop time in advance, that is, you can program the stop time before the actual start time has elapsed.
- Programming the width between the rising edge and corresponding falling edge of PPS signal output in terms of number of units of sub-second increment value programmed in the MAC_Sub_Second_In- crement register. You can program the width of pulse from 1 to 232-1 units of sub-second increment value.

- Programming the interval, between the rising edges of PPS signal, in terms of number of units of sub-second increment value. You can program the interval between pulses from 1 to 232-1 units of sub-second increment value.
- Option to cancel the programmed PPS start or stop request.
- Error if the start or stop time being programmed has already elapsed.

Note

The PTP Reference clock mentioned in the following sections is the clock at which the system time gets updated. When the TSCFUPDT bit of the MAC_Timestamp_Control register is set to 0, this clock is similar to the clk_ptp_ref_i clock. In the Fine Correction mode, this is the clock tick at which the system time gets updated (using incr_sub_sec_reg shown in Figure 43-12).

43.3.2.5.1 PPS Start or Stop Time

The start time in the Target Time registers can be programmed initially.

You can initially program the start time in the Target Time registers. If you have enabled additional flexible PPS outputs, you need to program the following registers corresponding to an enabled flexible PPS output:

- MAC_PPS1_Target_Time_Seconds and MAC_PPS1_Target_Time_Nanoseconds registers for second Flexible PPS output
- MAC_PPS2_Target_Time_Seconds and MAC_PPS2_Target_Time_Nanoseconds registers for third Flexible PPS output
- MAC_PPS3_Target_Time_Seconds and MAC_PPS3_Target_Time_Nanoseconds registers for fourth Flexible PPS output

If required, you can again program the start or stop time but you can do it only after the earlier programmed value is synchronized to the PTP clock domain. Bit 31 of MAC_PPS#_Target_Time_Nanoseconds register indicates that the synchronization is complete. This enables you to program the start or stop time in advance even before the earlier stop or start time has elapsed.

To ensure proper PPS signal output, you should program advanced system time for the start or stop time. If the application programs a start or stop time that has already elapsed, the MAC sets an error status bit indicating the programming error. If enabled, the MAC also sets the Target Time Reached interrupt event. The application can cancel the start or stop request only if the corresponding start or stop time has not elapsed. If the time has elapsed, the cancel command has no effect.

43.3.2.5.2 PPS Width and Interval

The PPS width and interval are programmed in terms of granularity of system time, that is, number of the units of sub-second increment value.

The PPS width and interval are programmed in terms of granularity of system time, that is, number of the units of sub-second increment value. For example, to have a PPS pulse width of 40 ns and interval of 100 ns with PTP reference clock of 50 MHz, you should program the width and interval to values 2 and 5, respectively. You can achieve smaller granularity by using a faster PTP reference clock.

Before giving the command to trigger a pulse or pulse train on the PPS output, you should program or update the interval and width of the PPS signal output.

43.3.3 Packet Filtering

The Ethernet module provides filtering of incoming packets at Layer 2, Layer 3,Layer 4. VLAN based filtering can also be enabled in this module. This section provides the sequence of filters and then provides details of each type of filter.

43.3.3.1 Packet Filtering Sequence

The sequence shown in Figure 43-13 is valid when all the filters (L2, VLAN, L3, L4) are active. If any of the Layer filters are not enabled, that filter is bypassed and the subsequent filter is applied. A packet that fails any of the filters is discarded. However, the discarded packet can be forwarded to the host based on the register control. For example, when Bit RA of MAC_Packet_Filter register is set to 1, all discarded packets are forwarded to the host but with its packet status indicating the specific filter-failure. If RA=0, Bits VTFE and IPFE of MAC_Packet_Filter register controls if the packets that fail the VLAN filter and Layer 3-4 filter should be discarded or forwarded to the host.

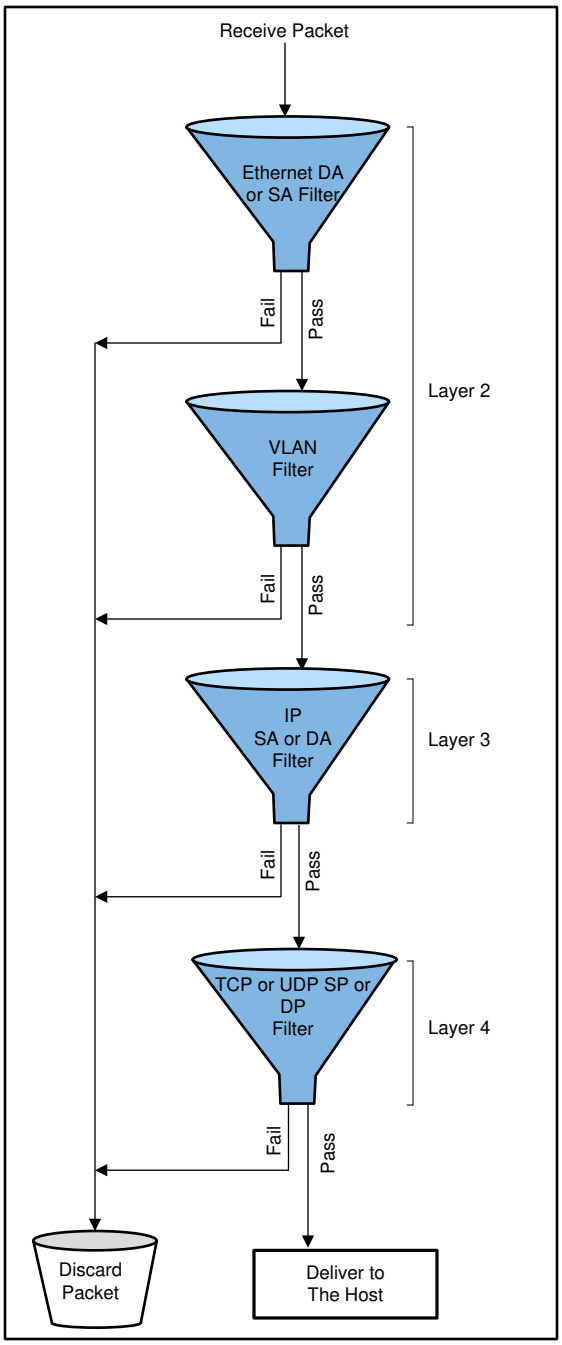


Figure 43-13. Packet Filtering

43.3.3.2 Destination Address Filtering

The Address Filtering Module of the MAC checks the source address (SA) and destination address (DA) fields of each incoming packet. The MAC supports up to 128 MAC addresses for unicast perfect filtering. If perfect filtering is selected (HUC bit of MAC_Packet_Filter register is reset), the MAC compares all 48 bits of received unicast address with the programmed MAC address for any match. The default MacAddr0 is always enabled.

The MacAddr1 to MacAddr127 addresses are selected with an individual enable bit. For MacAddr1 to MacAddr31 addresses, you can mask each byte during comparison with corresponding received DA byte by setting the corresponding Mask Byte Control bit in the register. This enables group address filtering for the DA. The MacAddr32 to MacAddr127 addresses do not have mask control and all 6-bytes of the MAC address are compared with the received 6-bytes of DA.

In hash filtering mode (when HUC bit is set), the MAC performs imperfect filtering for unicast addresses using a 64-bit Hash table. For hash filtering, the MAC uses the upper 6 bits CRC of the received destination address to index the content of the Hash table. A value of 00000 selects Bit 0 of selected register, and a value of 11111 selects Bit 63 of Hash Table register. If the corresponding bit (indicated by the 6-bit CRC) is set to 1, the unicast packet is considered to have passed the Hash filter; otherwise, the packet is considered to have failed the Hash filter.

To program the MAC to pass all multicast packets, set the PM bit in MAC_Packet_Filter register. If the PM bit is reset, the MAC performs the filtering for multicast addresses based on the HMC bit of the MAC_Packet_Filter register.

The multicast address is compared with the programmed MAC Destination Address registers (1–31). Group address filtering is also supported.

In Hash filtering mode, the MAC performs imperfect filtering using a 64-bit Hash table. The MAC uses the upper 6-bits CRC of received multicast address to index the content of the Hash table. A value of 000000 selects Bit 0 of selected register and a value of 111111 selects Bit 63 of the Hash Table register. If the corresponding bit is set to 1, the multicast packet is considered to have passed the Hash filter. Otherwise, the packet is considered to have failed the Hash filter.

To configure the DA filter to pass a packet when its DA matches either the Hash filter or the Perfect filter, set the HPF bit and the corresponding HUC or HMC bits in MAC_Packet_Filter register. This is applicable to both unicast and multicast packets. If the HPF bit is reset, only one of the filters (Hash or Perfect) is applied to received packet.

The MAC does not filter any broadcast packets by default. To program the MAC to reject all broadcast packets, set the DBF bit in MAC_Packet_Filter register.

Table 43-18. Destination Address Filtering

Packet Type	PR	HPF	HUC	DAIF	HMC	PM	DBF	DA Filter Operation
Broadcast	1	X	X	X	X	X	X	Pass
	0	X	X	X	X	X	0	Pass
	0	X	X	X	X	X	1	Fail
Unicast	1	X	X	X	X	X	X	Pass all packets
	0	X	0	0	X	X	X	Pass on Perfect/Group filter match
	0	X	0	1	X	X	X	Fail on Perfect/Group filter match
	0	0	1	0	X	X	X	Pass on Hash filter match
	0	0	1	1	X	X	X	Fail on Hash filter match
	0	1	1	0	X	X	X	Pass on Hash or Perfect/Group filter match
	0	1	1	1	X	X	X	Fail on Hash or Perfect/Group filter match

Table 43-18. Destination Address Filtering (continued)

Packet Type	PR	HPF	HUC	DAIF	HMC	PM	DBF	DA Filter Operation
Multicast	1	X	X	X	X	X	X	Pass all packets
	X	X	X	X	X	1	X	Pass all packets
	0	X	X	0	0	0	X	Pass on Perfect/Group filter match and drop Pause packets if PCF = 0x
	0	0	X	0	1	0	X	Pass on Hash filter match and drop Pause packets if PCF = 0x
	0	1	X	0	1	0	X	Pass on Hash or Perfect/Group filter match and drop Pause packets if PCF = 0x
	0	X	X	1	0	0	X	Fail on Perfect/Group filter match and drop Pause packets if PCF = 0x
	0	0	X	1	1	0	X	Fail on Hash filter match and drop Pause packets if PCF = 0x
	0	1	X	1	1	0	X	Fail on Hash or Perfect/Group filter match and drop Pause packets if PCF = 0x

43.3.3.3 Source Address Filtering

The MAC can perform perfect filtering based on the source address field of received packets. By default, the MAC compares the SA field with the values programmed in the SA registers. You can configure the MAC Address registers[1–31] to use SA instead of DA for comparison by setting Bit 30 of corresponding register.

The MAC also supports group filtering with SA. You can filter a group of addresses by masking one or more bytes of the address. The MAC drops the packets that fail the SA filter if the SAF bit is set in MAC_Packet_Filter register. Otherwise, the result of the SA filter is given as a status bit in the Receive Status word. When the SAF bit is set, the SA filter and DA filter result is AND'ed to decide whether the packet needs to be forwarded. This means that the packet is dropped if either filter fails. The packet is forwarded to the application only if the packet passes both filters in-order.

Table 43-19. Source Address Filtering

Packet Type	PR	SAIF	SAF	SA Filter Operation
Unicast	1	X	X	Pass all packets.
	0	0	0	Pass status on Perfect or Group filter match but do not drop packets that fail
	0	1	0	Fail status on Perfect or Group filter match but do not drop packet
	0	0	1	Pass on Perfect or Group filter match and drop packets that fail
	0	1	1	Fail on Perfect or Group filter match and drop packets that fail

43.3.3.4 Inverse Filtering

For DA and SA filtering, you can invert the filter-match result at the final output by setting the DAIF and SAIF bits of MAC_Packet_Filter register. The DAIF bit is applicable for both Unicast and Multicast DA packets. The result of the unicast or multicast destination address filter is inverted in this mode. Similarly, when the SAIF bit is set, the result of unicast SA filter is reversed.

43.3.3.5 VLAN Filtering

The MAC Receiver can classify the received packets based on VLAN Tag and steer them to a specific Rx DMA channel. The MAC compares the received frame's VLAN Tag with all the enabled and relevant filters and provides a filtering result. If any of the perfect filters give a pass result and if the respective filter's DMA channel Number is enabled, the frame is routed to that DMA Channel.

In addition to filtering, routing can also be done. For more details about routing, see *Extended VLAN Based DMA Selection*.

43.3.3.5.1 Comparison Modes

For each VLAN Tag Filter, the application has the following comparison options:

- It can program the MAC to compare an outer VLAN Tag or an inner VLAN Tag with the programmed VID.
- It can choose if 12 or 16 bits of the VID field need to be compared.
- Type check can be disabled or enabled for each filter; if enabled, the application can choose if the VID comparison is for SVLAN or CVLAN type frames only.
  For example, if a filter is enabled for 16-bit comparison, SVLAN Type, and Outer VLAN Tag, any single or double VLAN Tagged frames with Outer SVLAN Tags are compared with this filter, and a pass or fail result is obtained.

Note

The inner VLAN Tag comparison is applicable only if Double VLAN Tag processing is enabled through the parameter and the MAC VLAN Control Register bit.

The application can enable both Perfect and Hash Filtering. The overall VLAN Filter Result is based on the perfect filter result and the Hash Filter result (if enabled). The filter result is passed to the application as part of the status bits.

Perfect filtering is done based on the MAC_VLAN_Tag_Filter registers. For each VLAN Tag Filter, the MAC will compare the relevant VLAN Tag ID and gives a result. If any one of the VLAN Tag Filters gives a match then the frame is considered to have passed the VLAN Tag Filters. If the frame mismatches all the filters, then the frame is considered to have failed the VLAN filter. This behavior is applicable only when the Inverse Filtering is not enabled in MAC_VLAN_Tag_Ctrl Register.

If inverse Filtering is enabled and the frame has mis-matched all the relevant filters then it is considered to have passed the VLAN filter. If the frame matches any one of the relevant filters then it is considered as a fail. If none of the enabled filters can perform a comparison or if none of the filters are enabled, then the frame is bypassed to the application.

The overall filter result and the programming on the VTFE and RA bits of the MAC Filter Register determine if the frame will be dropped or forwarded to the application. If RA = 1 or VTFE = 0, it does not matter if the filter result is a pass or fail. The frame is always forwarded. If RA = 0 and VTFE = 1, only then, if the VLAN Tag Filter result is a pass does the MAC forward the frame. If the frame is forwarded to the application, then the relevant filter result is indicated through the Status bits.

43.3.3.5.2 Filter Status

The Extended Receive VLAN Filtering & Routing feature provides two status bits to indicate the comparison result of the VLAN tags.

By default, the MAC indicates the VLAN Filter Status through one bit in the status – VF in RDES2. When Extended RX VLAN filtering and routing is enabled, two status bits are used to indicate the comparison result of VLAN tags. The Outer VLAN Tag Filter Pass and Inner VLAN Tag Filter Pass bits are defined in the following positions The status indicated through these bits is highly dependent on the programming as explained.

- Bit 15 – Outer VLAN Tag Filter Status
- Bit 14 – Inner VLAN Tag Filter Status

In ARI status: MAC Filter status:

- Bit 15 – Outer VLAN Tag Filter Status
- Bit 14 – Inner VLAN Tag Filter Status

In MRI Status:

- Bit 47 – Outer VLAN Tag Filter Status
- Inner VLAN Tag Filter Status

Outer VLAN Tag Filter Status (OTS)

- In perfect Filtering, without inverse filtering enabled, if this bit is set, it indicates that the frame's Outer VLAN Tag has matched one of the VLAN Tag Filters.
- If this bit is reset, it indicates that the frame's Outer VLAN Tag has either failed the relevant Outer VLAN Tag Filters or bypassed them.
- If none of the filters are enabled for Outer VLAN Tag Comparison, then this bit is reset.
- If Inverse Filtering is enabled and this bit is set, then the frame's VLAN Tag has passed all the relevant VLAN Tag Filters. If it is reset, then it has failed at least one of or bypassed all the Filters programmed for Outer VLAN Tag Comparison.
- This bit is valid for both Single and Double VLAN Tagged frames.

Inner VLAN Tag Filter Status (ITS)

- In perfect Filtering, without inverse filtering enabled, if this bit is set, it indicates that the frame's Inner VLAN Tag has matched one of the VLAN Tag Filters.
- If this bit is reset, it indicates that the frame's Inner VLAN Tag has either failed the relevant Inner VLAN Tag Filters or bypassed them.
- If none of the filters are enabled for Inner VLAN Tag Comparison, then this bit is reset.
- If Inverse Filtering is enabled and this bit is set, then the frame's VLAN Tag has passed all the relevant VLAN Tag Filters. If it is reset, then it has failed at least one of or bypassed all the Filters programmed for Inner VLAN Tag Comparison.
- This bit is valid for only Double VLAN Tagged frames, when Double VLAN Processing is enabled.

The application must look at the status bits and the programming to determine if the Frame has passed or failed the VLAN Filter.

Table 43-20 and Table 43-21 show the possible Filter combinations and the corresponding filter results. These tables explain the scenarios when Double VLAN Processing and Hash VLAN filter are enabled in the design.

Table 43-20 show the possible values of status bits (OTS and ITS) when at least one Perfect filter is enabled.

- VTIM: VLAN Tag Inverse Match Enable – bit 17 in VLAN_Tag_Ctrl Register.
- HFO: Hash Filter enabled for Outer VLAN Tag Comparison . bit 25 and bit 27 in VLAN_Tag_Ctrl Register.
- HFI: Hash Filter enabled for Inner VLAN Tag Comparison – bit 25 and bit 27 in VLAN_Tag_Ctrl Register.
- PFO – Perfect Filter comparison enabled for Outer VLAN Tag - Any of the MAC_VLAN_Tag_Filter Registers is enabled (bit 16 is set) and programmed for Outer VLAN Tag comparison (bit 20 is set to 0).
- PFI – Perfect Filter comparison enabled for Inner VLAN Tag - Any of the MAC_VLAN_Tag_Filter Registers is enabled (bit 16 is set) and programmed for Inner VLAN Tag comparison (bit 20 is set to 1).
- OTS – Outer VLAN Tag Filter Status
- ITS – Inner VLAN Tag Filter Status

Table 43-20. OTS and ITS Bit Values with At Least 1 Perfect Filter Enabled

VTIM	HFO	HFI	PFO	PFI	OTS	ITS
0	0	0	0	1	0	1/0
0	0	0	1	0	1/0	0
0	0	0	1	1	1/0	1/0
0	1	0	1	1	1/0	1/0
0	1	0	1	0	1/0	0
0	1	0	0	1	1/0	1/0
0	0	1	1	1	1/0	1/0
0	0	1	1	0	1/0	1/0
0	0	1	0	1	0	1/0
1	0	0	0	1	0	1/0
1	0	0	1	0	1/0	0
1	0	0	1	1	1/0	1/0
1	1	0	1	1	1/0	1/0
1	1	0	1	0	1/0	0
1	1	0	0	1	1/0	1/0
1	0	1	1	1	1/0	1/0
1	0	1	1	0	1/0	1/0
1	0	1	0	1	0	1/0

Table 43-21 shows the possible values of status bits (OTS and ITS) when none of the perfect filters are enabled and only the VLAN Hash Filter is enabled.

Table 43-21. OTS and ITS Bit Values with Only VLAN Hash Filter Enabled

VTIM	HFO	HFI	OTS	ITS
0	0	0	0	0
0	1	0	1/0	0
0	0	1	1/0	1/0
1	0	0	1/0	0
1	1	0	1/0	0
1	0	1	1/0	1/0

With no perfect filters enabled, any VLAN packet is considered to have bypassed the perfect filter. If Hash Filter is enabled for one of the Tags, then the respective Status bit depends on the Filter's result. The Status bits are set to 0 if VLAN Hash Filter is also not enabled.

If the value of ITS/OTS is shown as 1/0; then it indicates that the final result is dependent on the enabled relevant filter's result.

Example 1: The second row of table 1-1 indicates that at least one Perfect Filter is enabled for Outer VLAN tag comparison and none of the filters are enabled for Inner VLAN tag comparison. Inverse VLAN Filtering is not enabled. The bit OTS is given as 1/0. If the received frame passes atleast one of the enabled Outer VLAN Tag filters then the bit is set to 1. If the frame doesn't pass any of the enabled Outer VLAN Tag filters, then the bit is set to 0.

Example 2: Last Row of Table 1-1 indicates that Inverse Filtering is enabled, Hash Filter and at least one perfect filter is enabled for Inner VLAN Tag comparison, then if the received frame's Inner VLAN tag mismatches with both the Hash Filter and all the enabled Perfect filters, then the frame will have the ITS bit set to 1 else it is set to 0. OTS will be set to 0 as no comparison is performed.

43.3.3.5.3 Stripping

Each of the VLAN Tags has individual control over stripping. The programming options of Always strip, never strip, strip on pass and strip on fail are available. Inner or Outer VLAN Tag Stripping is based on the pass or fail results of the individual tag. If a tag is bypassed by all the relevant filters, stripping is not applicable for the tag.

- If strip on Pass is enabled for the outer VLAN Tag, then the stripping occurs only if the Outer VLAN tag has passed the relevant Filters. The Outer VLAN Tag Filter Result bit will be set.
- If strip on Fail is enabled for the outer VLAN Tag, then stripping occurs only if the Outer VLAN Tag has failed relevant filters. The Outer VLAN Tag Filter Result Bit will be reset.
- If the Outer VLAN tag of the received frame is bypassed by the entire filter (no comparison has been made), then the tag is not stripped, though the Status Bit is still 0.
- As multiple filters are enabled, it is possible that the received VLAN frame could have matched any one or more of the filters. The VLAN Tag's value is not always deterministic from the filter status bits.
- If the application strips the VLAN Tag based on the filter result, it might lose the VID. So the suggested use is, if Stripping is enabled for any of the tags, the tag can be put in the status. For this the application will have to enable the respective "VLAN Tag in Status" bit - 24 or 31 in the MAC VLAN Tag Control Register.

43.3.3.6 Layer 3 and Layer 4 Filtering

The Ethernet module supports Layer 3 and Layer 4 based packet filtering. The Layer 3 filtering refers to the IP Source or Destination Address filtering in the IPv4 or IPv6 packets whereas Layer 4 filtering refers to the Source or Destination Port number filtering in TCP or UDP.

When Layer 3 and Layer 4 filtering is enabled, the packets are filtered in the following way:

- Matched Packets: The MAC forwards the packets that match all enabled fields to the application along with the status. The MAC gives the matched field status only if the IPC bit of MAC_Configura- tion register is set and one of the following conditions is true:
  - All enabled Layer 3 and Layer 4 fields match
  - At least one of the enabled field matches and other fields are bypassed or disabled

  When multiple Layer 3 and Layer 4 filters are enabled, any filter match is considered as a match. If more than one filter matches, the MAC provides the status of the lowest filter where Filter 0 is the lowest filter and Filter 3 is the highest filter. For example, if Filter 0 and Filter 1 match, the MAC gives the status corresponding to filter 0

Note

The source or destination address and VLAN tag filters (if enabled) have precedence over Layer 3 and Layer 4 filter. This means that a packet which fails the source or destination address or VLAN tag filter is dropped irrespective of the Layer 3 and Layer 4 filter results.

- **Unmatched Packets:** The MAC drops the packets that do not match any of the enabled fields. You can use the inverse match feature to block or drop a packet with specific TCP or UDP over IP fields and forward all other packets.
  the aborted or partial packets can be dropped in the MTL Rx FIFO. If the Rx FIFO operates in the Threshold (cut-through) mode and the threshold is programmed to a small value, such that packet transfer to application starts before the failed Layer 3 and Layer 4 filter results are available, the application may receive a partial packet with appropriate abort status.
- **Non-TCP or UDP IP Packets:** By default, all non-TCP or UDP IP packets are bypassed from the Layer 3 and Layer 4 filters. You can optionally program the MAC to drop all non-TCP or UDP over IP packets.

43.3.3.6.1 Layer 3 Filtering

The Ethernet module supports perfect matching or inverse matching for IP Source Address and Destination Address. In addition, you can match the complete IP address or mask the lower bits matching, that is, compare all bits of the address except the specified lower mask bits.

For IPv6 packets filtering, you can enable the last four data registers of a register set to contain the 128-bit IP Source Address or IP Destination Address. The IP Source Address or Destination Address should be programmed in the order defined in the IPv6 specification, that is, the first byte of the IP Source Address or Destination Address in the received packet is in the higher byte of the register and the subsequent registers follow the same order.

For IPv4 packet filtering, you can enable the second and third data registers of a register set to contain the 32-bit IP Source Address and IP Destination Address. The remaining two data registers are reserved. The IP Source Address or Destination Address should be programmed in the order defined in the IPv4 specification, that is, the first byte of IP Source Address and Destination Address in the received packet in the higher byte of the respective register.

43.3.4 VLAN Support

The VLAN related features are detailed in following subsections.

43.3.4.1 Double VLAN Processing

The Ethernet module supports two VLAN tags, namely inner and outer, for processing double VLANs.This feature is referred as the double VLAN tagging feature in which the MAC can process two VLAN tags. With this feature, the Ethernet module supports the following:

- Insertion, replacement, or deletion of up to two VLAN tags in the Transmit path.
- Packet filtering and stripping based on any one of the two VLAN Tags in the Receive path. Stripping and providing up to two VLAN Tags in the Receive path as a part of the Receive status.

43.3.4.1.1 Transmit Path

Table 43-22 describes the features supported by the MAC on the Transmit side.

Table 43-22. Double VLAN Processing Features in Transmit Path

Feature	Description
Support for C-VLAN and	The inner or outer VLAN tag can be of C-VLAN and S-VLAN type. The VLAN type is specified through the CSVL bit of MAC_VLAN_Incl and MAC_Inner_VLAN_Incl registers. The Ethernet module supports processing of any sequence of outer and inner VLAN tags.
S-VLAN Tag types	Note: The Ethernet module does not support the C-VLAN S-VLAN sequence. The MAC does not check whether the packet provided by the application has a valid sequence of the VLAN Tag types or the insertion or replacement operation results in invalid sequence of VLAN Tag type. Therefore, the application must provide correct sequence of VLAN Tag types and program the MAC in such a way that it results in correct sequence of VLAN Tag types in the transmitted packet. The application must ensure the following: ■ The inner tag should not be S-VLAN when outer C-VLAN Tag insertion is enabled. ■ The outer tag should not be C-VLAN when inner S-VLAN Tag insertion is enabled. ■ The inner tag should not be S-VLAN when outer tag should be replaced with C-VLAN. ■ The outer tag should not be C-VLAN when inner tag should be replaced with S-VLAN.
VLAN Tag deletion	You can enable the VLAN tag deletion for outer or inner tag through VLC field in the MAC_VLAN_Incl or MAC_Inner_VLAN_Incl register, respectively. When VLAN deletion is enabled, the MAC deletes the tag present at the corresponding position. When a packet has only one tag, it is considered as the outer tag. If inner tag deletion is enabled and the packet has only one tag, the MAC does not delete the tag.
VLAN Tag Insertion or Replacement	You can enable the VLAN tag insertion or replacement for outer or inner tag through VLC field in the MAC_VLAN_Incl or MAC_Inner_VLAN_Incl register, respectively. When VLAN tag insertion or replacement is enabled, the VLTI bit in the MAC_VLAN_Incl or MAC_Inner_VLAN_Incl register is used to determine whether the VLAN tag should be taken from the register or the Control Word.

43.3.4.1.2 Receive Path

Table 43-23 describes the features supported by the MAC on the Receive side and the corresponding bits in the MAC_VLAN_Tag register.

Table 43-23. Receive Path

Feature	Description
Outer or inner VLAN tag-based filtering	The MAC can filter packets based on the outer or inner VLAN tag through the ERIVLT bit.
C-VLAN or S-VLAN tag-based filtering	The MAC can filter packets based on the C-VLAN or S-VLAN type based on the ERSVLM bit.
Outer and Inner VLAN Tag stripping	The MAC can strip the outer and inner VLAN Tags from received frame based on the EVLS and EIVLS bits.
16-bit outer and inner VLAN Tag and Type in Rx status	The MAC can provide the 16-bit outer and inner VLAN Tag and Type in the Rx status based on the EVLRXS and EIVLRXS bits, respectively.
Disabling or skipping checking of outer VLAN Tag type	The MAC can disable or skip checking of outer VLAN Tag type to match C-VLAN or S-VLAN based on the DOVLTC bit.

43.3.4.2 Double VLAN-Related Registers

Following are the Double VLAN-related registers

- MAC_VLAN_Incl
- MAC_Inner_VLAN_Incl
- MAC_VLAN_Tag

43.3.4.3 Source Address and VLAN Insertion, Replacement, or Deletion

Ethernet module supports SA and VLAN insertion, replacement and deletion.

The source address (SA) and VLAN fields are Tx packet-related control information that are provided as part of the control word through ATI interfaces. Ethernet module supports the feature to insert or replace the source address based on the information in the MAC Address Registers, and also the feature to insert, replace or delete the VLAN fields (VLAN Type and VLAN Tag) based on the setting of the VLTI bit in the MAC_VLAN_Incl register. You can enable the SA insertion or replacement feature for all Transmit packets or selective packets. Similarly, you can enable the VLAN insertion, replacement, or deletion feature for all Tx packets or selective packets.

The software can use the SA insertion or replacement feature to instruct the MAC to do the following for Tx packets:

- Insert the content of the MAC Address Registers in the SA field
- Replace the content of the SA field with the content of the MAC Address Registers

When SA insertion is enabled, the application must ensure that the packets sent to the MAC do not have the SA field. The MAC does not check whether the SA field is present in the Transmit packet and it inserts the content of MAC Address Registers in the SA field. Similarly, when SA replacement is enabled, the application must ensure that the SA field is present in the packets sent to the MAC. The MAC replaces the six bytes following the Destination Address field in the Transmit packet with the content of the MAC Address Registers.

You can enable the SA insertion or replacement feature for all Transmit packets or selective packets:

- Enabling SA insertion or replacement for all packets.
  To enable this feature for all packets, program the SARC field of the MAC_Configuration register.
- Enabling SA insertion or replacement for selective packets

Program the SA Insertion Control field (Bits[25:23] of TDES3) in the first Transmit descriptor of the packet. When Bit 25 of TDES3 is set, the SA Insertion Control field indicates insertion or replace- ment by MAC Address1 registers. When Bit 25 of TDES3 is reset, it indicates insertion or replace- ment by MAC Address 0 registers.

If MAC Address1 Registers are not enabled, the MAC Address0 registers are used for insertion or replacement irrespective of the value of the most-significant bit of the SA Insertion Control field.

43.3.4.3.1 Programming VLAN Insertion, Replacement, or Deletion

The software can use the VLAN insertion, replacement, or deletion feature to instruct the MAC to do the following for Tx packets:

- Delete the VLAN Type and VLAN Tag fields

- Insert or replace the VLAN Type and VLAN Tag fields
- Insertion or replacement is done based on the setting of VLTI bit in the MAC_VLAN_Incl register as described in Table 43-24

Table 43-24. VLAN Insertion or Replacement Based on VLTI Bit

VLTI Bit Condition	Description
Set	The MAC inserts or replaces the following: ■ VLAN Type field (C-VLAN or S-VLAN as indicated by the CSVL bit of MAC_VLAN_Incl register) ■ VLAN Tag field with cntent of the VT field of Transmit context descriptor of the packet
Reset	The MAC inserts or replaces the following: ■ VLAN Type field (C-VLAN or S-VLAN as indicated by the CSVL bit of MAC_VLAN_Incl register) ■ VLAN Tag field with the VLT field of MAC_VLAN_Incl register

When VLAN replacement or deletion is enabled, the MAC checks if the VLAN Type field (0x8100 or 0x88a8) is present after the DA and SA fields in the Transmit packet. The replace or delete operation does not occur if the VLAN Type field is not detected in two bytes following the DA and SA fields. However, when VLAN insertion is enabled, the MAC does not check the presence of VLAN Type field in the Transmit packet and just inserts the VLAN Type and VLAN Tag fields.

You can enable the VLAN insertion, replacement, or deletion feature for all Tx packets or selective packets:

- Enabling VLAN insertion, replacement, or deletion for all packets
  To enable this feature for all packets, program the VLC and VLP fields of MAC_VLAN_Incl register.
- Enabling VLAN insertion, replacement, or deletion for selective packets
  Program the VTIR field of TDES2 Normal Descriptor
  In addition, the VLP (VLAN Priority control) bit must be reset in Register 24 (for outer VLAN) and Register 25 (in inner VLAN) for the MAC to take the control inputs from the host

43.3.4.4 Queue/Channel Based VLAN Tag Insertion on Tx

Ethernet module supports channel/queue based VLAN tag insertion on all transmitted packets.

Queue/Channel specific VLAN tag registers are accessed using indirect addressing via the MAC_VLAN_Incl register. VLAN type and tag value can be independently programmed for each queue/channel.

When you enable this feature and the CBTI field is set in the MAC_VLAN_Incl register, the VLAN tag is inserted on every packet that is transmitted from a queue/channel, with the programmed tag value taken from queue/channel specific VLAN tag register.

43.3.5 TCP/IP Offloading Features

Communication protocols such as TCP and UDP implement checksum fields, which help determine the integrity of data transmitted over a network. The most widespread use of Ethernet is to encapsulate TCP and UDP over IP datagrams. Therefore, the MAC has an optional Checksum Offload Engine (COE) to support checksum calculation and insertion in the Transmit path, and error detection in the Receive path.

The TCP Segmentation Offload (TSO) engine is useful in offloading the TCP segmentation functions to the hardware.

43.3.5.1 Transmit Checksum Offload Engine

The MAC has a Checksum Offload Engine (COE) to support checksum calculation and insertion in the Transmit path, using which, the software can offload the task of checksum insertion to the hardware. In the transmit path MAC calculates the checksum and inserts it in the Tx packet. This feature helps in reducing the load on the software and can improve the overall throughput of the system.

The checksum offload engine module supports two types of checksum calculation and insertion. The checksum engine can be controlled for each packet by setting the CIC bits (TDES3 Bits[17:16]).

Note

1. The checksum for TCP, UDP, or ICMP is calculated over a complete packet, and then inserted into its corresponding header field. Because of this requirement, when this function is enabled, the Tx FIFO automatically operates in the store-and-forward mode even if the Ethernet module is configured for Threshold (cut-through) mode.
2. See IETF specifications RFC 791, RFC 793, RFC 768, RFC 792, RFC 2460, and RFC 4443 for IPv4, TCP, UDP, ICMP, IPv6, and ICMPv6 packet header specifications, respectively.

43.3.5.1.1 IP Header Checksum Engine

In IPv4 datagrams, the integrity of the header fields is indicated by the 16-bit Header Checksum field (the eleventh and twelfth bytes of the IPv4 datagram). The COE detects an IPv4 datagram when the Type field of Ethernet packet has the value 0x0800 and the Version field of IP datagram has the value 0x4. The checksum field of the input packet is ignored during calculation and replaced with the calculated value.

Note

IPv6 headers do not have a checksum field. Therefore, the COE does not modify the IPv6 header fields.

The result of this IP header checksum calculation is indicated by the IP Header Error status bit in the Transmit status (Bit 0). This status bit is set whenever the values of the Ethernet Type field and the Version field of IP header are not consistent, or when the Ethernet packet does not have enough data, as indicated by the IP header Length field. In other words, this bit is set when an IP header error is asserted under the following circumstances:

- For IPv4 datagrams:
  - The received Ethernet type is 0x0800, but the Version field of IP header is not equal to 0x4.
  - The IPv4 Header Length field indicates a value less than 0x5 (20 bytes).
  - The total packet length is less than the value given in the IPv4 Header Length field.
- For IPv6 datagrams:
  - The Ethernet type is 0x86dd but the IP header Version field is not equal to 0x6.
  - The packet ends before the IPv6 header (40 bytes) or extension header (as given in the corre- sponding Header Length field in an extension header) is completely received.

43.3.5.1.2 TCP/UDP/ICMP Checksum Engine

The TCP/UDP/ICMP Checksum Engine processes the IPv4 or IPv6 header (including extension headers) and determines whether the encapsulated payload is TCP, UDP, or ICMP. The checksum is calculated for the TCP, UDP, or ICMP payload and inserted into its corresponding field in the header. The Tx COE can work in the following two modes:

- The TCP, UDP, or ICMPv6 pseudo-header is not included in the checksum calculation and is assumed to be present in the Checksum field of the input packet. This engine includes the Checksum field in the checksum calculation, and then replaces the Checksum field with the final calculated checksum.
- The engine ignores the Checksum field, includes the TCP, UDP, or ICMPv6 pseudo-header data into the checksum calculation, and overwrites the checksum field with the final calculated value.

Note

For ICMP-over-IPv4 packets, the Checksum field in the ICMP packet must always be 16'h0000 in both modes, because pseudo-headers are not defined for such packets. If it does not equal 16’h0000, an incorrect checksum may be inserted into the packet.

The result of this operation is indicated by the Payload Checksum Error status bit in the Transmit Status vector (Bit 12). This engine sets the Payload Checksum Error status bit when it detects that the packet has been forwarded to the MAC Transmitter engine in the store-and-forward mode without the end of packet (EOP) being

written to the FIFO, or when the packet ends before the number of bytes indicated by the Payload Length field in the IP Header is received. When the packet is longer than the indicated payload length, the COE ignores them as stuff bytes, and no error is reported. When this engine detects the first type of error, it does not modify the TCP, UDP, or ICMP header. For the second error type, it still inserts the calculated checksum into the corresponding header field.

Table 43-25 describes the functions supported by Transmit Checksum Offload engine based on the packet type. When the MAC does not insert the checksum, it is indicated as “No” in Table 43-25.

Table 43-25. Transmit Checksum Offload Engine Functions for Different Packet Types

Packet Type	Hardware IP Header Checksum Insertion	Hardware TCP/UDP Checksum Insertion
Non-IPv4 or IPv6 packet	No	No
IPv4 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in MAC) but less than or equal to the frame size constraint specified in “Transmit Checksum Offload Engine”	Yes	Yes
IPv6 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in MAC) but less than or equal to the frame size constraint specified in “Transmit Checksum Offload Engine”	Not Applicable	Yes
IPv4 with TCP, UDP, or ICMP	Yes	Yes
IPv4 packet has IP options (IP header is longer than 20 bytes)	Yes	Yes
Packet is an IPv4 fragment	Yes	No
IPv6 packet with the following next header fields in main or extension headers:		
■ Hop-by-hop options (in IPv6 main header)	Not Applicable	Yes
■ Hop-by-hop options (in IPv6 extension header)	Not Applicable	No
■ Destinations options	Not Applicable	Yes
■ Routing (with segment left 0)	Not Applicable	No
■ Routing (with segment left > 0)	Not Applicable	No
■ TCP, UDP, or ICMP	Not Applicable	Yes
■ Authentication	Not Applicable	No
■ Any other next header field in main or extension headers	Not Applicable	No
IPv4 packet has TCP header with Options fields	Yes	Yes
IPv4 Tunnels:		
■ IPv4 packet in an IPv4 tunnel	Yes (IPv4 tunnel header)	No
■ IPv6 packet in an IPv4 tunnel	Yes (IPv4 tunnel header)	No
IPv6 Tunnels:		
■ IPv4 packet in an IPv6 tunnel	Not applicable	No
■ IPv6 packet in an IPv6 tunnel	Not applicable	No
IPv4 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled).	Yes	Yes
IPv6 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled).	Not applicable	Yes
IPv4 frames with security features (such as encapsulated security payload)	Yes	No
IPv6 frames with security features (such as encapsulated security payload)	Not applicable	No

43.3.5.2 Receive Checksum Offload Engine

Ethernet module provides the Checksum Offload Engine that is used to detect any error in an IPv4 or IPv6 packet in the receive path. The MAC verifies the checksum field of the received packet with the internally calculated checksum and provides the status.

The Receive Checksum Offload engine is used to detect errors in IP packets by calculating the header checksum and further matching it with the received header checksum. This engine also identifies a TCP, UDP, or ICMP payload in received IP packets and calculates the checksum of such payloads appropriately.

Here, both IPv4 and IPv6 packet in the received Ethernet packets are detected and processed for data integrity. The MAC receiver identifies IPv4 or IPv6 packets by checking for value 0x0800 or 0x86DD, respectively, in the Type field of the received Ethernet packet. This identification is applicable to single VLAN-tagged packets. It is also applicable to double VLAN-tagged packets when the Enable Double VLAN Processing option is selected and the EDVLP bit of the MAC_VLAN_Tag register is set.

The Rx COE calculates the IPv4 header checksums and checks that they match the received IPv4 header checksums. The result of this operation (pass or fail) is given to the RFC module for insertion into the receive status word. The IP Header Error bit is set for any mismatch between the indicated payload type (Ethernet Type field) and the IP header version, or when the received packet does not have enough bytes, as indicated by the Length field of the IPv4 header (or when fewer than 20 bytes are available in an IPv4 or IPv6 header).

This engine also identifies a TCP, UDP, or ICMP payload in the received IP datagrams (IPv4 or IPv6) and calculates the checksum of such payloads properly, as defined in the TCP, UDP, or ICMP specifications. This engine includes the TCP, UDP, or ICMPv6 pseudo-header bytes for checksum calculation and checks whether the received checksum field matches the calculated value. The result of this operation is given as a Payload Checksum Error bit in the receive status word. This status bit is also set if the length of the TCP, UDP, or ICMP payload does not match the expected payload length given in the IP header.

Table 43-26 describes the functions supported by the Rx COE based on the packet type. When the payload of an IP packet is not processed (indicated as "No" in Table 43-26), the information (whether the checksum engine is bypassed or not) is given in the receive status.

Note

The MAC does not append any payload checksum bytes to the received Ethernet packets.

Table 43-26. Receive Checksum Offload Engine Functions for Different Packet Types

Packet Type	Hardware IP Header Checksum Checking	Hardware TCP/UDP/ICMP Checksum Checking
Non-IPv4 or IPv6	No	No
IPv4 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in the MAC)	Yes	Yes
IPv6 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in the MAC)	Not Applicable	Yes
IPv4 with TCP, UDP, or ICMP	Yes	Yes
IPv4 header's protocol field contains a protocol other than TCP, UDP, or ICMP	Yes	No
IPv4 packet has IP options (IP header is longer than 20 bytes)	Yes	Yes
Packet is an IPv4 fragment	Yes	No
IPv6 packet with the following next header fields in main or extension headers:	Not Applicable	Yes
■ Hop-by-hop options (in IPv6 main header)	Not Applicable	No
■ Hop-by-hop options (in IPv6 extension header)	Not Applicable	Yes
■ Destinations options	Not Applicable	Yes
■ Routing (with segment left 0)	Not Applicable	No
■ Routing (with segment left > 0)TCP, UDP, or ICMP	Not Applicable	Yes
■ Any other next header field in main or extension headers	Not Applicable	No

Table 43-26. Receive Checksum Offload Engine Functions for Different Packet Types (continued)

Packet Type	Hardware IP Header Checksum Checking	Hardware TCP/UDP/ICMP Checksum Checking
IPv4 packet has TCP header with Options fields	Yes	Yes
IPv4 Tunnels:		
■ IPv4 packet in an IPv4 tunnel	Yes (IPv4 tunnel header)	No
■ IPv6 packet in an IPv4 tunnel	Yes (IPv4 tunnel header)	No
IPv4 Tunnels:		
■ IPv4 packet in an IPv6 tunnel	Not Applicable	No
■ IPv6 packet in an IPv6 tunnel	Not Applicable	No
IPv4 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled)	Yes	Yes
IPv6 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled)	Not Applicable	Yes
IPv4 frames with security features (such as encapsulated security payload)	Yes	No
IPv6 frames with security features (such as encapsulated security payload)	Not Applicable	No

43.3.5.3 TCP/IP Segmentation Offload (TSO) Engine

The TCP Segmentation Offload (TSO) engine is useful in offloading the TCP segmentation functions to the hardware.

It also supports UDP Segmentation Offload (USO) in which the UDP payload is segmented in the hardware. There are no sequencing/ordering controls available or updated in the segments, so it can be used in point to point applications in which out of order packets are not expected by the receiver. The description and flow of TSO mentioned in this section is same as USO, any deviation is provided as notes.

In the TCP segmentation offload (TSO) feature, the DMA splits a large TCP packet into multiple small packets and passes these packets to the MTL as shown in Figure 43-14

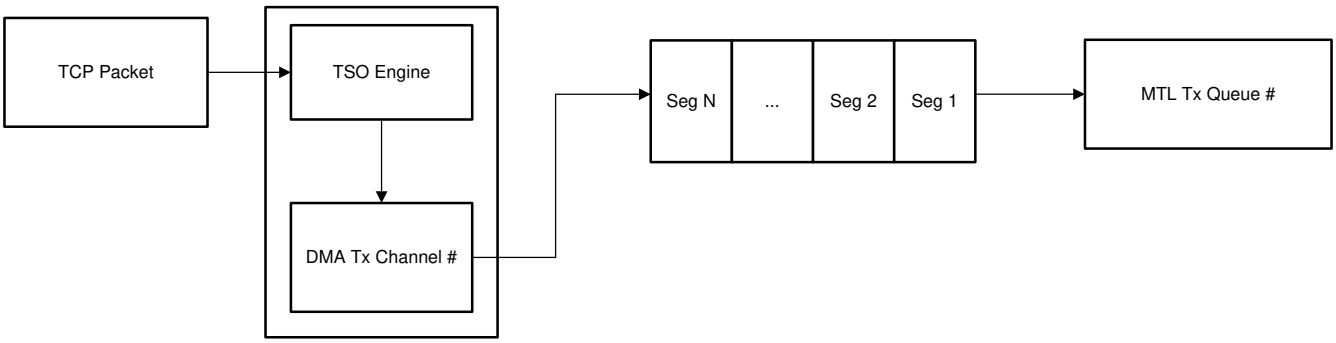


Figure 43-14. TCP Segmentation Overview

43.3.5.3.1 DMA Operation with TSO Feature

For the TSO feature, the Tx DMA operation is as follows:

1. The application sets up the Transmit descriptor (TDES0-TDES3) and sets the Own bit (TDES3[31]) after setting up the corresponding data buffer(s) with Ethernet Packet data.
2. The application increases the offset value of the Descriptor Tail pointer of the DMA Tx channel.
3. While in the Run state, the DMA fetches the next available descriptor and does one of the following:
   - If the descriptor is a context descriptor and the context is not between the first and last descriptors of a packet, the DMA stores the context values.
   - If the descriptor is a context descriptor and the context is between the first and last descriptors of a packet, the DMA closes the context descriptor indicating a Context Descriptor Error (TDES3[23]) and fetches the next descriptor.
   - If the descriptor is a normal descriptor, the DMA checks for the TSE bit. If TSE bit is not set, the DMA continues with the default mode of operation or OSF operation (if enabled).
4. The DMA calculates the number of segments from the TCP payload length(TDES3[17:0]) and the MSS value.
5. The DMA goes through channel arbitration to fetch the data buffers. The DMA fetches the header and payload separately.
6. For the first segment, the DMA fetches the header from the system memory and stores it in the TSO memory (if present and when the length of header is not greater than the TSO memory size). If the current segmented packet is not the first segment, the DMA fetches the header from the local TSO memory if available. Otherwise, it will fetch the header buffer in system memory again, as done for the first segment. In such cases (header not available is TSO memory), the DMA will not close the first descriptor containing the header buffer until the header for last segment is fetched.
7. The required fields in the header bytes are modified/updated as per the segmentation requirements and written into the corresponding MTL TxQ.
8. The DMA then takes the payload buffer pointer, fetches the MSS number of payload bytes from the system memory and directly pushes it into the MTL TxQ. In case the buffer(s) in the descriptor do not have enough data for the MSS payload (except for the last segment), the DMA closes this descriptor.
9. The DMA jumps to Step 3 and repeats the process until the last segment is written into the TxQ.
10. The DMA closes the last descriptor and also the first descriptor (containing the header buffer when it is not stored in TSO memory) and then moves on to the next packet transfer.

The DMA repeats all these steps if more descriptors are available. When no descriptor is available, the DMA enters the suspend state.

Note

The TSO engine determines whether to perform TSO or USO operation based on the THL field (TCP Header Length) in TDES3 of first Normal Tx descriptor for the packet. The value of 2 indicates USO and any value greater than or equal to 5 indicates TSO. The DMA repeats all these steps if more descriptors are available. When no descriptor is available, the DMA enters the suspend state.

43.3.5.3.1.1 TCP/IP Header Fields

While segmenting a TCP packet, the DMA automatically updates the TCP/IP header fields. Table 43-27 describes how the TCP and IP headers are updated.

Table 43-27. TSO: TCP and IP Header Fields

Packet Sequence	TCP Header	IP Header
First packet	1. The sequence number is not updated. The value provided in the header is used.	■ IPv4 Header ❑ Total Length = MSS + TCP Header Length + IP Header Length
	2. The TCP checksum is calculated again.	❑ Identification field is not modified.
	3. If set, the FIN and PSH flags are cleared.	It is sent as per the header provided by the software. ❑ IPv4 Header Checksum is recalculated. ■ IPv6 Header Payload Length = MSS + TCP Header Length + IPExtension Header Length
Subsequent packets	1. The sequence number is updated. The MSS value is added to the sequence number value of previous segment.	■ IPv4 Header ❑ Total Length = MSS + TCP Header Length + IP Header Length
	2. If set, the FIN and PSH flags are cleared.	❑ Identification field = Previous Identification Field + 1
	3. The TCP checksum is calculated again.	❑ IPv4 Header Checksum is recalculated ■ IPv6 Header Payload Length = MSS + TCP Header Length + IPExtension Header Length
Last packet	1. The sequence number is updated. The MSS value is added to the sequence number value of previous segment.	■ IPv4 Header ❑ Total Length = Remaining Payload + TCP Header Length + IP Header Length
	2. If FIN and PSH flags were set in original header, these flags are set.	❑ Identification Field = Previous Identification
	3. The TCP checksum is calculated again.	Field + 1 ❑ IPv4 header Checksum is recalculated ■ IPv6 Header Payload Length = Remaining Payload Length + TCP Header Length + IP Extension Header Length

43.3.5.3.1.2 Header and Payload Fields of Segmented Packets

After segmentation, the split packets use the same header as the parent TCP packet for header fields other than the ones described in Table 43-27. Figure 43-15 shows how same header is used for the header fields of segmented packets.

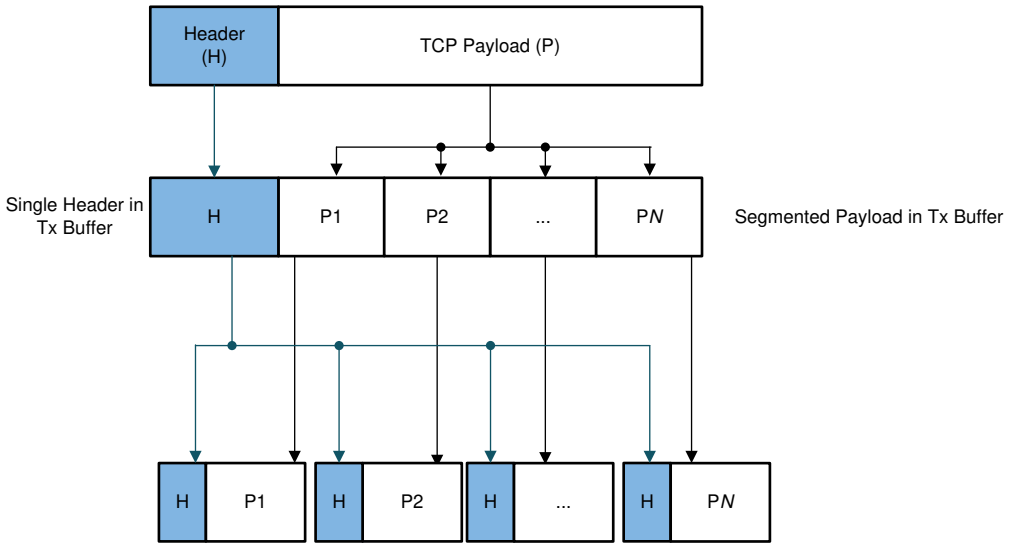


Figure 43-15. Header and Payload Fields of Segmented Packets

The application must create the header in Buffer 1 of the first descriptor of the packet to be segmented and provide the header length in TDES2 of the first descriptor (FD = 1). When the FD bit is set, the DMA reads the header from the header buffer to which the TDES0 is pointing. Buffer 2 of the first descriptor can be used for payload and TDES0 and TDES1 of subsequent descriptors. For subsequent descriptors (FD = 0), the address to which the TDES0 and TDES1 are pointing is treated as payload buffer address of the same packet.

Fragmentation is supported only for UDP over IP. Fragmentation is not supported for TCP packets.

43.3.5.4 Segmentation Versus Fragmentation

This table shows the differences between UDP packet fragmentation and UDP segmentation.

Table 43-28. Segmentation Versus Fragmentation

Segmentation	Fragmentation
The UDP header is replicated for each segment. The length field in the UDP header is updated for each segment.	The UDP header is not updated; however, checksum field in the UDP header is updated.
MSS field indicates the size of the maximum segment after the L4 (TCP/UDP) header.	MFS field indicates the size of the fragment after the L3 (IPv4) header.
The TCP packet is broken down into multiple chunks by keeping L2+L3+L4 header.	The UDP packet is broken down into multiple chunks by keeping the L2 + L3 header.

43.3.5.5 Using the IPv4 ARP Offload Engine

The Ethernet module supports the Address Recognition Protocol (ARP) Offload for IPv4 packets. This feature allows the processing of the IPv4 ARP request packet in the receive path and generating corresponding ARP response packet in the transmit path. Ethernet module generates the ARP reply packets for appropriate ARP request packets. The ARP packet for IPv4 is L2 layer packet with Length/Type of 0x0806.

The ARP offloading process is as follows:

1. The MAC receiver gets an ARP request if the Target Protocol Address of request matches the IPv4 address programmed in the L3 register of the MAC.
2. The MAC generates an ARP reply packet.
3. The MAC copies the Sender Hardware Address field in the ARP request to the following fields:
   - DA field of the Ethernet packet header
   - Target Hardware Address field of the ARP reply packet
4. The MAC copies the Sender Protocol Address field in the ARP request to the Target Protocol Address field in the ARP reply packet.
5. The MAC places its MAC address in the following fields:
   - SA field of the Ethernet packet header
   - Sender Hardware Address field of the ARP reply packet
6. The MAC copies the Target Protocol Address field in the ARP request to the Sender Protocol Address field in the ARP reply packet.
7. The MAC sets the opcode field in ARP reply packet to 2 indicating ARP reply.
8. The MAC recalculates the CRC and performs padding for generated ARP reply packet.
9. The MAC transmitter sends the ARP reply.

The MAC processes only one ARP request at a time. It does not store the fields of multiple ARP requests. If the MAC receives an ARP request when it is already processing an earlier ARP request, the MAC does not generate the ARP reply for new ARP request. The MAC forwards the new ARP request packet to application with ARP Reply Not Generated (Bit 34) status bit set. However, in power-down mode, if the MAC receives an ARP request when it is already processing an earlier ARP request, the MAC drops the new ARP request.

If the Disable CRC check bit of the MAC Extension Configuration bit is set, then the MAC does not check for valid CRC of a ARP request Packet. It can generate an ARP response packet if the other conditions are valid.

The ARP request Packet must always have a valid CRC.

Note

When the received ARP request is less than 64 bytes packet length, Ethernet module does not send a ARP response. It is treated as normal packet and forwarded to the application based on the Ethernet module filter settings.

43.3.5.6 Energy Efficient Ethernet Support

The Ethernet module supports the following techniques to save system power.

- The Ethernet module supports the following techniques to save system power.
- Remote Wakeup
- Energy Efficient Ethernet

The Magic Packet and Remote Wakeup techniques are used to save power consumption at the companion PHY. Using these features, the Ethernet module. On receiving the specific packets from the network, the MAC provides the trigger to restore the power and come back to normal state.

The Energy Efficient Ethernet (EEE) mode is compliant with the IEEE 802.3az-2010 standard. It is primarily targeted to save power in the Ethernet port when there is no traffic on the line. In this mode, the host indicates to the far-end that it does not have any packets to transmit for near future and puts the transmitter port

(MAC Controller, PCS and PHY layers) into low-power mode. Similarly, the Receiver port can also be put into low-power mode when the far-end host indicates that it does not have any traffic to transfer. This allows significant saving of power in the Ethernet port (mainly in the PHY) with intermittent and bursty traffic profile. The triggering of entry and exit out of the EEE mode is controlled by the MAC and is supported by the Ethernet module.

Simultaneous operation of the EEE mode along with any or both the other power saving modes is also supported in the Ethernet module.

43.3.5.6.1 Magic Packet

The magic packet contains a unique pattern at any offset after the Destination address, Source address, and Length/Type fields. In addition to the unique pattern matching, the MAC receiver also checks for the following, to detect the received packet as a valid magic packet:

- The packet must be addressed to it (Destination Address of the received packet should perfect match the MAC_Address0_High and MAC_Address0_Low registers) or with multicast/broadcast address
- The packet must not have length error, FCS error, dribble bit error, MII error, and collision
- The packet must not be runt (length including Ethernet header and FCS is at least 64 bytes)

The content of the unique pattern in magic packet is:

- 6 bytes of all-ones (48'hFF_FF_FF_FF_FF_FF) called the synchronization stream. There can be more than six bytes of 8'hFF, but last 6 are considered.
- The synchronization stream is immediately followed by 16 repetitions of Destination address field of the packet (MAC Address (MAC_Address0_High and MAC_Address0_Low registers) or multi- cast/broadcast address)
- No break or interruption between synchronization stream and first repetition of Destination address field or within its 16 repetitions

If the MAC address of a node is 48'h 00_11_22_33_44_55, the MAC scans for the following data sequence:

Destination Address Source Address Length/Type……. FF FF FF FF FF FF00 11 22

33 44 55 00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 5500 11 22 33 44 55

00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 5500 11 22 33 44 55 00 11 22

33 44 55 00 11 22 33 44 55 00 11 22 33 44 5500 11 22 33 44 55 00 11 22 33 44 55

00 11 22 33 44 55 00 11 22 33 44 55…CRC

43.3.5.6.2 Remote Wakeup Filter

In the Remote Wakeup Magic Packet based power saving mode, the reception of expected remote wakeup packet by MAC receiver triggers the exit from low-power mode. The MAC enters power saving mode when PWRDWN bit of MAC_PMT_Control_Status register is programmed to 1. Exit from the remote wakeup based low-power mode is enabled by programming RWKPKTEN bit of MAC_PMT_Control_Status register to 1.

The MAC implements a filter lookup table (programmed through MAC_RWK_Packet_Filter register) in which CRC, offset, and byte mask of the pattern embedded in remote wakeup packet and the filter operation commands are programmed.

The pattern embedded in the remote wakeup packet is located at any offset after the Destination address and Source address fields. In addition to the CRC match for the pattern, the MAC receiver also checks the following, to detect the received packet as a valid remote wakeup packet:

- The packet must be addressed to it (Destination Address of the received packet should perfect match the MAC_Address0_High and MAC_Address0_Low registers) or with multicast/broadcast address
- The packet must not have length error, FCS error, dribble bit error, MII error, and collision
- The packet must not be runt (length including Ethernet header and FCS is at least 64 bytes)

When a valid remote wakeup packet is received, the MAC receiver sets the RWKPRCVD bit in MAC_PMT_Control_Status register and triggers the PMT interrupt . The PMTIS bit in MAC_Interrupt_Status

register is set when power-gating is not enabled in low-power mode. An interrupt is triggered to the application (sbd_intr) when interrupt is enabled (PMTIE bit in MAC_Interrupt_Enable register is set).

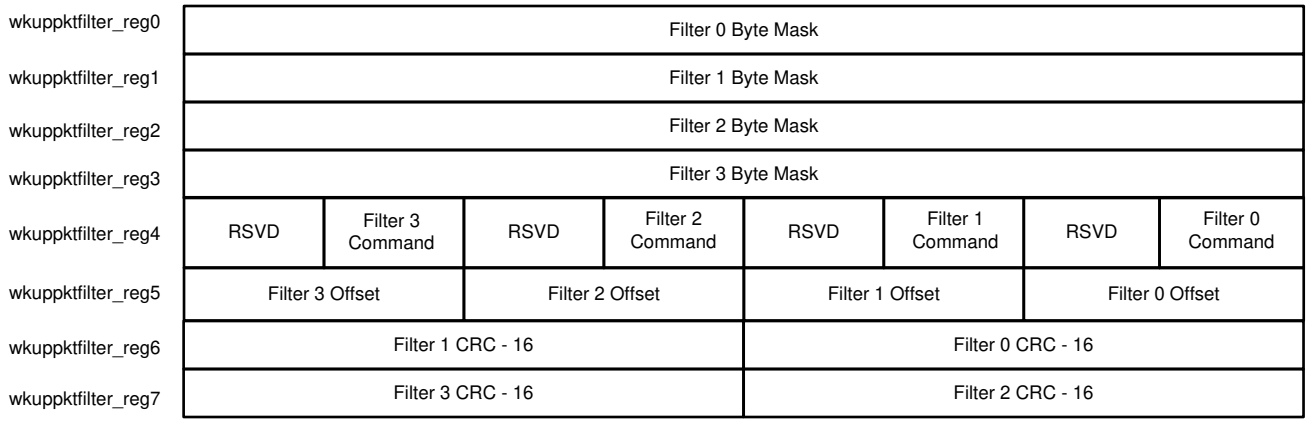


Figure 43-16. Wakeup Filter Register Layout

The Remote Wakeup Filters are arranged in blocks of 4 filters each and each such block have eight 32-bit wide registers, viz. wkuppktfilter_reg0-7, wkuppktfilter_reg8-15, wkuppktfilter_reg16-23 and wkuppktfilter_reg24-31. The fields of Remote Wakeup Filter are as follows:

Filter i Byte Mask

The filter i byte mask register defines the bytes of the packet that are examined by filter i (0, 1, 2, 3, .., 15) to determine whether or not a packet is a wake-up packet.

- The MSB (31st bit) must be zero.
- Bit j[30:0] is the byte mask.
- If Bit j (byte number) of the byte mask is set, the CRC block processes the Filter i Offset + j of the incoming packet; otherwise Filter i Offset + j is ignored

Filter i Command

The 4-bit filter i command controls the filter i operation.

- Bit 3 specifies the address type, defining the destination address type of the pattern. When the bit is set, the pattern applies to only multicast packets; when the bit is reset, the pattern applies only to unicast packet
- Bit 2 (Inverse Mode), when set, reverses the logic of the CRC16 hash function signal, to reject a packet with matching CRC-16 value. Bit 2, along with Bit 1, allows a MAC to reject a subset of remote wake-up packets by creating filter logic such as "Pattern 1 AND NOT Pattern 2".
- Bit 1 (And_Previous) implements the Boolean logic. When set, the result of the current entry is logi- cally ANDed with the result of the previous filter. This AND logic allows a filter pattern longer than 32 bytes by splitting the mask among two, three, or four filters. This depends on the number of filters that have the And_Previous bit set. The details are as follows:
  - The And_Previous bit setting is applicable within a set of 4 filters.
  - ❑Setting of And_Previous bit of filter that is not enabled has no effect, that is setting And_Previous bit of lowest number filter in the set of 4 filters has no effect. For example, setting of And_Previous bit of Filter 0 has no effect.
  - If And_Previous bit is set for filter to form AND chained filter, the AND chain breaks at the point any filter is not enabled. For example: If Filter 2 And_Previous bit is set (bit 1 in Filter 2 command is set) but Filter 1 is not enabled (bit 0 in Filter 1 command is reset), then only Filter 2 result is considered. If Filter 2 And_Previous bit is set (bit 1 in Filter 2 command is set), Filter 3 And_Pre- vious bit is set (bit 1 in Filter 3 command is set), but Filter 1 is not enabled (bit 0 in Filter 1 command is reset), then only Filter 2 result ANDed with Filter 3 result is considered. If Filter 2 And_Previous bit is set (bit 1 in Filter 2 command is set), Filter 3 And_Previous bit is set (bit 1 in Filter 3 command is set), but Filter 2 is not enabled (bit 0 in Filter 2 command is reset), then since setting of Filter 2 And_Previous bit has no effect only Filter 1 result ORed with Filter 3 result is considered.

– If filters chained by And_Previous bit setting have complementary programming, then a frame may never pass the AND chained filter. For example, if Filter 2 And_Previous bit is set (bit 1 in Filter 2 command is set), Filter 1 Address_Type bit is set (bit 3 in Filter 1 command is set) indicating multicast detection and Filter 2 Address_Type bit is reset (bit 3 in Filter 2 command is reset) indi- cating unicast detection or vice versa, a remote wakeup frame does not pass the AND chained filter as a remote wakeup frame cannot be of both unicast and multicast address type.

Filter i Offset

The filter i offset register defines the offset (within the packet) from which the filter i examines the packets.

- This 8-bit pattern-offset is the offset for the filter i first byte to be examined.
- The minimum allowed offset is 12, which refers to the 13th byte of the packet.
- The offset value 0 refers to the first byte of the packet.

Filter i CRC-16

The filter i CRC-16 register contains the CRC-16 value calculated from the pattern and the byte mask programmed in the Remote Wakeup filter register.

- The 16-bit CRC calculation uses the following polynomial: G(x) = x^16 + x^15 + x^2 + 1
- Each mask, used in the hash function calculation, is compared with a 16-bit value associated with that mask. Each filter has the following:
  – 32-bit Mask: Each bit in this mask corresponds to one byte in the detected packet. If the bit is 1, the corresponding byte is taken into the CRC-16 calculation.
  – ❑ 8-bit Offset Pointer: Specifies the byte to start the CRC-16 computation. The pointer and the mask are used together to locate the bytes to be used in the CRC-16 calculations.

The Remote Wakeup Filter registers are implemented as 8 indirect access registers (wkuppktfilter_reg#i) and accessed by application through MAC_RWK_Packet_Filter register. When the Remote Wakeup Filters are to be programmed, the entire set of wkuppktfilter_reg registers must be written. The wkuppktfilter_reg register is programmed by sequentially writing the eight, sixteen or thirty-two register values in MAC_RWK_Packet_Filter register for wkuppktfilter_reg0, wkuppktfilter_reg1, ..., wkuppktfilter_reg31 respectively. The wkuppktfilter_reg register is read in a similar way. The MAC updates the wkuppktfilter_reg register current pointer value in RWKPTR field of MAC_PMT_Control_Status register.

Note: When MAC_RWK_Packet_Filter register is written, the content is transferred from CSR clock domain to PHY receive clock domain after the write operation, there should not be any further write to the MAC_RWK_Packet_Filter register until the first write is updated in PHY receive clock domain. Otherwise, the second write operation does not get updated to the PHY receive clock domain. Therefore, the delay between two writes to the MAC_RWK_Packet_Filter register should be at least 4 cycles of the PHY receive clock.

The PMT interrupt signal is asserted when a valid remote wake-up packet is received. As the PMT interrupt signal is generated in the PHY Rx clock domain, it is not cleared immediately when the PMT Control and Status register is read. This is because the resultant clear signal has to cross to the PHY Rx clock domain, and then clear the interrupt source. This delay is at least 4 clock cycles of Rx clock and can be significant when the Ethernet module is operating in the 10 Mbps mode. When the application clears the PWRDWN bit in Remote Wake-Up Packet Detection register, the MAC comes out of the power-down mode, but this event does not generate the PMT interrupt.

The MAC implements a set of registers for Layer3 and Layer4 based packet filtering. In a register set, there is a control register, such as MAC_L3_L4_Control (#i) (for i = 0; i <=3), to control the packet filtering. In addition, there are five address registers to program the Layer 3 and Layer 4 fields to be matched, such as:

- MAC_Layer4_Address(#i) (for i = 0; <=3)
- MAC_Layer3_Address0_Reg(#i) (for i = 0; i <=3)
- MAC_Layer3_Address1_Reg(#i) (for i = 0; i =3)
- MAC_Layer3_Address2_Reg(#i) (for i = 0; i <=3)
- MAC_Layer3_Address3_Reg(#i) (for i = 0; i =<3)

43.3.5.6.3 Energy Efficient Ethernet

EEE is an operational mode that enables the IEEE 802.3 Media Access Control (MAC) sub layer along with a family of physical layers to operate in the Low-Power Idle (LPI) mode. The EEE operational mode supports the IEEE 802.3 MAC operation at 100 Mbps. The Ethernet module supports the IEEE 802.3az-2010 for EEE.

The LPI mode allows power saving by switching off the parts of the communication device functionality when there is no data to be transmitted and received. The systems on both sides of the link can disable some functionalities to save power during the periods of low-link utilization. The MAC controls whether the system should enter or exit the LPI mode and communicates this to the PHY.

The EEE specifies the capabilities negotiation methods that the link partners can use to determine whether EEE is supported, and then select the set of parameters that are common to both devices.

43.3.5.6.3.1 Transmit Path Functions

The transmit path functions include tasks that the MAC must perform to make the PHY to enter the LPI state. In the transmit path, the software must set the LPIEN bit of the MAC_LPI_Control_Status register to indicate to the MAC to stop transmission and initiate the LPI protocol. The MAC completes the transmission in progress, generates its transmission status, and starts transmitting the LPI pattern instead of the IDLE pattern if the link status has been up continuously for a period specified in the LPI LS TIMER field of MAC_LPI_Control_Status register. The PHY Link Status bit of the LPI Control and Status Register indicates the link status of the PHY.

To make the PHY enter the LPI state, the MAC performs the following tasks:

1. De-asserts TX_EN
2. Asserts TX_ER
3. Sets TXD[3:0] to 0x1 (for 100 Mbps) or TXD[7:0] to 0x01 (for 1,000 Mbps)
4. Updates the status (TLPIEN bit of MAC_LPI_Control_Status register) and generates an interrupt

Note

The MAC maintains the same state of the TX_EN, TX_ER, and TXD signals for the entire duration during which the PHY remains in the LPI state.

To bring the PHY out of the LPI state, that is, when the software resets the LPIEN bit, the MAC performs the following tasks:

1. Stops transmitting the LPI pattern and starts transmitting the IDLE pattern.
2. Starts the LPI TW TIMER The MAC cannot start the transmission until the wake-up time specified for the PHY expires. The auto-negotiated wake-up interval is programmed in the TWT field of the MAC_LPI_Timers_Control register.
3. Updates the LPI exit status (TLPIEX bit of the MAC_LPI_Control_Status register) and generates an interrupt.

Figure 43-17 shows the behavior of TX_EN, TX_ER, and TXD[3:0] signals during the LPI mode transitions.

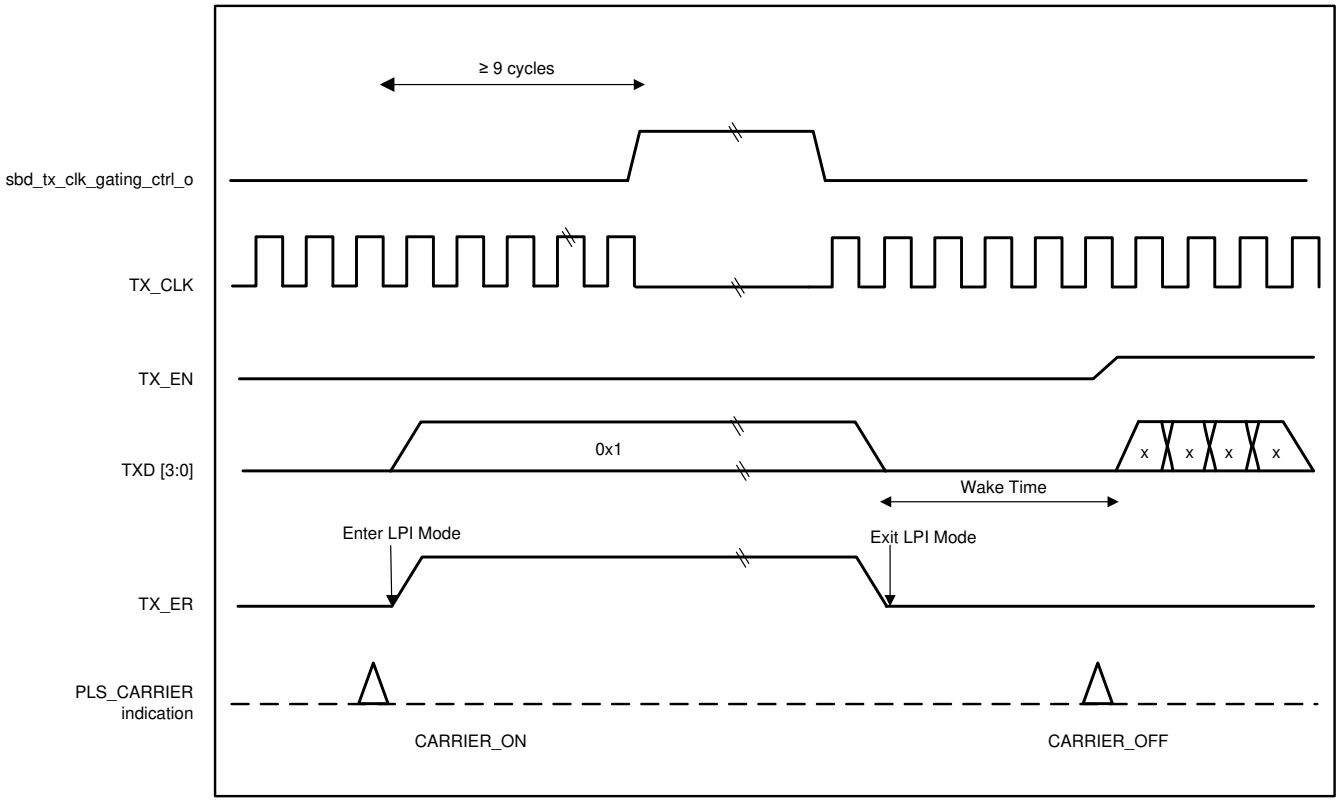


Figure 43-17. LPI Transitions on Transmit

Note

The MAC does not stop the TX_CLK clock. It is stopped at SoC level (as shown in Figure 43-17) if your PHY supports it and when the MAC sets the sbd_tx_clk_gating_ctrl_o signal to 1. The assertion of the sbd_tx_clk_gating_ctrl_o signal is dependent on the LPITCSE bit of the MAC_LPI_Control_Status register.

43.3.5.6.4 Automated Entry/Exit of LPI mode in Transmit Path

The MAC transmitter can be programmed to enter and exit LPI IDLE mode automatically based on whether it is IDLE for a specific period of time or has a packet to transfer. These modes are enabled and controlled by MAC_LPI_Control_Status register.

When LPITXA (Bit[19]) and LPITXEN (Bit[16]) of MAC_LPI_Control_Status register are set, the MAC transmitter enters LPI IDLE state when the MAC transmit path (including the MTL layers and DMA layers) are idle. The MAC transmitter will exit the LPI IDLE state and clear the LPITXEN bit as soon as any of functions in the TX path (DMA, MTL or MAC) becomes non-idle due to initiation of a packet transfer.

In addition to the above, when Bit[20] (LPIATE) is also set, the MAC transmitter will enter LPI IDLE state only if the Transmit path remains in idle state (no activity) for the time period indicated by the value in MAC_LPI_Entry_Timer. In this mode also, the MAC transmitter will exit the LPI IDLE state as soon as any of the functions becomes non-idle. However, the LPITXEN bit is not cleared but remains active so that reentry to LPI IDLE state is possible without any software intervention when the MAC becomes idle again.

When both LPIATE and LPITXA bits are cleared, you can directly control the entry and exit of LPI IDLE state by programming the LPITXEN bit.

43.3.5.6.5 Receive Path Functions

The receive path functions include the tasks that the PHY and MAC must perform when the PHY receives signals from the link partner to exit the LPI state.

In the receive path, when the PHY receives the signals from the link partner to enter into the LPI state, the PHY and MAC perform the following tasks:

1. The PHY asserts RX_ER .
2. The PHY sets RXD[7:0] to 0x01.
3. The PHY de-asserts RX_DV.
4. The MAC updates the RLPIEN bit of the MAC_LPI_Control_Status register and immediately generates an interrupt.

Note

The MAC does not stop the TX_CLK clock. It is stopped at SoC level (as shown in Figure 43-17) if your PHY supports it and when the MAC sets the sbd_tx_clk_gating_ctrl_o signal to 1. The assertion of the sbd_tx_clk_gating_ctrl_o signal is dependent on the LPITCSE bit of the MAC_LPI_Control_Status register.

43.3.5.7 Automated Entry/Exit of LPI Mode in Transmit Path

The MAC transmitter can be programmed to enter and exit LPI IDLE mode automatically based on whether it is IDLE for a specific period of time or has a packet to transfer. These modes are enabled and controlled by MAC_LPI_Control_Status register.

When LPITXA (Bit[19]) and LPITXEN (Bit[16]) of MAC_LPI_Control_Status register are set, the MAC transmitter enters LPI IDLE state when the MAC transmit path (including the MTL layers and DMA layers) are idle. The MAC transmitter will exit the LPI IDLE state and clear the LPITXEN bit as soon as any of functions in the TX path (DMA, MTL or MAC) becomes non-idle due to initiation of a packet transfer.

In addition to the above, when Bit[20] (LPIATE) is also set, the MAC transmitter will enter LPI IDLE state only if the Transmit path remains in idle state (no activity) for the time period indicated by the value in MAC_LPI_Entry_Timer. In this mode also, the MAC transmitter will exit the LPI IDLE state as soon as any of the functions becomes non-idle. However, the LPITXEN bit is not cleared but remains active so that reentry to LPI IDLE state is possible without any software intervention when the MAC becomes idle again.

When both LPIATE and LPITXA bits are cleared, you can directly control the entry and exit of LPI IDLE state by programming the LPITXEN bit.

43.3.5.8 Receive Path Functions

The receive path functions include the tasks that the PHY and MAC must perform when the PHY receives signals from the link partner to exit the LPI state.

In the receive path, when the PHY receives the signals from the link partner to enter into the LPI state, the PHY and MAC perform the following tasks:

1. The PHY asserts RX_ER
2. The PHY sets RXD[7:0] to 0x01
3. The PHY de-asserts RX_DV
4. The MAC updates the RLPIEN bit of the MAC_LPI_Control_Status register and immediately generates an interrupt
5. The PHY maintains the same state of the RX_ER, RXD, and RX_DV signals for the entire duration during which it remains in the LPI state.
   - If the LPI pattern is detected for a very short duration (that is, less than two cycles of Rx clock), the MAC does not enter the Rx LPI mode.
   - If the duration between end of the current Rx LPI pattern and start of the next Rx LPI pattern, is very short (that is, less than two cycles of Rx clock), then the MAC exits and again enters the Rx LPI mode. The MAC does not give the Rx LPI Exit and Entry interrupts

When the PHY receives signals from the link partner to exit the LPI state, the PHY and MAC perform the following tasks:

1. The PHY de-asserts RX_ER and returns to a normal inter-packet state.
2. The MAC updates the RLPIEX bit of the MAC_LPI_Control_Status register and generates an interrupt immediately. The sideband signal lpi_intr_o (synchronous to Rx clock) is also asserted.

Figure 43-18 shows the behavior of RX_ER, RX_DV, and RXD[3:0] signals during the LPI mode transitions.

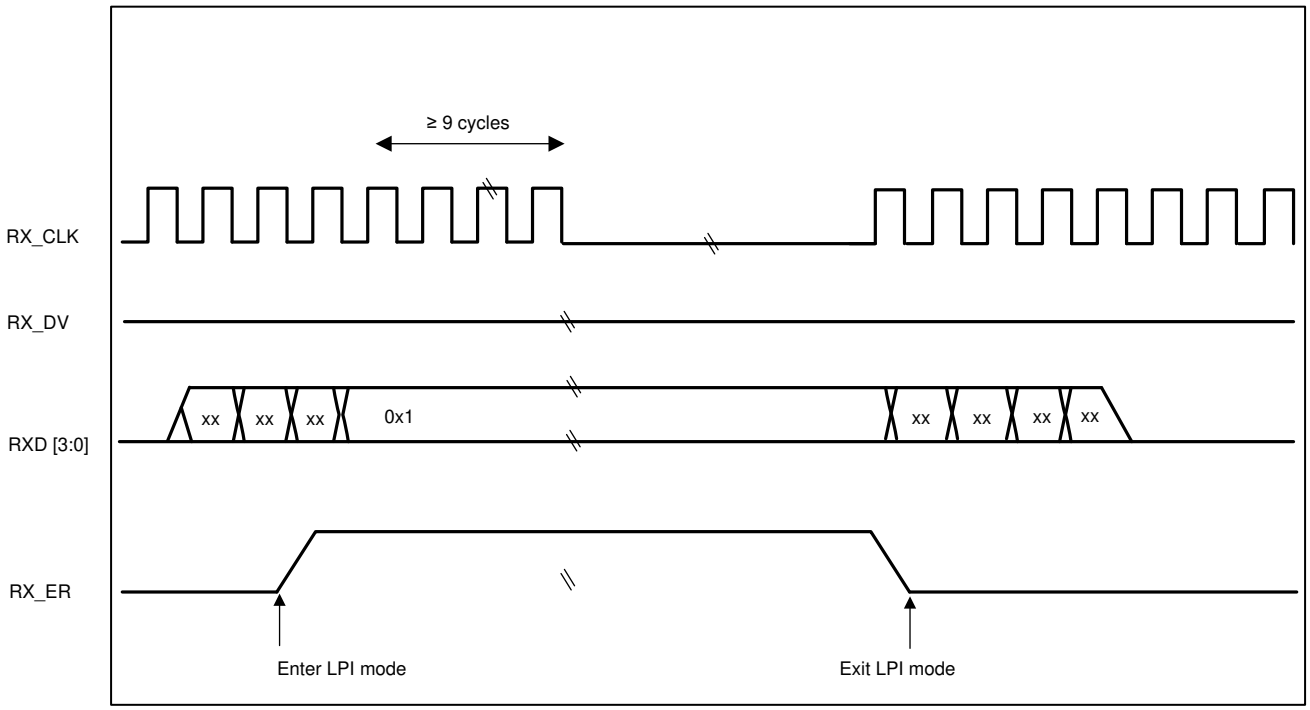


Figure 43-18. LPI Transitions on Receive

43.3.6 Loopback Mode

The MAC supports Loopback of transmitted packets to its receiver. The following are some guidelines for using the loopback mode:

- Enable loopback only with the full-duplex mode. In half-duplex mode, the carrier sense signal (CRS) or collision (COL) signal inputs get sampled which may result into issues such as packet dropping.
- If the loopback mode is enabled without connecting a PHY chip (for example, in FPGA setup), you should enable internal clock for loopback to generate the Tx and Rx clocks and provide these clocks to the MAC.
- Program the CLK_LM field of EMACSS_CTRLSTS Register to program the loopback clocks (either from the external source or from the internal source).
  - If internal source is selected the divider ETHDIV and clocksource ETHDIVSRCSEL should be properly set to derive 50 Mhz clock as in the case of RMII clocking.
  - If External source is selected :
    - For RMII mode the ENET_RMII_CLK should be provided with 50 Mhz clock
    - For MII mode the ENET_MII_TX_CLK and ENER_MII_RX_CLK should be provided with 25 Mhz(for 100 mbps) or 2.5 Mhz (for 10 mbps link)
- Do not loop back big packets. Big packets may get corrupted in the loopback FIFO.

At the end of every received packet, the Receive Protocol Engine module generates received packet status and sends it to the Receive Packet Controller module. The control, missed packet, and filter fail status are added to the Receive status in the Receive Packet Controller module.

The MAC does not process ARP or PMT packets that are looped back.

43.3.7 Reverse Media Independent Interface (RevMII)

This section gives the RevMII register blocks. For detailed flow on using RevMII, refer to the RevMII Application note.

43.3.7.1 RevMII Register Maps

Table 43-29 provides the address map and high-level summary of the RevMII registers for MAC.

Table 43-29. RevMII Register Maps - MAC

Register Number	Address Offset	Register Name and Description
0	5'b00000	MAC_RevMII_PHY_Control Controls the MAC side of the RevMII Controller.
1	5'b00001	MAC_RevMII_Common_Status Shows the status of the RevMII. This is a shared register for both MACs.
2–14	5'b00010–5'b01110	Reserved
15	5'b01111	MAC_RevMII_Common_Ext_Status Shows the status of the RevMII supporting 1000 Mbps speed. This is a common register for both MACs.
16	5'b10000	MAC_RevMII_Interrupt_Status_Mask Shows the status of the interrupt generated for the MAC and also provides interrupt masking signal for the generated interrupts.
17	5'b10001	MAC_RevMII_Remote_PHY_Status Shows the status of speed and duplex-mode programmed in the remote PHY Control register.
18–31	5'b10010–5'b11111	Reserved

Table 43-30 provides the address map and high-level summary of the RevMII registers of the remote MAC.

Table 43-30. RevMII Register Map - Remote MAC

Register Number	Address Offset	Register Name and Description
0	5'b00000	MAC_RevMII_RemotePHY_Control Controls the remote side of RevMII Controller.This register is similar to the MAC_RevMII_PHY_Control register.
1	5'b00001	MAC_RevMII_Common_Status Shows the status of the RevMII. This is a common register for both MACs.
2–14	5'b00010–5'b01110	Reserved
15	5'b01111	MAC_RevMII_Common_Ext_Status Shows the status of the RevMII supporting 1000 Mbps speed. This is a common register for both MACs.
16	5'b10000	MAC_RevMII_RemotePHY_Interrupt_Status_Mask Shows the status of the interrupt generated for remote MAC and also provides interrupt masking signal for the generated interrupts. This register is similar to MAC_RevMII_Interrupt_Status_Mask.
17	5'b10001	MAC_RevMII_PHY_Status Register Shows the status of speed and duplex-mode programmed in the RevMII PHY Control register.
18-31	5'b10010–5'b11111	Reserved

43.3.7.2 MAC_RevMII_PHY_Control

This register controls the RevMII operations for the MAC and enables the RevMII modes.

Table 43-31. MAC_RevMII_PHY_Control Register

15	14	13	12	11	10	9	8	7	6	5:0
REVRST	REVLPBCK	REVSSL	REVANEN	REVPWRDN	REVISOL	REVREAN	REVDM	REV COLTST	REVSSH	Rsvd

Table 43-32. MAC_RevMII_PHY_Control Register Description

Field	Name	Description	Reset	Access
15	REVRST	Reset When this bit is set, it configures the PHY Control register to its default values. This bit is cleared after the reset operation is complete.	0	R_W_SC
14	REVLPBCK	Loopback When this bit is set, it enables the loopback mode. When this bit is reset, it disables the loopback mode.	0	R/W
13	REVSSL	Speed Selection (LSB) This bit along with Bit 6 (MSB) indicates the link speed. Bit 6 | Bit 13 | Speed 1 | 1 | Reserved 1 | 0 | Reserved 0 | 1 | 100 Mbps 0 | 0 | 10 Mbps When you select 10/100 Mbps as the Mode of Operation, the reset value of this bit is 1.	0	R/W
12	REVANEN	Auto-Negotiation Enable This bit is not used in RevMII.	0	RO
11	REVPWRDN	Power Down When this bit is set, it enables the power down mode. When this bit is reset, it disables the power-down mode.	0	R/W
10	REVISOL	Isolate When this bit is set, it enables the isolate mode. When this bit is reset, it disables the isolate mode.	0	R/W
9	REVREAN	Restart Auto-Negotiation This bit is not used in RevMII.	0	RO
8	REVDM	Duplex Mode When this bit is set, it configures the RevMII PHY for the full-duplex operation. When this bit is reset, it configures the RevMII PHY for the half-duplex operation. When you select the Disable Half-Duplex Operation option, the reset value of this bit is 1.	0	R/W
7	REVCOLTST	Collision Test When this bit is set, it enables the collision test mode. When this bit is reset, it disables the collision test mode.	0	R/W
6	REVSSH	Speed Selection (MSB) When set, this bit along with Bit 6 (LSB) indicates the link speed. For more information, see REVSSL.When you select 10/100 Mbps as the Mode of Operation, the reset value of this bit is 0.	1	R/W
5-0	Rsvd	Reserved	0	RO

43.3.7.3 MAC_RevMII_Common_Status

This register provides the RevMII status. This register is common for the MAC and the remote MAC.

Table 43-33. MAC_RevMII_Common_Status Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
100T4	100XFD	100XHD	10FD	10HD	100T2FD	100T2HD	EXTSTS	Rsvd	PRESUP	ANC	RMTFLT	ANA	LNKSTS	JABDET	EXTCAP

Table 43-34. MAC_RevMII_Common_Status Register Description

Field	Name	Description	Reset	Access
15	100T4	100BASE-T4 When this bit is set, it indicates that the RevMII PHY can perform the link transmission and reception using the 100BASE-T4 signaling specification.	1	RO
14	100XFD	100BASE-X Full-Duplex When this bit is set, it indicates that the RevMII PHY can perform the full-duplex link transmission and reception using the 100BASE-X signaling specification.	1	RO
13	100XHD	100BASE-X Half-Duplex When this bit is set, it indicates that the RevMII PHY can perform the half-duplex link transmission and reception using the 100BASE-X signaling specification. When you select the Disable Half-Duplex Operation option, the reset value of this bit is 0.	1	RO
12	10FD	10 Mbps Full-Duplex When this bit is set, it indicates that the RevMII PHY can perform the full-duplex link transmission and reception while operating in 10 Mbps mode.	1	RO
11	10HD	10 Mbps Half-Duplex When this bit is set, it indicates that the RevMII PHY can perform the half-duplex link transmission and reception while operating in 10 Mbps mode. When you select the Disable Half-Duplex Operation option, the reset value of this bit is 0.	1	RO
10	100T2FD	100BASE-T2 Full-Duplex When this bit is set, it indicates that the RevMII PHY can perform full-duplex link transmission and reception using the 100BASE-T2 signaling specification.	1	RO
9	100T2HD	100BASE-T2 Half-Duplex When this bit is set, it indicates that the RevMII PHY can perform the half-duplex link transmission and reception using the 100BASE-T2 signaling specification. When you select the Disable Half-Duplex Operation option, the reset value of this bit is 0.	1	RO
8	EXTSTS	Reserved	1	RO
7	Rsvd	Reserved	0	RO
6	PRESUP	MF Preamble Suppression This bit indicates that the RevMII can receive management frames irrespective of the Preamble length.	1	RO
5	ANC	Auto-Negotiation Complete This bit is not used in RevMII.	0	RO
4	RMTFLT	Remote Fault This bit is not used in RevMII.	0	RO
3	ANA	Auto-Negotiation Ability This bit is not used in RevMII.	0	RO

Table 43-34. MAC_RevMII_Common_Status Register Description (continued)

Field	Name	Description	Reset	Access
2	LNKSTS	Link Status When this bit is set, it indicates that a valid link has been established. When this bit is reset, it indicates that the link is not valid.	0	R_SS_ SC_LL O
1	JABDET	Jabber Detect This bit is not used in RevMII.	0	RO
0	EXTCAP	Extended Capability This bit is always set because RevMII supports extended register capability.	1	RO

43.3.7.4 MAC_RevMII_Common_Ext_Status

This register is common for the MAC and the remote MAC. This register is implemented for RevMII supporting 1000 Mbps speed. It is not present since MAC supports only 10/100 Mbps operations.

Table 43-35. MAC_RevMII_Common_Ext_Status Register

15	14	13	12	11:0
1000XFD	1000XHD	1000TFD	1000THD	Rsvd

Table 43-36. MAC_RevMII_Common_Ext_Status Register Description

Field	Name	Description	Reset	Access
15	1000XFD	1000BASE-X Full-Duplex When set, this bit indicates that the RevMII PHY can perform the full-duplex link transmission and reception using the 1000BASE-X signaling specification. When you select 10/100 Mbps as the Mode of Operation, the reset value of this bit is 0.	1	RO
14	1000XHD	1000BASE-X Half-Duplex When set, this bit indicates that the RevMII PHY can perform the half-duplex link transmission and reception using the 1000BASE-X signaling specification. When you select either the Disable Half-Duplex Operation option or 10/100 Mbps as the Mode of Operation, the reset value of this bit is 0.	1	RO
13	1000TFD	1000BASE-T Full-Duplex When set, this bit indicates that the RevMII PHY can perform the full-duplex link transmission and reception using the 1000BASE-T signaling specification. When you select 10/100 Mbps as the Mode of Operation, the reset value of this bit is 0.	1	RO
12	1000THD	1000BASE-T Half Duplex When set, this bit indicates that the RevMII PHY can perform the half-duplex link transmission and reception using the 1000BASE-T signaling specification. When you select either the Disable Half-Duplex Operation option or 10/100 Mbps as the Mode of Operation, the reset value of this bit is 0.	1	RO
11-0	Rsvd	Reserved	0	RO

43.3.7.5 MAC_RevMII_Interrupt_Status_Mask

This register provides the status of the interrupts and enables you to mask the interrupt signal. The status bits are cleared when this register is read.

Table 43-37. MAC_RevMII_Interrupt_Status_Mask Register

15:9	8	7:1	0
Rsvd	LSI	Rsvd	LSIM

Table 43-38. MAC_RevMII_Interrupt_Status_Mask Register Description

Field	Name	Description	Reset	Access
15-9	Rsvd	Reserved	0	RO
8	LSI	Link Status Change Interrupt When this bit is set, it indicates that the link status has changed. This bit is cleared on a read (or this bit is written to 1 when RWCE bit of MAC_CSR_SW_Ctrl register is set).	0	R_SS_R C_W1C
7-1	Rsvd	Reserved	0	RO
0	LSIM	Link Status Change Interrupt Mask When this bit is set, it disables the assertion of the interrupt signal because of the setting of the LSI bit.	0	R/W

43.3.7.6 MAC_RevMII_Remote_PHY_Status

This register shows the status of speed and duplex-mode programmed in the remote PHY Control register. You can use this register for MAC to MAC handshake when the link between the MAC and remote MAC is down because of the speed or duplex-mode mismatch.

Table 43-39. MAC_RevMII_Remote_PHY_Status Register

15:3	2	1	0
Rsvd	RMACDM	RMACSSH	RMACSSL

Table 43-40. MAC_RevMII_Remote_PHY_Status Register Description

Field	Name	Description	Reset	Access
15-3	Rsvd	Reserved	0	RO
2	RMACDM	Remote MAC Duplex Mode When this bit is set, it indicates the duplex mode configured in Bit 8 of the MAC_RevMII_RemotePHY_Control register. When you select the Disable Half-Duplex Operation option, the reset value of this bit is 1.	0	RO
1	RMACSSH	Remote MAC Speed Select MSB When this bit is set, it indicates the link speed specified in Bit 6 of the MAC_RevMII_RemotePHY_Control register. When you select 10/100 Mbps as the Mode of Operation, the reset value of this bit is 0.	1	RO
0	RMACSSL	Remote MAC Speed Select LSB When this bit is set, this bit indicates the link speed specified in Bit 13 of the MAC_RevMII_RemotePHY_Control register. When you select 10/100 Mbps as the Mode of Operation, the reset value of this bit is 1.	0	RO

43.3.7.7 MAC_RevMII_PHY_Status Register

Table 43-41 describes the RevMII PHY Status Register of a remote MAC.

Table 43-41. MAC_RevMII_PHY_Status Register

15:3	2	1	0
Rsvd	MACDM	MACSSH	MACSSL

Table 43-42. MAC_RevMII_PHY_Status Register Description

Field	Name	Description	Reset	Access
15-3	Rsvd	Reserved	0	RO
2	MACDM	MAC Duplex Mode When set, this bit indicates the duplex mode configured in Bit 8 of the MAC_RevMII_PHY_Control register. When you select the Disable Half-Duplex Operation option, the reset value of this bit is 1.	0	RO
1	MACSSH	MAC Speed Select MSB When set, this bit indicates the link speed specified in Bit 6 of the MAC_RevMII_PHY_Control register. When you select 10/100 Mbps as the Mode of Operation, the reset value of this bit is 0.	1	RO
0	MACSSL	MAC Speed Select LSB When set, this bit indicates the link speed specified in Bit 13 of the MAC_RevMII_PHY_Control register. When you select 10/100 Mbps as the Mode of Operation, the reset value of this bit is 1.	0	RO

43.4 Descriptors

The DMA in the Ethernet subsystem transfers data based on a linked list of descriptors. The application creates the descriptors in the system memory. The Ethernet module supports the following two types of descriptors:

- **Normal Descriptor:** Normal descriptors are used for packet data and to provide control information applicable to the packets to be transmitted.
- **Context Descriptor:** Context descriptors are used to provide control information applicable to the packet to be transmitted.

Each normal descriptor contains two buffers and two address pointers. These buffers enable the adapter port to be compatible with various types of memory management schemes.

Note

There is no limit for the number of descriptors that can be used for a single packet.

43.4.1 Descriptor Structure

In Ring structure (Figure 43-19), descriptors are separated by the Word, DWord, or LWord number programmed in the DSL field of the DMA_CH#_Control register. The application needs to program the total ring length, that is, the total number of descriptors in ring span in the following registers of a DMA channel:

- Transmit Descriptor Ring Length Register (DMA_CH#_TxDesc_Ring_Length)
- Receive Descriptor Ring Length Register (DMA_CH#_RxDesc_Ring_Length)

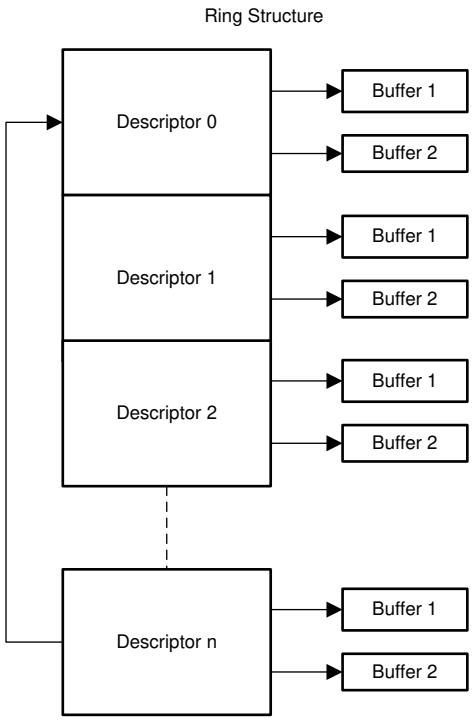


Figure 43-19. Descriptor Ring Structure

The Descriptor Tail Pointer Register contains the pointer to the descriptor address (N). The base address and the current descriptor pointer decide the address of the current descriptor that the DMA can process. The descriptors up to one location less than the one indicated by the descriptor tail pointer (N – 1) are owned by the DMA. The DMA continues to process the descriptors until the following condition occurs:

```
Current Descriptor Pointer == Descriptor Tail Pointer;
```

The DMA goes into the Suspend mode when this condition occurs. The application must perform a write to the Descriptor Tail pointer register and update the tail pointer so that the following condition is true:

```
Current Descriptor Pointer == Descriptor Tail Pointer;
```

The DMA automatically wraps around the base address when the end of ring is reached, as shown in Figure 43-20

For descriptors owned by the application, the OWN bit of DES3 is reset to 0. For descriptors owned by the DMA, the OWN bit is set to 1. If the application has only one descriptor in the beginning, the application sets the last descriptor address (tail pointer) to Descriptor Base Address + 1. The DMA processes the first descriptor and then waits for the application to advance the tail pointer.

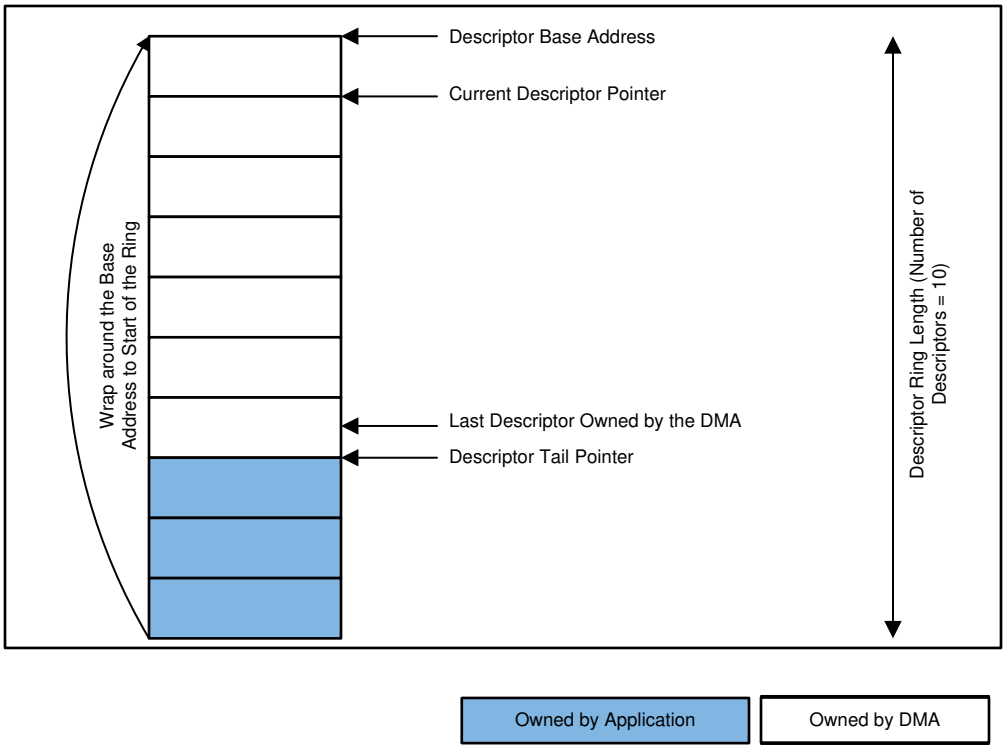


Figure 43-20. DMA Descriptor Ring

43.4.2 Transmit Descriptor

The DMA in Ethernet module requires at least one descriptor for a transmit packet. In addition to two buffers, two byte-count buffers, and two address pointers, the transmit descriptor has control fields that are used to control the MAC operation on per-transmit packet basis. The Transmit Normal descriptor has two formats: Read format and Write-Back format.

43.4.2.1 Transmit Normal Descriptor (Read Format)

Figure 43-21 shows the Read Format for a Transmit normal descriptor.

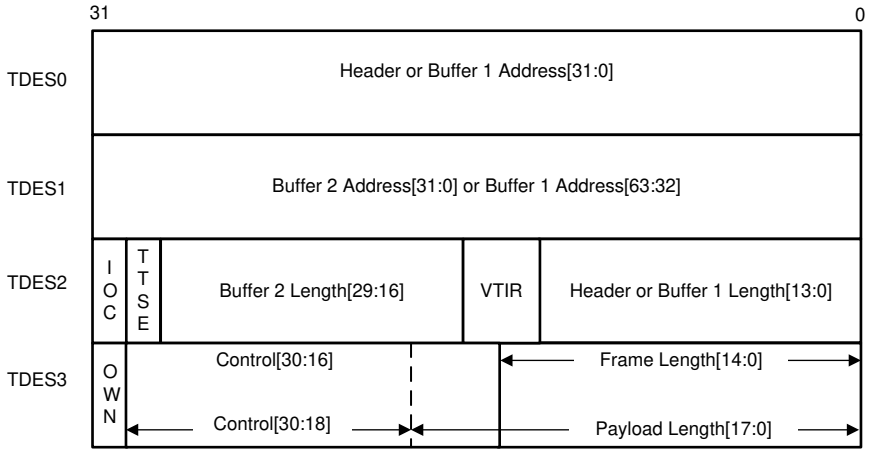


Figure 43-21. Transmit Normal Descriptor Read Format

43.4.2.1.1 TDES0 Normal Descriptor (Read Format)

Table 43-43. TDES0 Normal Descriptor (Read Format) Description

Bit	Name	Description
31-0	BUF1AP	Buffer 1 Address Pointer or TSO Header Address Pointer These bits indicate the physical address of Buffer 1. These bits indicate the TSO Header Address pointer when the following bits are set: ■ TSE bit of TDES3 ■ FD bit of TDES3

43.4.2.1.2 TDES1 Normal Descriptor (Read Format)

Table 43-44. TDES1 Normal Descriptor (Read Format) Description

Bit	Name	Description
31-0	BUF2AP	Buffer 2 or Buffer 1 Address Pointer This bit indicates the physical address of Buffer 2 when a descriptor ring structure is used. There is no limitation for the buffer address alignment. In 40- or 48-bit addressing mode, these bits indicate the most-significant 8- or 16- bits of the Buffer 1 Address Pointer.

43.4.2.1.3 TDES2 Normal Descriptor (Read Format)

Table 43-45. TDES2 Normal Descriptor (Read Format) Description

31	30	29:16	15:14	13:0
IOC	TTSE/ TMWD	B2L	VTIR	HL or B1L

Table 43-46. TDES2 Normal Descriptor (Read Format) Description

Bit	Name	Description
31	IOC	Interrupt on Completion This bit controls the setting of TI and ETI status bits in the DMA_CH#_Status register. When ETIC = 1 and TDES2[LD] = 0, this bit sets the ETI bit. When TDES3[LD] = 1, this bit sets the TI status bit.
30	TTSE/TMWD	Transmit Timestamp Enable or External TSO Memory Write Enable This bit enables the IEEE1588 time stamping for Transmit packet referenced by the descriptor, if TSE bit is not set. If TSE bit is set and external TSO memory is enabled, setting this bit disables external TSO memory writing for this packet.
29-16	B2L	Buffer 2 Length The driver sets this field. When set, this field indicates Buffer 2 length.
15-14	VTIR	VLAN Tag Insertion or Replacement These bits request the MAC to perform VLAN tagging or untagging before transmitting the packets. The application must set the CRC Pad Control bits appropriately when VLAN Tag Insertion, Replacement, or Deletion is enabled for the packet. The following list describes the values of these bits: 00: Do not add a VLAN tag. 01: Remove the VLAN tag from the packets before transmission. This option should be used only with the VLAN packets. 10: Insert a VLAN tag with the tag value programmed in the MAC_VLAN_Incl register or context descriptor. 11: Replace the VLAN tag in packets with the tag value programmed in the MAC_VLAN_Incl register or context descriptor. This option should be used only with the VLAN packets. These bits are valid when the Enable SA and VLAN Insertion on Tx option is selected while configuring the core.
13-0	HL or B1L	Header Length or Buffer 1 Length For Header length only bits [9:0] are taken. The size 13:0 is applicable only when interpreting buffer 1 length. If the TCP Segmentation Offload feature is enabled through the TSE bit of TDES3, this field is equal to the header length. When the TSE bit is set in TDES3, the header length includes the length in bytes from Ethernet Source address till the end of the TCP header. The maximum header length supported for TSO feature is 1023 bytes. The maximum header length supported for TSO feature is 1023 bytes. If the TCP Segmentation Offload feature is not enabled, this field is equal to Buffer 1 length.

43.4.2.1.4 TDES3 Normal Descriptor (Read Format)

Table 43-47. TDES3 Normal Descriptor (Read Format)

31	30	29	28	27:26	25:23	22:19	18	17:16	15	14:0
OWN	CTXT	FD	LD	CPC	SAIC	SLOTNUM or THL	TSE	CIC/TPL	TPL	FL/TPL

Table 43-48. TDES3 Normal Descriptor (Read Format) Description

Bit	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit after it completes the transfer of data given in the associated buffer(s).
30	CTXT	Context Type This bit should be set to 1'b0 for normal descriptor.
29	FD	First Descriptor When this bit is set, it indicates that the buffer contains the first segment of a packet.
28	LD	Last Descriptor When this bit is set, it indicates that the buffer contains the last segment of the packet. When this bit is set, the B1L or B2L field should have a non-zero value.
27-26	CPC	CRC Pad Control This field controls the CRC and Pad Insertion for Tx packet. This field is valid only when the first descriptor bit (TDES3[29]) is set. The following list describes the values of Bits[27:26]: 00: CRC and Pad Insertion The MAC appends the cyclic redundancy check (CRC) at the end of the transmitted packet of length greater than or equal to 60 bytes. The MAC automatically appends padding and CRC to a packet with length less than 60 bytes. 01: CRC Insertion (Disable Pad Insertion) The MAC appends the CRC at the end of the transmitted packet but it does not append padding. The application should ensure that the padding bytes are present in the packet being transferred from the Transmit Buffer, that is, the packet being transferred from the Transmit Buffer is of length greater than or equal to 60 bytes. 10: Disable CRC Insertion The MAC does not append the CRC at the end of the transmitted packet. The application should ensure that the padding and CRC bytes are present in the packet being transferred from the Transmit Buffer. 11: CRC Replacement The MAC replaces the last four bytes of the transmitted packet with recalculated CRC bytes. The application should ensure that the padding and CRC bytes are present in the packet being transferred from the Transmit Buffer. This field is valid only for the first descriptor. Note: When the TSE bit is set, the MAC ignores this field because the CRC and pad insertion is always done for segmentation.

Table 43-48. TDES3 Normal Descriptor (Read Format) Description (continued)

Bit	Name	Description
25-23	SAIC	SA Insertion Control These bits request the MAC to add or replace the Source Address field in the Ethernet packet with the value given in the MAC Address 0 register. The application must set the CRC Pad Control bits appropriately when SA Insertion Control is enabled for the packet. Bit 25 specifies the MAC Address Register (1 or 0) value that is used for Source Address insertion or replacement. The following list describes the values of Bits[24:23]: 00: Do not include the source address 01: Include or insert the source address. For reliable transmission, the application must provide frames without source addresses. 10: Replace the source address. For reliable transmission, the application must provide frames with source addresses. 11: Reserved These bits are valid when the First Segment control bit (TDES3 [29]) is set. This field is valid only for the first descriptor.
22-19	SLOTNUM or THL	SLOTNUM: Slot Number Control Bits in AV Mode These bits indicate the slot interval in which the data should be fetched from the corresponding buffers addressed by TDES0 or TDES1. When the Transmit descriptor is fetched, the DMA compares the slot number value in this field with the slot interval maintained in the RSN field DMA_CH#_Slot_Function_Control_Status. It fetches the data from the buffers only if a value matches. These bits are valid only for the AV channels. THL: TCP/UDP Header Length If the TSE bit is set, this field contains the length of the TCP/UDP header. The minimum value of this field must be 5 for TCP header. The value must be equal to 2 for UDP header. This field is valid only for the first descriptor.
18	TSE	TCP Segmentation Enable When this bit is set, the DMA performs the TCP/UDP segmentation or UDP fragmentation for a packet depending on the TSE_MODE[1:0] bit of the DMA_CH(#i)_Tx_Control Register. This bit is valid only if the FD bit is set.
17-16	CIC/TPL	Checksum Insertion Control or TCP Payload Length These bits control the checksum calculation and insertion. The following list describes the bit encoding: 00: Checksum Insertion Disabled. 01: Only IP header checksum calculation and insertion are enabled. 10: IP header checksum and payload checksum calculation and insertion are enabled, but pseudo-header checksum is not calculated in hardware. 11: IP Header checksum and payload checksum calculation and insertion are enabled, and pseudo-header checksum is calculated in hardware. This field is valid when the Enable Transmit TCP/IP Checksum Offload option is selected and the TSE bit is reset. When the TSE bit is set, this field contains the upper bits [17:16] of the TCP Payload (or IP Payload for UDP fragmentation). This allows the TCP/UDP packet length field to be spanned across TDES3[17:0] to provide 256 KB packet length support. This field is valid only for the first descriptor.
15	TPL	Reserved or TCP Payload Length When the TSE bit is reset, this bit is reserved. When the TSE bit is set, this is Bit 15 of the TCP payload length [17:0]. This field is valid only when the Enable TCP Segmentation Offloading for TCP/IP Packets option is selected while configuring the core.

Table 43-48. TDES3 Normal Descriptor (Read Format) Description (continued)

Bit	Name	Description
14-0	FL/TPL	Frame Length or TCP Payload Length This field is equal to the length of the packet to be transmitted in bytes. When the TSE bit is not set, this field is equal to the total length of the packet to be transmitted: Ethernet Header Length + TCP /IP Header Length – Preamble Length – SFD Length + Ethernet Payload Length When the TSE bit is set, this field is equal to the lower 15 bits of the TCP payload length in case of segmentation and IP payload in case of UDP fragmentation. In case of segmentation, this length does not include Ethernet header or TCP/UDP/IP header length. In case of fragmentation, this length does not include Ethernet header and IP header. When DWRR/WFQ algorithm is NOT enabled, value written into this field is not used when TSE=0.

43.4.2.2 Transmit Normal Descriptor (Write-Back Format)

The write-back format of the Transmit Descriptor includes timestamp low, timestamp high, OWN, and Status bits.

The write-back format is applicable only for the last descriptor of the corresponding packet. The LD bit (TDES3[28]) is set in the descriptor where the DMA writes back the status and timestamp information for the corresponding Transmit packet.

Figure 43-22 illustrates the write-back format of the Transmit Descriptor.

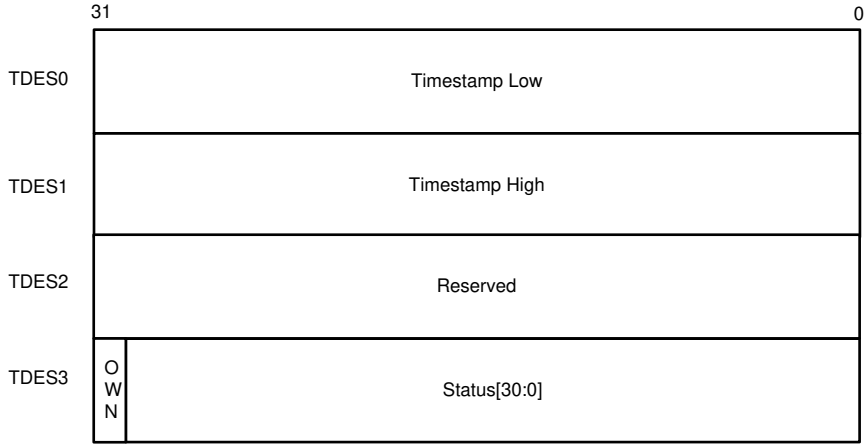


Figure 43-22. Transmit Write Back Format

43.4.2.2.1 TDES0 Normal Descriptor (Write-Back Format)

As described in Table 43-50, this format is only applicable to the last descriptor of a packet.

Table 43-49. TDES0 Normal Descriptor (Write-Back Format) Description

Bit	Name	Description
31-0	TTSL	Transmit Packet Timestamp Low The DMA updates this field with least significant 32 bits of the timestamp captured for the corresponding Transmit packet. The DMA writes the timestamp only if TTSE bit of TDES2 is set in the first descriptor of the packet. This field has the timestamp only if the Last Segment bit (LS) in the descriptor is set and the Timestamp status (TTSS) bit is set.

43.4.2.2.2 TDES1 Normal Descriptor (Write-Back Format)

Table 43-50. TDES1 Normal Descriptor (Write-Back Format) Description

Bit	Name	Description
31-0	TTSH	Transmit Packet Timestamp High The DMA updates this field with the most significant 32 bits of the timestamp captured for corresponding transmit packet. The DMA writes the timestamp only if the TTSE bit of TDES2 is set in the first descriptor of the packet. This field has the timestamp only if the Last Segment bit (LS) in the descriptor is set and Timestamp status (TTSS) bit is set

43.4.2.2.3 TDES2 Normal Descriptor (Write-Back Format)

This format is applicable only to the last descriptor of a packet.

Table 43-51. TDES2 Normal Descriptor (Write-Back Format) Description

Bit	Description
31-0	Reserved

43.4.2.2.4 TDES3 Normal Descriptor (Write-Back Format)

This format is applicable only to the last descriptor of a packet.

Table 43-52. TDES3 Normal Descriptor Layout (Write-Back Format)

31	30	29	28	27:24/ 22:18	23	17	16	15	14	13	12	11	10	9	8	7:4	3	2	1	0
OWN	CTXT	FD	LD	Rsvd	DE	TTSS	EUE	ES	JT	FF	PCE	LoC	NC	LC	EC	CC	ED	UF	DB	IHE

Table 43-53. TDES3 Normal Descriptor (Write-Back Format) Description

Bit	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the Ethernet module DMA owns the descriptor. The DMA clears this bit when it completes the packet transmission. After the write-back is complete, this bit is cleared to 0.
30	CTXT	Context Type This bit should be cleared to 0 for Normal descriptor.
29	FD	First Descriptor This bit indicates that the buffer contains the first segment of a packet.
28	LD	Last Descriptor This bit is set 1 for last descriptor of a packet. The DMA writes the status fields only in the last descriptor of the packet.
27-24	Rsvd	Reserved
23	DE	Descriptor Error When this bit is set, it indicates that the descriptor content is incorrect. The DMA sets this bit during write-back while closing the descriptor. Descriptor Errors can be: **Note 1:** When Descriptor Error occurs due to All 1s or CTXT, LD, and FD bits set to 1, the Transmit DMA closes the transmit descriptor with DE and LD bits set to 1. When IOC bit in TDES2 of corresponding first descriptor is set to 1, Transmit DMA will set the TI bit in the DMA_CH#_Status register **Note 2:** Based on CTXT, LD, and FD bits of the transmit descriptor, the subsequent descriptor might be considered as the First Descriptor (even if FD bit is not set) and partial packet is sent.
22-18	Rsvd	Reserved
17	TTSS	Tx Timestamp Status This status bit indicates that a timestamp has been captured for the corresponding transmit packet. When this bit is set, TDES0 and TDES1 have timestamp values that were captured for the Transmit packet. This field is valid only when the Last Segment control bit (TDES3 [28]) in a descriptor is set. This bit is valid only when IEEE1588 timestamping feature is enabled; otherwise, it is reserved.
16	EUE	ECC Uncorrectable Error Status Indicates the ECC uncorrectable error in the TSO memory. Note: Uncorrectable error in Transmit FIFO memory is reported with (Bit 13) FF = 1. This is because, all such packets are flushed by Ethernet module.

Table 43-53. TDES3 Normal Descriptor (Write-Back Format) Description (continued)

Bit	Name	Description
15	ES	Error Summary This bit indicates the logical OR of the following bits: ■TDES3[0]: IP Header Error ■TDES3[14]: Jabber Timeout ■TDES3[13]: Packet Flush ■TDES3[12]: Payload Checksum Error ■TDES3[11]: Loss of Carrier ■TDES3[10]: No Carrier ■TDES3[9]: Late Collision ■TDES3[8]: Excessive Collision ■TDES3[3]: Excessive Deferral ■TDES3[2]: Underflow Error This bit is also set when EUE (bit 16) is set.
14	JT	Jabber Timeout This bit indicates that the MAC transmitter has experienced a jabber time-out. This bit is set only when the JD bit of the MAC_Configuration register is not set.
13	FF	Packet Flushed This bit indicates that the DMA or MTL flushed the packet because of a software flush command given by the CPU.
12	PCE	Payload Checksum Error This bit indicates that the Checksum Offload engine had a failure and did not insert any checksum into the encapsulated TCP, UDP, or ICMP payload. This failure can be either because of insufficient bytes, as indicated by the Payload Length field of the IP Header or the MTL starting to forward the packet to the MAC transmitter in Store-and-Forward mode without the checksum having been calculated yet. This second error condition only occurs when the Transmit FIFO depth is less than the length of the Ethernet packet being transmitted to avoid deadlock, the MTL starts forwarding the packet when the FIFO is full, even in the store-and-forward mode. This error can also occur when Bus Error is detected during packet transfer. When the Full Checksum Offload engine is not enabled, this bit is reserved.
11	LoC	Loss of Carrier This bit indicates that Loss of Carrier occurred during packet transmission (that is, the gmii_crs_i signal was inactive for one or more transmit clock periods during packet transmission). This is valid only for the packets transmitted without collision and when the MAC operates in the half-duplex mode.
10	NC	No Carrier This bit indicates that the carrier sense signal form the PHY was not asserted during transmission.
9	LC	Late Collision This bit indicates that packet transmission was aborted because a collision occurred after the collision window (64 byte times including Preamble in MII mode). This bit is not valid if Underflow Error is set.
8	EC	Excessive Collision This bit indicates that the transmission was aborted after 16 successive collisions while attempting to transmit the current packet. If the DR bit is set in the MAC_Configuration register, this bit is set after first collision and the transmission of the packet is aborted.
7-4	CC	Collision Count This 4-bit counter value indicates the number of collisions occurred before the packet was transmitted. The count is not valid when the EC bit is set.

Table 43-53. TDES3 Normal Descriptor (Write-Back Format) Description (continued)

Bit	Name	Description
3	ED	Excessive Deferral This bit indicates that the transmission ended because of excessive deferral of over 24,288 bit times (155,680 bits times in 1000 Mbps mode or Jumbo Packet enabled mode) if DC bit is set in the MAC_Configuration register. When TBS is enabled in full duplex mode and this bit is set, it indicates that the frame has been dropped after the expiry time has reached.
2	UF	Underflow Error This bit indicates that the MAC aborted the packet because the data arrived late from the system memory. The underflow error can occur because of either of the following conditions: ■The DMA encountered an empty Transmit Buffer while transmitting the packe.t ■The application filled the MTL Tx FIFO slower than the MAC transmit rate. The transmission process enters the suspended state and sets the underflow bit corresponding to a queue in the MTL_Interrupt_Status register.
1	DB	Deferred Bit This bit indicates that the MAC deferred before transmitting because of presence of carrier. This bit is valid only in the half-duplex mode.
0	IHE	IP Header Error When IP Header Error is set, this bit indicates that the Checksum Offload engine detected an IP header error. This bit is valid only when Tx Checksum Offload is enabled. Otherwise, it is reserved. If COE detects an IP header error, it still inserts an IPv4 header checksum if the Ethernet Type field indicates an IPv4 payload In full duplex mode, when EST/Qbv is enabled and this bit is set, it indicates the frame drop status due to Frame Size error or Schedule Error.

43.4.2.3 Transmit Context Descriptor

The context descriptor is used to provide the timestamps for one-step timestamp correction, VLAN Tag ID for VLAN insertion feature. The Transmit Context descriptor can be provided any time before a packet descriptor. The context is valid for the current packet and subsequent packets. The context descriptor is used to provide the timestamps for one-step timestamp correction and VLAN Tag ID for VLAN insertion feature. Write back is done on a context descriptor only to reset the OWN bit.

The VLAN Tag IDs and MSS values, provided by the application in a context descriptor with their corresponding Valid bits set, are stored internally by the DMA. When the outer or inner VLAN tag is provided with the Valid bit set, the DMA always passes the last valid VLAN tag to the MTL. The application cannot invalidate the valid VLAN tag stored by the DMA. The VLAN tag is inserted or replaced based on the control inputs provided for the packet.

The Inner VLAN Tag Control input is used only for the next packet that immediately follows the context descriptor. The application must provide a context descriptor before the normal descriptor of each packet for which the DMA should use the inner VLAN Tag control input. Figure 43-23 shows the format of the Transmit Context descriptor.

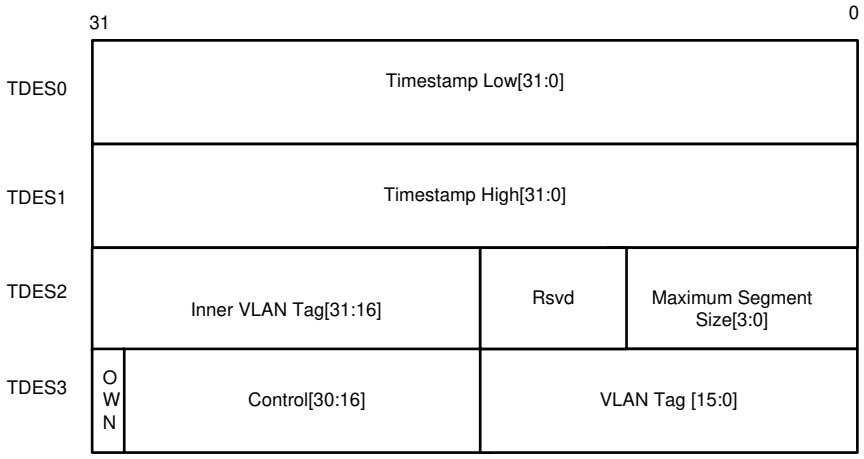


Figure 43-23. Transmit Context Descriptor Format

43.4.2.3.1 TDES0 Context Descriptor

Table 43-54. TDES0 Context Descriptor Description

Bit	Name	Description
31-0	TTSL	TDES0 Context Descriptor For one-step correction, the driver can provide the lower 32 bits of timestamp in this descriptor word. The DMA uses this value as the low word for doing one-step timestamp correction. This field is valid only if the OSTC and TCMSSV bits of TDES3 context descriptor are set.

43.4.2.3.2 TDES1 Context Descriptor

Table 43-55. TDES1 Context Descriptor Description

Bit	Name	Description
31-0	TTSH	Transmit Packet Timestamp High For one-step correction, the driver can provide the upper 32 bits of timestamp in this descriptor. The DMA uses this value as the high word for doing one-step timestamp correction. This field is valid only if the OSTC and TCMSSV bits of TDES3 context descriptor are set.

43.4.2.3.3 TDES2 Context Descriptor

Table 43-56. TDES2 Context Descriptor Description

Bit	Name	Description
31-16	IVT	Transmit Packet Timestamp High For one-step correction, the driver can provide the upper 32 bits of timestamp in this descriptor. The DMA uses this value as the high word for doing one-step timestamp correction. This field is valid only if the OSTC and TCMSSV bits of TDES3 context descriptor are set.
15-14	Rsvd	Reserved
13-0	MSS	Maximum Segment Size When the Enable TCP Segmentation Offloading for TCP/IP Packets option is selected, the driver can provide maximum segment size in this field. This segment size is used while segmenting the TCP/IP payload. This field is valid only if the TCMSSV bit of TDES3 context descriptor is set and the OSTC bit of the TDES3 context descriptor is reset.

43.4.2.3.4 TDES3 Context Descriptor

Table 43-57. TDES3 Context Descriptor Layout

31	30	29:28	27	26	25:24	23	22:18	17	16	15:0
OWN	CTXT	Rsvd	OSTC	TCMSSV	Rsvd	CDE	Rsvd	IVLTV	VLTV	VT

Table 43-58. TDES3 Context Descriptor Description

Bit	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the Ethernet module DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit immediately after the read.
30	CTXT	Context Type This bit should be set to 1 for Context descriptor.
29-28	Rsvd	Reserved
27	OSTC	One-Step Timestamp Correction Enable When this bit is set, the DMA performs a one-step timestamp correction with reference to the timestamp values provided in TDES0 and TDES1.
26	TCMSSV	One-Step Timestamp Correction Input or MSS Valid When this bit and the OSTC bit are set, it indicates that the Timestamp Correction input provided in TDES0 and TDES1 is valid. When the OSTC bit is reset and this bit and the TSE bit of TDES3 are set in subsequent normal descriptor, it indicates that the MSS input in TDES2 is valid.
25-24	Rsvd	Reserved
23	DE	Descriptor Error When this bit is set, it indicates that the descriptor content is incorrect.The DMA sets this bit during write-back while closing the context descriptor. Descriptor Errors can be: ■ Incorrect sequence from the context descriptor. For example, a location before the first descriptor for a packet. ■ All 1s. ■ CD, LD, and FD bits set to 1. **Note 1:** When Descriptor Error occurs due to All 1s or CTXT, LD, and FD bits set to 1, the Transmit DMA closes the transmit descriptor with DE and LD bits set to 1. When IOC bit in TDES2 of corresponding first descriptor is set to 1, Transmit DMA will set the TI bit in the DMA_CH#_Status register **Note 2:** Based on CTXT, LD, and FD bits of the transmit descriptor, the subsequent descriptor might be considered as the First Descriptor (even if FD bit is not set) and partial packet is sent.

Table 43-58. TDES3 Context Descriptor Description (continued)

Bit	Name	Description
22-20	Rsvd	Reserved
19-18	IVTIR	Inner VLAN Tag Insert or Replace When this bit is set, these bits request the MAC to perform Inner VLAN tagging or un-tagging before transmitting the packets. If the packet is modified for VLAN tags, the MAC automatically recalculates and replaces the CRC bytes. The following list describes the values of these bits: 00: Do not add the inner VLAN tag. 01: Remove the inner VLAN tag from the packets before transmission. This option should be used only with the VLAN frames. 10: Insert an inner VLAN tag with the tag value programmed in the MAC_Inner_VLAN_Incl register or context descriptor. 11: Replace the inner VLAN tag in packets with the tag value programmed in the MAC_Inner_VLAN_Incl register or context descriptor. This option should be used only with the VLAN frames. These bits are valid when the Enable SA and VLAN Insertion on Tx and Enable Double VLAN Processing options are selected.
17	IVLTV	Inner VLAN Tag Valid When this bit is set, it indicates that the IVT field of TDES2 is valid.
16	VLTV	VLAN Tag Valid When this bit is set, it indicates that the VT field of TDES3 is valid.
15-0	VT	VLAN Tag This field contains the VLAN Tag to be inserted or replaced in the packet. This field is used as VLAN Tag only when the VLTI bit of the MAC_VLAN_Incl register is reset.

43.4.3 Receive Descriptor

The DMA in Ethernet module attempts to read a descriptor only if the Tail Pointer is different from the Base Pointer or current pointer. It is recommended to have a descriptor ring with a length that can accommodate at least two complete packets received by the MAC. Otherwise, the performance of the DMA is impacted greatly because of the unavailability of the descriptors. In such situations, the RxFIFO in MTL becomes full and starts dropping packets.

The following Receive Descriptors are present:

- Normal descriptors
- Context descriptors

All RX descriptors are prepared by the software and given to the DMA as “Normal” Descriptors with the content as shown in Receive Normal Descriptor (Read Format). The DMA reads this descriptor and after transferring a received packet (or part of) to the buffers indicated by the descriptor, the Rx DMA will close the descriptor with the corresponding packet status. The format of this status is given in the “Receive Normal Descriptor (Write-Back Format)”. For some packets, the normal descriptor bits are not enough to write the complete status. For such packets, the RX DMA writes the extended status to the next descriptor (without processing or using the Buffers/ Pointers embedded in that descriptor). The format and content of the descriptor write back is described in Section 43.4.3.3.

43.4.3.1 Receive Normal Descriptor (Read Format)

The read format for a Receive Normal descriptor is made up of a header or Buffer 1 address, reserved field, payload or Buffer 2 or Next Descriptor address, a 30-bit reserved filed, OWN bit, and an interrupt bit.

Figure 43-24 shows the Read Format for a Receive normal descriptor.

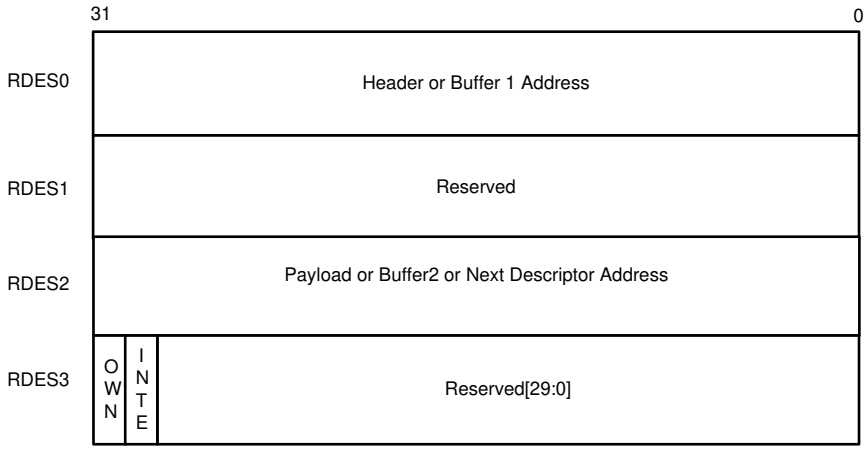


Figure 43-24. Receive Normal Descriptor Read Format

Note

In the Receive Descriptor (Read Format), if the Buffer Address field is all 0s, Ethernet module does not transfer data to that buffer and skips to the next buffer or next descriptor.

43.4.3.1.1 RDES0 Normal Descriptor (Read Format)

Table 43-59 describes the read format of the RDES1 Normal Descriptor.

Table 43-59. RDES0 Normal Descriptor (Read Format) Description

Bit	Name	Description
31-0	BUF1AP	RDES0 Normal Descriptor (Read Format) When the SPH bit of Control register of a channel is reset, these bits indicate the physical address of Buffer 1. When the SPH bit is set, these bits indicate the physical address of Header Buffer where the Rx DMA writes the L2/L3/L4 header bytes of the received packet. The application can program a byte-aligned address for this buffer which means that the LS bits of this field can be non-zero. However, while transferring the start of packet, the DMA performs a Write operation with RDES0[1:0] as zero. However, the packet data is shifted as per actual offset as given by buffer address pointer. If the address pointer points to a buffer where the middle or last part of the packet is stored, the DMA ignores the offset address and writes to the full location as indicated by the data-width.

43.4.3.1.2 RDES1 Normal Descriptor (Read Format)

Table 43-60 describes the read format of the RDES1 Normal Descriptor.

Table 43-60. RDES1 Normal Descriptor (Read Format) Description

Bit	Name	Description
31-0	Reserved or BUF1AP	In 64-bit addressing mode, this field contains the most-significant 32 bits of the Buffer 1 Address Pointer. Otherwise, this field is reserved.

43.4.3.1.3 RDES2 Normal Descriptor (Read Format)

Table 43-61 describes the read format of the RDES2 Normal Descriptor.

Table 43-61. RDES2 Normal Descriptor (Read Format) Description

Bit	Name	Description
31-0	BUF1AP	Buffer 2 Address Pointer These bits indicate the physical address of Buffer 2. When the SPH bit of the DMA_CH#_Control register is set, the buffer address pointer must be bus width-aligned, that is, RDES2[3:0, 2:0, or 1:0] = 0 corresponding to 128, 64, or 32 bus width. LSBs are ignored internally. When the SPH bit of the DMA_CH#_Control register is reset, there is no limitations on the RDES2 value. However, the RxDMA uses the LS Bits of the pointer address only while transferring the start bytes of a packet. If the BUF2AP is giving the address of a buffer in which the middle or last part of a packet is stored, the DMA ignores BUF2AP[3:0 or 2:0 or 1:0] (corresponding to 128-, 64-, or 32-bit data-bus) and writes to the complete location.

43.4.3.1.4 RDES3 Normal Descriptor (Read Format)

Table 43-62 describes the read format of the RDES3 Normal Descriptor.

Table 43-62. RDES3 Normal Descriptor

31	30	19-26	25	24	23:0
OWN	IOC	Rsvd	BUF2V	BUF1V	Rsvd

Table 43-63. RDES3 Normal Descriptor (Read Format) Description

Bit	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the Ethernet module DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit when either of the following conditions is true: ■ The DMA completes the packet reception ■ The buffers associated with the descriptor are full
30	IOC	Interrupt Enabled on Completion When this bit is set, an interrupt is issued to the application when the DMA closes this descriptor.
29-26	Rsvd	Reserved
25	BUF2V	Buffer 2 Address Valid When this bit is set, it indicates to the DMA that the buffer 2 address specified in RDES2 is valid. The application must set this bit so that the DMA can use the address, to which the Buffer 2 address in RDES2 is pointing, to write received packet data.
24	BUF1V	Buffer 1 Address Valid When set, this indicates to the DMA that the buffer 1 address specified in RDES1 is valid. The application must set this value if the address pointed to by Buffer 1 address in RDES1 can be used by the DMA to write received packet data.
23-0	Rsvd	Reserved

43.4.3.2 Receive Normal Descriptor (Write-Back Format)

Figure 43-25 illustrates the write-back format for a Receive Normal descriptor.

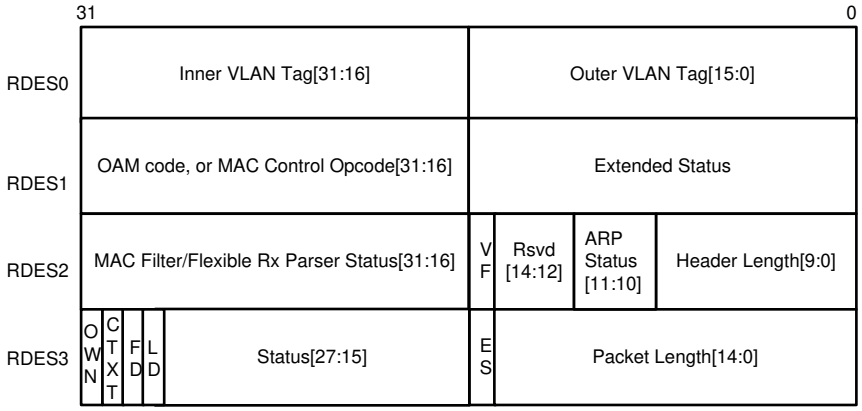


Figure 43-25. Receive Normal Write Back Format

Note

When Flexible RX Parser is enabled, RDES2[31:16] indicates Parser status, and not the MAC filter status. MAC filter status is not available when Flexible RX parser is enabled.

43.4.3.2.1 RDES0 Normal Descriptor (Write-Back Format)

Table 43-64. RDES0 Normal Descriptor (Write-Back Format) Description

Bit	Name	Description
31-16	IVT	Inner VLAN Tag This field contains the Inner VLAN tag of the received packet if the RS0V bit of RDES3 is set. This is valid only when Double VLAN tag processing and VLAN tag stripping are enabled.
15-0	OVT	Outer VLAN Tag This field contains the Outer VLAN tag of the received packet if the RS0V bit of RDES3 is set.

43.4.3.2.2 RDES1 Normal Descriptor (Write-Back Format)

The Status fields in write-back format are valid only for the last descriptor (RDES3[28] is set)

Table 43-65. RDES1 Normal Descriptor (Write-Back Format)

31:16	15	14	13	12	11:8	7	6	5	4	3	2:0
OPC	TD	TSA	PV	PFT	PMT	IPCE	IPCB	IPV6	IPV4	IPHE	PT

Table 43-66. RDES1 Normal Descriptor (Write-Back Format) Description

Bit	Name	Description
31-16	OPC	OAM Sub-Type Code or MAC Control Packet opcode If LT Bits[18:16] of RDES3 are set to 111, this field contains the OAM sub-type and code fields. If LT Bits[18:16] of RDES3 are set to 110, this field contains the MAC Control packet opcode field.
15	TD	Timestamp Dropped This bit indicates that the timestamp was captured for this packet but it got dropped in the MTL Rx FIFO because of overflow. This bit is available only when you select the Timestamp feature. Otherwise, this bit is reserved.
14	TSA	Timestamp Available When Timestamp is present, this bit indicates that the timestamp value is available in a context descriptor word 2 (RDES2) and word 1(RDES1). This is valid only when the Last Descriptor bit (RDES3[28]) is set. The context descriptor is written in the next descriptor just after the last normal descriptor for a packet.
13	PV	PTP Version This bit indicates that the received PTP message has the IEEE 1588 version 2 format. When this bit is reset, it indicates the IEEE 1588 version 1 format. This bit is available only when you select the Timestamp feature. Otherwise, this bit is reserved.
12	PFT	PTP Packet Type This bit indicates that the PTP message is sent directly over Ethernet. This bit is available only when you select the Timestamp feature. Otherwise, this bit is reserved.
11-8	PMT	PTP Message Type These bits are encoded to give the type of the message received: 0000: No PTP message received 0001: SYNC (all clock types) 0010: Follow_Up (all clock types) 0011: Delay_Req (all clock types) 0100: Delay_Resp (all clock types) 0101: Pdelay_Req (in peer-to-peer transparent clock) 0110: Pdelay_Resp (in peer-to-peer transparent clock) 0111: Pdelay_Resp_Follow_Up (in peer-to-peer transparent clock) 1000: Announce 1001: Management 1010: Signaling 1011–1110: Reserved 1111: PTP packet with Reserved message type These bits are available only when you select the Timestamp feature.

Table 43-66. RDES1 Normal Descriptor (Write-Back Format) Description (continued)

Bit	Name	Description
7	IPCE	IP Payload Error When this bit is set, it indicates one of the following: ■ The 16-bit IP payload checksum (that is, the TCP, UDP, or ICMP checksum) calculated by the MAC does not match the corresponding checksum field in the received segment. ■ The TCP, UDP, or ICMP segment length does not match the payload length value in the IP Header field. ■ The TCP, UDP, or ICMP segment length is less than minimum allowed segment length for TCP, UDP, or ICMP. Bit 15 (ES) of RDES3 is not set when this bit is set.
6	IPCB	IP Checksum Bypassed This bit indicates that the checksum offload engine is bypassed. This bit is available when you select the Enable Receive TCP/IP Checksum Check feature.
5	IPV6	IPv6 header Present This bit indicates that an IPV6 header is detected. When the Enable Split Header Feature option is selected and the SPH bit of Control Register of a channel is set, the IPV6 header is available in the header buffer area to which RDES0 is pointing.
4	IPV4	IPV4 Header Present This bit indicates that an IPV4 header is detected. When the SPH bit of RDES3 is set, the IPV4 header is available in the header buffer area to which RDES0 is pointing.
3	IPHE	IP Header Error When this bit is set, it indicates one of the following: ■ The 16-bit IPv4 header checksum calculated by the MAC does not match the received checksum bytes. ■ The IP datagram version is not consistent with the Ethernet Type value. ■ Ethernet packet does not have the expected number of IP header bytes. This bit is valid when either Bit 5 or Bit 4 is set. This bit is available when you select the Enable Receive TCP/IP Checksum Check feature.
2-0	PT	Payload Type These bits indicate the type of payload encapsulated in the IP datagram processed by the Receive Checksum Offload Engine (COE): 000: Unknown type or IP/AV payload is not processed 001: UDP 010: TCP 011: ICMP 100: IGMP if IPV4 Header Present bit is set else DCB (LLDP) Control Packet 101: AV Untagged Control Packet 110: AV Tagged Data Packet 111: AV Tagged Control Packet If the COE does not process the payload of an IP datagram because there is an IP header error or fragmented IP, these bitsare cleared to 000.

43.4.3.2.3 RDES2 Normal Descriptor (Write-Back Format)

Table 43-67. RDES2 Normal Descriptor (Write-Back Format)

31:29	28	27	26:19	18	17	16	15	14	13:11	10	9:0
L3L4FM	L4FM	L3FM	MADRM	HF	DAF	SAF	OTS	ITS	Rsvd	ARPNR	HL

Table 43-68. RDES2 Normal Descriptor (Write-Back Format) Description

Bit	Name	Description
31-29	L3L4FM	Layer 3 and Layer 4 Filter Number Matched These bits indicate the number of the Layer 3 and Layer 4 Filter that matched the received packet: 000: Filter 0 001: Filter 1 010: Filter 2 011: Filter 3 100: Filter 4 101: Filter 5 110: Filter 6 111: Filter 7 This field is valid only when Bit 28 or Bit 27 is set high. When more than one filter matches, these bits give the number of the lowest filter. Note: This status is not available when Flexible RX Parser is enabled.
28	L4FM	Layer 4 Filter Match When this bit is set, it indicates that the received packet matches one of the enabled Layer 4 Port Number fields. This status is given only when one of the following conditions is true: ■ Layer 3 fields are not enabled and all enabled Layer 4 fields match. ■ All enabled Layer 3 and Layer 4 filter fields match. When more than one filter matches, this bit gives the layer 4 filter status of the filter indicated by Bits[31:29]. Note: This status is not available when Flexible RX Parser is enabled.
27	L3FM	Layer 3 Filter Match When this bit is set, it indicates that the received packet matches one of the enabled Layer 3 IP Address fields. This status is given only when one of the following conditions is true: ■ All enabled Layer 3 fields match and all enabled Layer 4 fields are bypassed. ■ All enabled filter fields match. When more than one filter matches, this bit gives the layer 3 filter status of filter indicated by Bits[31:29]. Note: This status is not available when Flexible RX Parser is enabled.
26-19	MADRM	MAC Address Match or Hash Value When the HF bit is reset, this field contains the MAC address register number that matched the Destination address of the received packet. This field is valid only if the DAF bit is reset. When the HF bit is set, this field contains the hash value computed by the MAC. A packet passes the hash filter when the bit corresponding to the hash value is set in the hash filter register. Note: This status is not available when Flexible RX Parser is enabled.
18	HF	Hash Filter Status When this bit is set, it indicates that the packet passed the MAC address hash filter. Bits[26:19] indicate the hash value. Note: This status is not available when Flexible RX Parser is enabled.

Table 43-68. RDES2 Normal Descriptor (Write-Back Format) Description (continued)

Bit	Name	Description
17	DAF/RXPI	Destination Address Filter Fail When Flexible RX Parser is disabled and this bit is set, it indicates that the packet failed the DA Filter in the MAC. When Flexible RX Parser is enabled, this bit is set to indicate that the packet parsing is incomplete (RXPI) due to ECC error. Note: When this bit is set, ES bit of RDES3 is also set.
16	SAF/RXPD	SA Address Filter Fail When Flexible RX Parser is disabled and this bit is set, it indicates that the packet failed the SA Filter in the MAC. When Flexible RX Parser is enabled, this bit is set to indicate that the packet is dropped (RXPD) by the parser. Note: When this bit is set, ES bit of RDES3 is also set.
15	OTS	Outer VLAN Tag Filter Status This bit is valid for both Single and Double VLAN Tagged frames.
14	ITS	Inner VLAN Tag Filter Status (ITS) This bit is valid only for Double VLAN Tagged frames, when Double VLAN Processing is enabled. For more information, see the Filter Status topic.
13-11	Rsvd	Reserved
10	ARPNR	ARP Reply Not Generated When this bit is set, it indicates that the MAC did not generate the ARP Reply for received ARP Request packet. This bit is set when the MAC is busy transmitting ARP reply to earlier ARP request (only one ARP request is processed at a time). This bit is reserved when the Enable IPv4 ARP Offload option is not selected.
9-0	HL	L3/L4 Header Length This field contains the length of the header of the packet split by the MAC at L3 or L4 header boundary as identified by the MAC receiver. This field is valid only when the first descriptor bit is set (FD = 1). The header data is written to the Buffer 1 address of the corresponding descriptor. If the header length is zero, this field is not valid. It implies that the MAC did not identify and split the header. This field is valid when the Enable Split Header Feature option is selected.

43.4.3.2.4 RDES3 Normal Descriptor (Write-Back Format)

Table 43-69. RDES3 Normal Descriptor (Write-Back Format)

31	30	29	28	27	26	25	24	23	22	21	20	19	18:16	15	14:0
OWN	CTXT	FD	LD	RS2V	RS1V	RS0V	CE	GP	RWT	OE	RE	DE	LT	ES	PL

Table 43-70. RDES3 Normal Descriptor (Write-Back Format) Description

Bit	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the Ethernet module DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit when either of the following conditions is true: ■ The DMA completes the packet reception. ■ The buffers associated with the descriptor are full.
30	CTXT	Receive Context Descriptor When this bit is set, it indicates that the current descriptor is a context type descriptor. The DMA writes 0 to this bit for normal receive descriptor. When CTXT and FD bits are used together, {CTXT, FD}: 00: Intermediate Descriptor 01: First Descriptor 10: Reserved 11: Descriptor Error (due to all 1s) Note: When Descriptor Error occurs, the Receive DMA closes the receive descriptor indicating Descriptor Error. This receive descriptor is skipped and the buffer addresses are not used to write the packet data Receive DMA will set the CDE bit in DMA_CH#_Status register but not the RI bit even when IOC is set, as this is not marked as last receive descriptor for the packet. The subsequent valid receive descriptor is used to write the packet data.
29	FD	First Descriptor When this bit is set, it indicates that this descriptor contains the first buffer of the packet. If the size of the first buffer is 0, the second buffer contains the beginning of the packet. If the size of the second buffer is also 0, the next descriptor contains the beginning of the packet. See the CTXT bit description for details of using the CTXT bit and FD bit together.
28	LD	Last Descriptor When this bit is set, it indicates that the buffers to which this descriptor is pointing are the last buffers of the packet.
27	RS2V	Receive Status RDES2 Valid When this bit is set, it indicates that the status in RDES2 is valid and it is written by the DMA. This bit is valid only when the LD bit of RDES3 is set.
26	RS1V	Receive Status RDES1 Valid When this bit is set, it indicates that the status in RDES1 is valid and it is written by the DMA. This bit is valid only when the LD bit of RDES3 is set.
25	RS0V	Receive Status RDES0 Valid When this bit is set, it indicates that the status in RDES0 is valid and it is written by the DMA. This bit is valid only when the LD bit of RDES3 is set.
24	CE	CRC Error When this bit is set, it indicates that a Cyclic Redundancy Check (CRC) Error occurred on the received packet. This field is valid only when the LD bit of RDES3 is set.
23	GP	Giant Packet When this bit is set, it indicates that the packet length exceeds the specified maximum Ethernet size of 1518, 1522, or 2000 bytes (9018 or 9022 bytes if jumbo packet enable is set). Note: Giant packet indicates only the packet length. It does not cause any packet truncation.
22	RWT	Receive Watchdog Timeout When this bit is set, it indicates that the Receive Watchdog Timer has expired while receiving the current packet. The current packet is truncated after watchdog timeout.

Table 43-70. RDES3 Normal Descriptor (Write-Back Format) Description (continued)

Bit	Name	Description
21	OE	Overflow Error When this bit is set, it indicates that the received packet is damaged because of buffer overflow in Rx FIFO. Note: This bit is set only when the DMA transfers a partial packet to the application. This happens only when the Rx FIFO is operating in the threshold mode. In the store-and-forward mode, all partial packets are dropped completely in Rx FIFO.
20	RE	Receive Error When this bit is set, it indicates that the gmii_rxer_i signal is asserted while the gmii_rxdv_i signal is asserted during packet reception. This error also includes carrier extension error in the MII and half-duplex mode. Error can be of less or no extension, or error (rxd!= 0f) during extension.
19	DE	Dribble Bit Error When this bit is set, it indicates that the received packet has a non-integer multiple of bytes (odd nibbles). This bit is valid only in the MII Mode.
18-16	LT	Length/Type Field This field indicates if the packet received is a length packet or a type packet. The encoding of the 3 bits is: 000: The packet is a length packet 001: The packet is a type packet 010: Reserved 011: The packet is a ARP Request packet type 100: The packet is a type packet with VLAN Tag 101: The packet is a type packet with Double VLAN Tag 110: The packet is a MAC Control packet type 111: The packet is a OAM packet type
15	ES	Error Summary When this bit is set, it indicates the logical OR of the following bits: ■ RDES3[24]: CRC Error ■ RDES3[19]: Dribble Error ■ RDES3[20]: Receive Error ■ RDES3[22]: Watchdog Timeout ■ RDES3[21]: Overflow Error ■ RDES3[23]: Giant Packet ■ RDES2[17]: Destination Address Filter Fail, when Flexible RX Parser is enabled ■ RDES2[16]: SA Address Filter Fail, when Flexible RX Parser is enabled This field is valid only when the LD bit of RDES3 is set.
14-0	PL	Packet Length These bits indicate the byte length of the received packet that was transferred to system memory (including CRC). This field is valid when the LD bit of RDES3 is set and Overflow Error bits are reset. The packet length also includes the two bytes appended to the Ethernet packet when IP checksum calculation is enabled and the received packet is not a MAC control packet. This field is valid when the LD bit of RDES3 is set. When the Last Descriptor and Error Summary bits are not set, this field indicates the accumulated number of bytes that have been transferred for the current packet.

43.4.3.3 Receive Context Descriptor

This descriptor is read-only for the application. Only the DMA can write to this descriptor. The context descriptor provides information about the extended status related to the last received packet. Bit 30 of RDES3 indicates the context type descriptor. Figure 43-26 shows the format of the Receive Context descriptor.

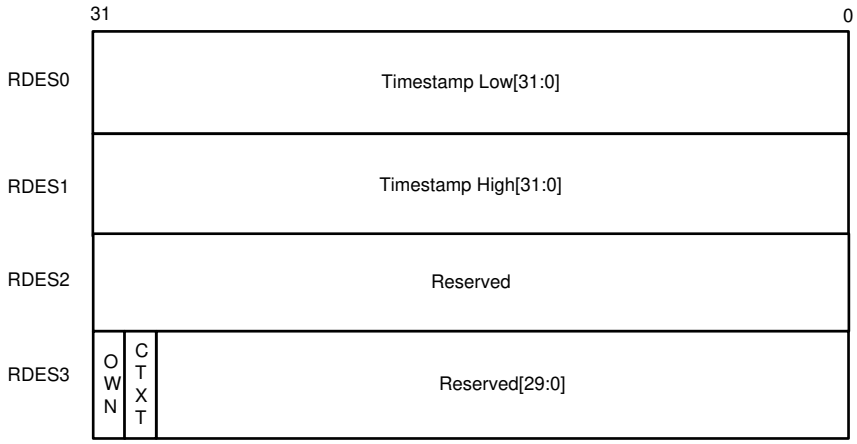


Figure 43-26. Receive Context Descriptor Format

43.4.3.3.1 RDES0 Context Descriptor

Table 43-71. RDES0 Context Descriptor

Bit	Name	Description
31-0	RTSL	Receive Packet Timestamp Low The DMA updates this field with the 32 least significant bits of the timestamp captured for corresponding Receive packet. When this field and the RTSH field of RDES1 show all-ones value, the timestamp must be considered as corrupt.

43.4.3.3.2 RDES1 Context Descriptor

Table 43-72. RDES1 Context Descriptor

Bit	Name	Description
31-0	RTSH	Receive Packet Timestamp High The DMA updates this field with the 32 most significant bits of the timestamp captured for corresponding receive packet. When this field and the RTSL field of RDES0 show all-ones value, the timestamp must be considered as corrupt.

43.4.3.3.3 RDES2 Context Descriptor

Table 43-73. RDES2 Context Descriptor

Bit	Description
31-0	Reserved

43.4.3.3.4 RDES3 Context Descriptor

Table 43-74. RDES3 Context Descriptor

31	30	29	28:0
OWN	CTXT	DE	Rsvd

Table 43-75. RDES3 Context Descriptor Description

Bit	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit when either of the following conditions is true: ■ The DMA completes the packet reception. ■ The buffers associated with the descriptor are full.
30	CTXT	Receive Context Descriptor When this bit is set, it indicates that the current descriptor is a context descriptor. The DMA writes 1 to this bit for context descriptor. The DMA writes 11 to indicate a descriptor error due to all 1s. When CTXT and DE bits are used together, {CTXT, DE}: 00: Reserved 01: Reserved 10: Context Descriptor 11: Descriptor Error Note: When Descriptor Error occurs, the Receive DMA closes the receive descriptor indicating Descriptor Error. This receive descriptor is skipped and the buffer addresses are not used to write the packet data Receive DMA will set the CDE bit in DMA_CH#_Status register but not the RI bit even when IOC is set, as this is not marked as last receive descriptor for the packet. The subsequent valid receive descriptor is used to write the packet data.
29	DE	Descriptor Error See the CTXT bit description for details of using theDE bit along with CTXT bit.
28-0	Rsvd	Reserved

43.5 Programming

The sequence of steps to program the Ethernet module is detailed in this section

43.5.1 Initializing DMA

Complete the following steps to initialize the DMA:

1. Provide a software reset. This resets all of the MAC internal registers and logic (bit-0 of DMA_Mode).
2. Wait for the completion of the reset process (poll bit 0 of the DMA_Mode, which is only cleared after the reset operation is completed).
3. Program the following fields to initialize the DMA_SysBus_Mode register:
   a. AAL
   b. Fixed burst or undefined burst
   c. Burst mode values in case of AHB bus interface, OSR_LMT in case of AXI bus interface.
   d. If fixed length value is enabled, select the maximum burst length possible on the AXI Bus (bits [7:1])
4. Create a descriptor list for transmit and receive. In addition, ensure that the descriptors are owned by DMA (set bit 31 of descriptor TDES3/RDES3). For more information about descriptors, see Descriptors
5. Program the Transmit and Receive Ring length registers (DMA_CH(#i)_TxDesc_Ring_Length (for i=0; i<=1) and DMA_CH(#1)_RxDesc_Ring_Length (for i=0; i<=1)). The ring length programmed must be at least 4.

Note

The descriptor addrfess from the start to the end of the ring must not cross the 4GB boundary.

6. Initialize receive and transmit descriptor list address with the base address of the transmit and receive descriptor (DMA_CH(#i)TxDesc_List_Address (for i = 0; i <=1), DMA_CH (#i)_RxDesc_List_Address (for i=0; i <=1)). also, program transmit and receive tail pointer registers indicating to the DMA about the available descriptors (DMA_CH (#i)_TxDesc_Tail_Pointer (for i=0; i <= 1) and DMA_CH (#i)_TxDesc_Tail_Pointer (for i=0; i<=1))..,

Note

(For 40-bit or 48-bit addressing mode, program the higher address List registers DMA_CH[n]_TxDesc_List_HAddress, DMA_CH[n]_RxDesc_List_HAddress). The tailpointer registers must be advanced to the location immediately after the descriptors that are set, for the DMA to know that additional descriptors are available.

43.5.2 Initializing MTL Registers

The Transaction Layer (MTL) registers must be initialized to establish the transmit and receive operating modes and commands.

Complete the following steps to initialize the MTL registers:

1. Program the Tx Scheduling (SCHALG) and Receive Arbitration Algorithm (RAA) fields in MTL_Operation_Mode to initialize the MTL operation in case of multiple Tx and Rx queues.
2. Program the Receive Queue to DMA mapping in MTL_RxQ_DMA_Map0 and MTL_RxQ_DMA_Map1 registers.
3. Program the following fields to initialize the mode of operation in the MTL_TxQ0_Operation_Mode register.
   a. Transmit Store And Forward (TSF) or Transmit Threshold Control (TTC) in case of threshold mode
   b. Transmit Queue Enable (TXQEN) to value 2'b10 to enable Transmit Queue0
   c. Transmit Queue Size (TQS)

   a.
4. Program the following fields to initialize the mode of operation in the MTL_RxQ0_Operation_Mode register:
   a. Receive Store and Forward (RSF) or RTC in case of Threshold mode
   b. Flow Control Activation and De-activation thresholds for MTL Receive FIFO (RFA and RFD)
   c. Error Packet and undersized good Packet forwarding enable (FEP and FUP)
   d. Receive Queue Size (RQS)
5. Repeat previous two steps for all MTL Tx and Rx queues.

43.5.3 Initializing MAC

The MAC configuration registers establish the operating mode of the MAC. These registers must be initial- ized before initializing the DMA.

The following MAC Initialization operations can be performed after DMA initialization. If the MAC initialization is completed before the DMA is configured, enable the MAC receiver (last step in the following sequence) only after the DMA is active. Otherwise, received frames fill the Rx FIFO and overflow.

1. Provide the MAC address registers: MAC_Address0_High and MAC_Address0_Low. If more than one MAC address is to be enabled program the MAC addresses appropriately.
2. Program the following fields to set the appropriate filters for the incoming frames in the MAC_Pack- et_Filter register:
   a. Receive All
   b. Promiscuous mode
   c. Hash or Perfect Filter
   d. Unicast, multicast, broadcast, and control frames filter settings
3. Program the following fields for proper flow control in the MAC_Q0_Tx_Flow_Ctrl register:
   a. Pause time and other Pause frame control bits
   b. Transmit Flow control bits
   c. Flow Control Busy
4. Program the MAC_Interrupt_Enable register, as required, and if applicable, for your configuration.
5. Program the appropriate fields in the MAC_Configuration register. For ex: Inter-packet gap while transmission and jabber disable.
6. Set bit 0 and 1 in MAC_Configuration registers to start the MAC transmitter and receiver

43.5.4 Performing Normal Receive and Transmit Operation

During normal operation of the Ethernet module, normal and transmit interrupts are read, descriptors polled, the DMA is suspended (if it does not own descriptors), and values of current host transmitter or receiver descriptor pointers are read for debugging.

For normal operation, complete the following steps:

1. For normal transmit and receive interrupts, read the interrupt status. Then, poll the descriptors, reading the status of the descriptor owned by the Host (either transmit or receive).
2. Set appropriate values for the descriptors, ensuring that transmit and receive descriptors are owned by the DMA to resume the transmission and reception of data.
3. If the descriptors are not owned by the DMA (or no descriptor is available), the DMA goes into SUSPEND state. The transmission or reception can be resumed by freeing the descriptors and writing the descriptor tail pointer to Tx/Rx tail pointer register (DMA_CH[n]_TxDesc_Tail_Pointer and DMA_CH[n]_RxDesc_Tail_Pointer).
4. The values of the current host transmitter or receiver descriptor address pointer can be read for the debug process (DMA_CH[n]_Current_App_TxDesc and DMA_CH[n]_Current_App_RxDesc).
5. The values of the current host transmit buffer address pointer and receive buffer address pointer can be read for the debug process (Register DMA_CH[n]_Current_App_TxBuffer and DMA_CH[n]_Current_App_RxBuffer).

43.5.5 Stopping and Starting Transmission

Complete the following steps to pause the transmission for some time. The steps are provided for Channel 0.

1. Disable the Transmit DMA (if applicable) by clearing Bit 0 (ST) of DMA
2. Wait for any previous frame transmissions to complete. You can check this by reading the appropriate bits of MTL_TxQ0_Debug Register (TRCSTS is not 01 and TXQSTS=0).
3. Disable the MAC transmitter and MAC receiver by clearing Bit (RE) and Bit 1(TE) of the MAC_Configuration Register.
4. Disable the Receive DMA (if applicable), after making sure that the data in the Rx FIFO is transferred to the system memory (by reading the appropriate bits of MTL_TxQ0_Debug Register, PRXQ=0 and RXQSTS=00).
5. Make sure that both Tx Queue and Rx Queue are empty (TXQSTS is 0 in MTL_TxQ0_Debug Register and RXQSTS is 0 in MTL_RxQ0_Debug Register).
6. To restart the operation, first start the DMAs, and then enable the MAC Transmitter and Receiver.

43.5.6 Programming Guidelines for Multi-Channel Multi-Queuing

The Transmit and Receive operation guidelines when using Multi channel/Multi-Queuing mode is detailed below

43.5.6.1 Transmit

Following these guidelines for transmitting.

1. Program the Transmit queue size in the TQS field of MTL_TxQ[n]_Operation_Mode register. Based on the value programmed in the TQS field, the size of the queue is determined. In the Transmit operation, the number of channels is equal to the number of the queues. Due to this reason, the Channel-to-Queue mapping is fixed.
2. For a queue to be used, the queue needs to be enabled in TXQEN in the corresponding MTL_TxQ[n]_Operation_Mode Register. the ST bit of DMA_CH[n]_Tx_Control Register and corresponding TXQEN in MTL_TxQ[n]_Operation Mode Register needs to be enabled.
3. The scheduling method needs to be programmed in SCHALG of MTL_Operation_Mode register.

43.5.6.2 Receive

Follow these guidelines for receiving:

1. Program the Receive queue size in the RQS field of MTL_RxQ[n]_Operation_Mode Register. Based on the value programmed in RQS field, the size of the queue is determined.
2. Enable the Receive Queues 0 to 1 in the fields RXQ0EN to RXQ1EN in MAC_RxQ_Ctrl0 Register SR bit of statically or dynamically mapped DMA_CH[n]_Rx_Control Register and corresponding RXQ[n]_EN in MAC_RxQ_Ctrl0 Register needs to be enabled.
3. The MAC routes the Rx packets to the Rx Queues based on following packet types:
   a. VLAN Tag Priority field in VLAN Tagged packets: Program PSRQ1-0 of the MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 Register for the routing of tagged packets based on the USP (user Priority) field of the received packets to the Rx Queues 0 to 1.

Note

The priorities set in PSRQ1-0 should be unique.

4. If multiple RX DMA channels are enabled, the following programming should be done for proper arbi- tration and mapping:
   a. Program the RAA field of MTL_Operation_Mode register to select the arbitration algorithm to decide which RxQ is read out from the RxFIFO memory.
   b. Program the MTL_RxQ[n]_Control to decide the weights and the packet arbitration for each RxQ.
   c. If static mapping is programmed in MTL_RxQ_DMA_Map[n] register (RXQ[n]DADMACH is reset to 0), bits RXQx2DMA and others need to be programmed to select the channel for which each queue is mapped.
   d. Set RXQ[n]DADMACH bit in MTL_RxQ_DMA_Map0 Register to select dynamic mapping of packets in each RxQueue.
   e. In dynamic channel mapping, the routing of a packet to a specific RxDMA channel is decided by the value of DCS field in the lowest MAC Address Register.

43.5.6.3 Programming Guidelines for Recovering from DMA Channel Failure

When the DMA channel issues a bus error, follow these steps to recover from the failure.

43.5.6.3.1 Recovering from the Receive DMA Channel Failure

Follow these steps if you get bus error in the Receive DMA channel:

1. Set the RPF bit to 1. This flushes all the packets one after the other. This step is optional. However, setting this bit prevents HOL (head-of-line) blocking in the Rx queues when packets sent to the RXDMA are stopped due to bus error.
2. Re-program the specific Registers of the DMA channel.
3. Start the DMA channel

43.5.6.3.2 Recovering from the Transmit DMA Channel Failure

Follow these steps:

1. Stop the specific DMA channel, even if it is in active state.
2. Flush the corresponding MTL queue.
3. Re-program the specific Registers of the DMA channel.
4. Start the DMA channel.

43.5.6.4 Programming Guidelines for IEEE 1588 Timestamping

You can enable the timestamp feature by setting Bit 0 of the MAC_Timestamp_Control Register. However, it is essential that the timestamp counter be initialized after this bit is set. Complete the following steps during Ethernet module initialization:

1. Mask the Timestamp Trigger interrupt by clearing the bit 16 of MAC_Interrupt_Enable Register.
2. Set Bit 0 of MAC_Timestamp_Control Register to enable timestamping.
3. Program MAC_Sub_Second_Increment Register based on the PTP clock frequency.
4. If you are using the Fine Correction approach, program MAC_Timestamp_Addend and set Bit 5 of MAC_Timestamp_Control Register.
5. Poll the MAC_Timestamp_Control Register until Bit 5 is cleared.
6. Program Bit 1 of MAC_Timestamp_Control Register to select the Fine Update method (if required).
7. Program MAC_System_Time_Seconds_Update Register and MAC_System_Time_Nanoseconds_Up- date Register with the appropriate time value.
8. Set Bit 2 in MAC_Timestamp_Control Register. The timestamp counter starts operation as soon as it is initialized with the value written in the Time- stamp Update registers. If one-step timestamping is enabled.
   a. To enable one-step timestamping, program Bit 27 of the TDES3 Context Descriptor.
   b. Program registers MAC_Timestamp_Ingress_Asym_Corr and MAC_Time- stamp_Egress_Asym_Corr to update the correction field in PDelay_Req PTP messages.
9. Enable the MAC receiver and transmitter for proper timestamping.

Note

If timestamp operation is disabled by clearing Bit 0 of MAC_Timestamp_Control Register, repeat all these steps to restart the timestamp operation.

43.5.6.4.1 Initialization Guidelines for System Time Generation

You can enable the timestamp feature by setting Bit 0 of the MAC_Timestamp_Control Register. However, it is essential that the timestamp counter be initialized after this bit is set. Complete the following steps during Ethernet module initialization:

1. Mask the Timestamp Trigger interrupt by clearing the bit 16 of MAC_Interrupt_Enable Register.
2. Set Bit 0 of MAC_Timestamp_Control Register to enable timestamping.
3. Program MAC_Sub_Second_Increment Register based on the PTP clock frequency.
4. If you are using the Fine Correction approach, program MAC_Timestamp_Addend and set Bit 5 of MAC_Timestamp_Control Register.
5. Poll the MAC_Timestamp_Control Register until Bit 5 is cleared.
6. Program Bit 1 of MAC_Timestamp_Control Register to select the Fine Update method (if required).
7. Program MAC_System_Time_Seconds_Update Register and MAC_System_Time_Nanoseconds_Up- date Register with the appropriate time value.
8. Set Bit 2 in MAC_Timestamp_Control Register. The timestamp counter starts operation as soon as it is initialized with the value written in the Time- stamp Update registers.
   If one-step timestamping is enabled
   a. To enable one-step timestamping, program Bit 27 of the TDES3 Context Descriptor.
   b. Program registers MAC_Timestamp_Ingress_Asym_Corr and MAC_Time- stamp_Egress_Asym_Corr to update the correction field in PDelay_Req PTP messages.
9. Enable the MAC receiver and transmitter for proper timestamping.

43.5.6.4.2 System Time Correction

Time correction can be done in Coarse or Fine methods.

43.5.6.4.2.1 Coarse Correction Method

To synchronize or update the system time in one process (coarse correction method), complete the following steps:

1. Set the offset (positive or negative) in the Timestamp Update registers(MAC_System_Time_Seconds_Update and MAC_System_Time_Nanoseconds_Update).
2. Set Bit 3 (TSUPDT) of MAC_Timestamp_Control Register. The value in the Timestamp Update registers is added to or subtracted from the system time when the TSUPDT bit is cleared.

43.5.6.4.2.2 Fine Correction Method

To synchronize or update the system time to reduce system-time jitter (fine correction method), complete the following steps:

- With the help of the algorithm explained in“System Time Register Module” calculate the rate by which you want to make the system time increments slower or faster.
- Update the MAC_Timestamp_Addend with the new value and set Bit 5 of the MAC_Timestamp_Con- trol Register.
- Wait for the time for which you want the new value of the Addend register to be active. You can do this by enabling the Timestamp Trigger interrupt after the system time reaches the target value.
- Program the required target time in MAC_PPS[n]_Target_Time_Seconds Register and MAC_PPS[n]_Target_Time_Nanoseconds Register.
- Enable the Timestamp interrupt in bit 12 of MAC_Interrupt_Enable register.
- Set bit 4 in Register MAC_Timestamp_Control.
- When this trigger causes an interrupt, read MAC_Interrupt_Status Register.
- Reprogram MAC_Timestamp_Addend Register with the old value and set bit 5 again

43.5.6.5 Programming Guidelines for Energy Efficient Ethernet

After you enable the Energy Efficient Ethernet (EEE) in the Ethernet module controller, you can program the following:

- Entering and Exiting Tx LPI mode during the QoS initialization
- Gating off the CSR clock in the Rx LPI mode
- Gating off the CSR clock in the Tx LPI mode

43.5.6.5.1 Entering and Exiting the Tx LPI Mode

Energy Efficient Ethernet (EEE) enables the IEEE 802.3 Media Access Control (MAC) sublayer along with a family of physical layers to operate in the Low-Power Idle (LPI) mode. In the Transmit path, the software must set the LPIEN bit of the MAC_LPI_Control_Status register to indicate to the MAC to stop transmission and initiate the LPI protocol.

Complete the following steps during QoS core initialization:

1. Read the PHY register through the MDIO interface, check if the remote end has the EEE capability, and then negotiate the timer values.
2. Program the PHY registers through the MDIO interface (including the RX_CLK_stoppable bit that indicates to the PHY whether to stop Rx clock in LPI mode.)
3. Program bits[25:16] and bits[15:0] in MAC_LPI_Timers_Control Register.
4. Read the link status of the PHY chip by using the MDIO interface and update bit 17 of MAC_LPI_Control_Status accordingly. This update should be done whenever the link status in the PHY chip changes
5. Program the MAC_1US_Tic_Counter as per the frequency of the clock used for accessing the CSR slave port.
6. Program the MAC_LPI_Entry_Timer register (LPIET) with the IDLE time for which the MAC should wait before entering the LPI state on its own.

7. Set LPITE and LPITXA (bit[20:19]) of MAC_LPI_Control_Status register to enable the auto-entry into LPI and auto-exit of MAC from LPI state.
8. Program the MAC_1US_Tic_Counter as per the frequency of the clock used for accessing the CSR slave port.
9. Program the MAC_LPI_Entry_Timer register (LPIET) with the IDLE time for which the MAC should wait before entering the LPI state on its own.
10. Set LPITE and LPITXA (bit[20:19]) of MAC_LPI_Control_Status register to enable the auto-entry into LPI and auto-exit of MAC from LPI state.
11. Set bit 16 of MAC_LPI_Control_Status Register to make the MAC Transmitter enter the LPI state. The MAC enters the LPI mode after completing all scheduled packets and remains IDLE for the time indicated by LPIET. It sets the TLPIEN (bit[0]) after entry to LPI state.
12. When a packet is scheduled for transmission (when the TxDMA comes out of IDLE state or when a packet is presented at ATI or MTI interface), the MAC Transmitter exits LPI state automatically. It waits for TWT time before setting the TLPIEX interrupt status bit and then resume the packet transmission.
13. MAC Transmitter re-enters LPI state if it remains IDLE for LPIET time and sets the TLPIEN bit and the entry-exit cycle continues.
14. Reset LPITXEN in case the application wants to over-ride the auto-entry/exit modes and make the MAC Transmitter exit the LPI state directly.

Note

To make the MAC enter the LPI state only after it completes the transmission of all queued frames in the Tx FIFO, you should set bit 19 in MAC_LPI_Control_Status Register.

To switch off the MII transmit clock during the LPI state, use the sbd_tx_clk_gating_ctrl_o signal for gating the clock input.

To switch off the CSR clock or power to the rest of the system during the LPI state, you should wait for the TLPIEN interrupt of MAC_LPI_Control_Status Register to be generated. Restore the clocks before performing step 6 when you want to come out of the LPI state.

43.5.6.5.2 Gating Off the CSR Clock in the LPI Mode

The Transmit MII Clock can be gated off in the module save the power when the MAC is in Low-Power Idle (LPI) mode. Setting the LPITCSE in MAC_LPI_Control_Status register will enable this feature. This will not be applicable to RMII mode since the Transmit clock is needed for transmitting the LPI Pattern.

43.5.6.5.3 Rx LPI Mode

The following operations can be performed when the MAC receives the LPI pattern from the PHY:

1. The MAC RX enters the LPI mode and the Rx LPI entry interrupt status [RLPIEN interrupt of MAC_LPI_Control_Status Register] is set.
2. The interrupt pin (sbd_intr_o) is asserted. The sbd_intr_o interrupt is cleared when the host reads the MAC_LPI_Control_Status Register.

After the sbd_intr_o interrupt is asserted if the MAC TX is not in the LPI mode when you gate off the CSR clock, the events on the MAC transmitter do not get reported or updated in the CSR.

wait for the LPI exit indication from the PHY after which the MAC asserts the LPI exit interrupt on lpi_intr_o (synchronous to clk_rx_i). The lpi_intr_o interrupt is cleared when MAC_LPI_Control_Status Register is read.

43.5.6.5.4 Gating Off the CSR Clock in the Tx LPI Mode

The following operations are performed when Bit 16 (LPIEN) of MAC_LPI_Control_Status is set:

1. The MAC RX enters the LPI mode and the Rx LPI entry interrupt status [RLPIEN interrupt of MAC_LPI_Control_Status Register] is set.
2. The interrupt pin (sbd_intr_o) is asserted. The sbd_intr_o interrupt is cleared when the host reads the MAC_LPI_Control_Status Register.

After the sbd_intr_o interrupt is asserted if the MAC TX is not in the LPI mode when you gate off the CSR clock, the events on the MAC transmitter do not get reported.

For restoring the Tx clock, wait for the LPI exit indication from the PHY after which the MAC asserts the LPI exit interrupt on lpi_intr_o (synchronous to clk_rx_i). The lpi_intr_o interrupt is cleared when MAC_LPI_Control_Status Register is read.

43.5.6.6 Programming Guidelines for Flexible Pulse-Per-Second Output

After you enable the Flexible Pulse-Per-Second Output feature in the Ethernet module controller, you can perform the following tasks:

- Generating Single Pulse on PPS
- Generating Next Pulse on PPS
- Generating a Pulse Train on PPS
- Generating an Interrupt without Affecting the PPS

43.5.6.6.1 Generating Single Pulse on PPS

To programsingle pulse, follow this procedure:

1. Program 11 or 10 (for interrupt) in Bits [6:5], TRGTMODSEL, of MAC_PPS_Control Register. This instructs the MAC to use the Target Time registers (MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds) for start time of PPS signal output.
2. Program the start time value in the Target Time registers (MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds).
3. Program the width of the PPS signal output in MAC_PPS(#i)_Width (for i=0; i<= 1) Register.
4. Program Bits [3:0], PPSCMD, of MAC_PPS_Control to 0001. This instructs the MAC to generate single pulse on the PPS signal output at the time programmed in the Target Time registers

43.5.6.6.2 Generating Next Pulse on PPS

When the PPSCMD is executed (PPSCMD bits = 0), you can cancel the pulse generation by giving the Cancel Start Command (PPSCMD=0011) before the programmed start time elapses. You can also program the behavior of the next pulse in advance.

To program the next pulse, follow this procedure:

1. Program the start time for the next pulse in the Target Time registers. This time should be more than the time at which the falling edge occurs for the previous pulse.
2. Program the width of the next PPS signal output in MAC_PPS (#i)_Width (for i=0; i<=1) Register.
3. Program Bits [3:0], PPSCMD, of MAC_PPS_Control to generate a single pulse after the time at which the previous pulse is de-asserted. This instructs the MAC to generate single pulse on the PPS signal output, at the time programmed in Target Time registers.
   If you give this command before the previous pulse becomes low, then the new command overwrites the previous command and the QOS may generate only 1 extended pulse.

43.5.6.6.3 Generating a Pulse Train on PPS

Complete the following steps to generate a pulse train on PPS:

1. Program 11 or 10 (for interrupt) in Bits [6:5], TRGTMODSEL, of MAC_PPS_Control Register. This instructs the MAC to use the Target Time registers for start time of the PPS signal output.
2. Program the start time value in the Target Time registers.
3. Program the interval value between the train of pulses on the PPS signal output in MAC_PPS (#i)_Width (for i=0;i <=1) Register.
4. Program the width of the PPS system in MAC_PPS (#i)_interval (for i=0;i <=1) Register.
5. Program Bits[3:0], PPSCMD, of MAC_PPS_Control Register to 0010. This instructs the MAC to generate train of pulses on the PPS signal output with start time programmed in Target Time registers.
6. Program the stop value in the Target Time registers. Ensure that Bit 31 (TSTRBUSY) of MAC_PPS(#i)_Target_Time_Nanoseconds (for i=0; i <=1) Register s reset before programming the Target Time registers again. .
7. Program the PPSCMD field (bit 3:0) of MAC_PPS_Control to 0100. This stops the train of pulses on PPS signal output after the programmed stop time specified in Step 6 elapses.

You can stop the pulse train at any time by programming 0101 in the PPSCMD field. Similarly, you can cancel the Stop Pulse train command (given in Step 7) by programming 0110 in the PPSCMD field before the time (programmed in Step 6) elapses. You can cancel the pulse train generation by programming 0011 in the PPSCMD field before the programmed start time (in Step 2) elapses.

43.5.6.6.4 Generating an Interrupt without Affecting the PPS

The Bits [6:5], TRGTMODSEL, of the MAC_PPS_Control Register enable you to program the Target Time registers to do any one of the following:

- Generate only interrupts.
- Generate interrupts and the PPS start and stop time.
- Generate only PPS start and stop time.

Complete the following steps to program the Target Time registers to generate only interrupt event:

1. Program 00 (for interrupt) in Bits [6:5], TRGTMODSEL, of MAC_PPS_Control Register. This instructs the MAC to use the Target Time registers for target time interrupt.
2. Program a target time value in the Target Time registers. This instructs the MAC to generate an inter- rupt when the target time elapses.

If Bits [6:5], TRGTMODSEL, are changed (for example, to control the PPS), then the interrupt generation is over-written with the new mode and new programmed Target Time register value.

43.5.6.7 Programming Guidelines for TSO

The TCP Segmentation Offload (TSO) engine is used to offload the TCP segmentation functions to the hardware. To program the TSO, set the TSE bit to enable TCP packet segmentation, and program descriptor fields to enable TSO for the current packet.

Complete the following steps to program TSO:

1. Program the TSE bit of corresponding DMA_CH[n]_Tx_Control register to enable TCP packet segmentation in that DMA.
2. In addition to the normal transfer descriptor setting, the following descriptor fields must be programmed to enable TSO for the current packet:
   a. Enable TSE in Bit 18 of TDES3.
   b. Program the length of the un-segmented TCP/IP packet payload in bits [17:0] of TDES3 and the TCP header in bits [22:19] of TDES3.
   c. Program the maximum size of the segment in MSS of DMA_CH[n]_Control register or MSS in the context descriptor. If MSS field is programmed in both DMA_CH[n]_Control register and in the context descriptor, the latest software programmed sequence is considered.
3. The header of the unsegmented TCP/IP packet should be in Buffer 1 of the first descriptor and this buffer must not hold any payload bytes. The payload is allocated to Buffer 2 and the buffers of the subsequent descriptors.

Note

If TSE is enabled in TDES3 for a non-TCP-IP packet, the result is unpredictable.

43.6 Software

43.6.1 ETHERNET Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/ethernet

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

43.6.1.1 Ethernet + IPC basic message passing example with interrupt - C28X_CM

FILE: ethernet_ipc_ex1_basic_c28x1.c

This example demonstrates how to configure IPC and pass information from C28x to CM core without message queues. This configures the Pinmux for Ethernet Prepares the Ethernet frame that is sent to the CM core over IPC Message RAM. Uses the IPC command interface to signal the IPC Command, Packet address, Packet Length which is used by CM side code to send it on the Ethernet Line, which is acknowledged by the CM core side code over IPC on succesfully receiving a packet It is recommended to run the C28x1 core first, followed by the CM core.

External Connections
- Connections for Ethernet in MII mode

Watch Variables
- pass

43.6.1.2 Ethernet + IPC basic message passing example with interrupt - C28X_CM

FILE: ethernet_ipc_ex1_basic_cm.c

This example demonstrates how to receive the message passed from C28x side containing the Ethernet Packet over IPC,sends the packet on Ethernet acknowledge the packet received over IPC to C28x core. This extends the IPC Example. Uses IPC C28x to CM core without message queues. It can be used as a reference for flow using Ethernet and IPC together. If the Packet data received over Ethernet is to be passed to C28x for control applications, it can send IPC message to C28x. It is recommended to run the C28x1 core first, followed by the CM core. The example actually uses the internal loopback mode of MAC hence the MII Tx and Rx signals are not used, but needs the MII Tx and Rx Clock signals which comes from the External PHY.

External Connections
- MII connections on Control card

Watch Variables
- None.

43.6.1.3 Ethernet MAC Internal Loopback - CM

FILE: ethernet_ex1_basic_tx_rx_loopback.c

This example demonstrates the steps to be followed in using the Ethernet of the Communication Manager Subsystem to initialize the Ethernet module and Configure the module in MAC Loop back mode Prepares a packet to be sent, Sends the packet and reads the statistics to check if the packet is received by the module Before running this Communication Manager code the C28x cpu1 code has to be run to configure the clocks to Communication manager and required IO pads for Ethernet module

External Connections
This example programs the Ethernet module in Internal Loop back mode and hence needs no external connection on the MII Data lines. But it needs the MII Tx and Rx clocks to be input on those pins Refer to the C28x CPU1 code of ethernet_config_c28x project for which GPIOs are used for connecting to the PHY

Watch Variables
- None

43.6.1.4 Ethernet Basic Transmit and Receive PHY Loopback - CM

FILE: ethernet_ex2_phy_loopback.c

This example demonstrates the steps to be followed in using the Ethernet of the Communication Manager Subsystem to initialize the Ethernet module and Configure the module in External Loop back mode the packet is looped back at external PHY. Prepares a packet to be sent, Sends the packet and reads the staticstics to check if the packet is received by the module Before running this Communication Manager code the C28x cpu1 code has to be run to configure the clocks to Communication manager and required IO pads for Ethernet module

External Connections
This example programs the Ethernet module in External Loop back mode (at PHY) and hence needs external connection to the PHY on the MII interface and also the MDIO Pins connected to the PHY. This example assumes DP83822 PHY for the PHY configurations if a different PHY is used the sequences might change Refer to the C28x CPU1 code of ethernet_config_c28x project for which GPIOs are used for connecting to the PHY

Watch Variables
- phyRegContent variable can be checked to know if PHY register read,write is working correctly
  - stats to know if the packet is received correctly after loopback at PHY side

43.6.1.5 Ethernet Threshold mode with level PHY loopback - CM

FILE: ethernet_ex3_threshold_mode_phy_loopback.c

This example demonstrates the steps to be followed in using the Ethernet of the Communication Manager Subsystem to initialize the Ethernet module in Threshold mode It Configures the module in MII External Loop back mode in which the packet is looped back at external PHY. Prepares a packet to be sent, Sends the packet and reads the staticstics to check if the packet is received by the module It configures the Transmit and Receive queues of Ethernet DMA in Threshold mode. The Transmit threshold mode generates early transmit interrupts when each buffer is transmitted from the memory into the transmit FIFO. The Buffer can be reclaimed by the application. The Receive threshold mode generates early receive interrupts where the module generates Early receive interrupts when programmed threshold buffer size is transferred into receive buffer memory from the receive FIFO. This will be of use when dealing with Time critical receive paths where the application can consume the packets at programmed burstLength unlike the conventional Store and Forward mode(Default) where the module generates the interrupts when the complete packet is transmitted on the Line and when the complete packet is received and checksum validation is succesful This example provides a starting point for configuring the threshold mode. Refer to the Driver API guide for the different callbacks available in the driver. It uses the example Interrupt Service Routines provided in the driver library. Users can modify those ISRs or write another as per their need. Before running this Communication Manager code the C28x cpu1 code has to be run to configure the clocks to Communication manager and required IO pads for Ethernet module

External Connections
This example programs the Ethernet module in External Loop back mode (at PHY) and hence needs external connection to the PHY on the MII interface and also the MDIO Pins connected to the PHY. This example assumes DP83822 PHY for the PHY configurations if a different PHY is used the sequences might change. Refer to the C28x CPU1 code of ethernet_config_c28x project for which GPIOs are used for connecting to the PHY

Watch Variables
- phyRegContent variable can be checked to know if PHY register read,write is working correctly
- Ethernet_txInterruptCount - this variable available in driver library provides a count of number of times Transmit completion interrupt (on full packet transmission onto the line) occured
- Ethernet_rxInterruptCount - this variable provides a count of number of times the complete Receive completion interrupt occured
- Ethernet_earlyRxInterruptCount - the number of times the early receive interrupt occured. This depends on the burstLength configured
- Ethernet_earlyTxInterruptCount - the number of times the early transmit interrupt occured. This depends on the number of fragments of the packets transmitted

43.6.1.6 Ethernet PTP Basic Master - CM

FILE: ethernet_ex4_ptp_basic_master.c

This example configures the device in IEEE PTPv2 Master mode and then periodically sends Sync packets to the slave. On receiving the DelayReq packets from the slave, the master also sends out the DelayResp packets.

External Connections
This example programs the Ethernet module in PTP Basic Master mode. The example project Ethernet PTP Basic Slave is intended to be used along with this project to see the whole PTP Protocol state in action. The second device is configured as Slave and both devices in conjunction exchange Sync, DelayReq and DelayResp packets.

Refer to the C28x CPU1 code of ethernet_config_c28x project for configuring the PTP clock that drives the system time counter on the Ethernet module.

Watch Variables
- gPtpMasterState

43.6.1.7 Ethernet PTP Basic Slave - CM

FILE: ethernet_ex4_ptp_basic_slave.c

This example configures the device in IEEE PTPv2 Slave mode and then waits for the Sync packets from the slave. On receiving configurable number of Sync packets from the master, the slave also sends out the DelayReq packets. In response to the DelayReq packets, the Master also sends out the DelayResp packets which is then received by the Slave. The slave parallely maintains the internal state of the PTP in terms of Master to Slave Delay and Slave to Master Delay. Consequently, the slave also calculates Offset From Master and Mean Path Delay.

External Connections
This example programs the Ethernet module in PTP Basic Slave mode. The example project Ethernet PTP Basic Master is intended to be used along with this project to see the whole PTP Protocol state in action. The second device is configured as Master and both devices in conjunction exchange Sync, DelayReq and DelayResp packets.

Refer to the C28x CPU1 code of ethernet_config_c28x project for configuring the PTP clock that drives the system time counter on the Ethernet module.

Watch Variables
- gPtpSlaveState

43.6.1.8 Ethernet PTP Offload Master - CM

FILE: ethernet_ex5_ptp_offload_master.c

This example configures the device in IEEE PTPv2 Master mode and sets the options that are needed by the offload engine to operate such as the domainNumber, LogSyncInterval among others. After that it enables sending SYNC messages periodically according to the interval already set previously.

External Connections
This example programs the Ethernet module in PTP Offload Master mode. The example project Ethernet PTP Offload Slave is intended to be used along with this project to see the whole PTP Offload engine in action. The second device is configured as Slave and both devices in conjunction exchange Sync, DelayReq and DelayResp packets.

Refer to the C28x CPU1 code of ethernet_config_c28x project for configuring the PTP clock that drives the system time counter on the Ethernet module.

Watch Variables
- Ethernet_ptpDelayReqPktCount

43.6.1.9 Ethernet PTP Offload Slave - CM

FILE: ethernet_ex5_ptp_offload_slave.c

This example configures the device in IEEE PTPv2 Slave mode and sets the options that are needed by the offload engine to operate such as the domainNumber, LogSyncInterval among others. After that it enables sending DelayReq messages periodically for every configurable number of Sync packets.

External Connections
This example programs the Ethernet module in PTP Offload Slave mode. The example project Ethernet PTP Offload Master is intended to be used along with this project to see the whole PTP Offload engine in action. The second device is configured as Slave and both devices in conjunction exchange Sync, DelayReq and DelayResp packets.

Refer to the C28x CPU1 code of ethernet_config_c28x project for configuring the PTP clock that drives the system time counter on the Ethernet module.

Watch Variables
- Ethernet_ptpSyncPktCount
- Ethernet_ptpDelayRespPktCount

43.6.1.10 Ethernet MAC CRC and Checksum Offload - CM

FILE: ethernet_ex6_crc_checksum_offload.c

This example demonstrates the steps to be followed in using the Ethernet of the Communication Manager Subsystem to initialize the Ethernet module and Configure the module in MAC Loop back mode. Demonstrates how to program the CRC offload and Checksum Offload(IP Checksum) Prepares a packet to be sent, Sends the packet and reads the statistics to check if the packet is received by the module Before running this Communication Manager code the C28x cpu1 code has to be run to configure the clocks to Communication manager and required IO pads for Ethernet module

External Connections
This example programs the Ethernet module in Internal Loop back mode and hence needs no external connection on the MII Data lines. But it needs the MII Tx and Rx clocks to be input on those pins Refer to the C28x CPU1 code of ethernet_config_c28x project for which GPIOs are used for connecting to the PHY

Watch Variables
- None

43.6.1.11 Ethernet Transmit Segmentation Offload - CM

FILE: ethernet_ex7_tranport_segmentation_offload.c

This example demonstrates the steps to be followed in using the Ethernet of the Communication Manager Subsystem to initialize the Ethernet module and Configure the module in MAC Loop back mode. Demonstrates how to program the Transport Segmentation Offload feature in the Low level driver which in turn programs the feature in the hardware The Hardware segments a single packet into configured segment size Prepares a packet to be sent, Sends the packet and reads the statistics to check if the packet is received by the module Before running this Communication Manager code the C28x cpu1 code has to be run to configure the clocks to Communication manager and required IO pads for Ethernet module

External Connections
This example programs the Ethernet module in Internal Loop back mode and hence needs no external connection on the MII Data lines. But it needs the MII Tx and Rx clocks to be input on those pins Refer to the C28x CPU1 code of ethernet_config_c28x project for which GPIOs are used for connecting to the PHY

Watch Variables
- None

43.6.1.12 Ethernet MAC Internal Loopback - CM

FILE: ethernet_ex8_vlan_filtering.c

This example demonstrates the steps to be followed in using the Ethernet of the Communication Manager Subsystem to initialize the Ethernet module and Configure the module in MAC Loop back mode The packet sent is inserted with a VLAN tag. A VLAN filter is configured to route it to a different channel Prepares a packet to be sent, Sends the packet and reads the statistics to check if the packet is received by the module Before running this Communication Manager code the C28x cpu1 code has to be run to configure the clocks to Communication manager and required IO pads for Ethernet module

External Connections
This example programs the Ethernet module in Internal Loop back mode and hence needs no external connection on the MII Data lines. But it needs the MII Tx and Rx clocks to be input on those pins Refer to the C28x CPU1 code of ethernet_config_c28x project for which GPIOs are used for connecting to the PHY

Watch Variables
- None

43.6.1.13 Ethernet RevMII Example MII side - CM

FILE: ethernet_ex9_revmii_example_mii_remote.c

This example demonstrates the steps to be followed in using the Ethernet of the Communication Manager Subsystem to initialize the Ethernet module and Configure the module in MII mode. The other device is running a RevMII mode This demonstrates the sequence for MII - RevMII communication Before running this Communication Manager code the C28x cpu1 code has to be run to configure the clocks to Communication manager and required IO pads for Ethernet module This Example has been validated on an TI internal board and will not run on Control Card. This has been run with RevMII Example Remote MAC side. Once the RevMII mode is configured this appears like a PHY to the external MAC which is connected over MDIO. Even though there is no physical PHY the RevMII mode lets the remote MAC see this side as a MAC.

External Connections
TBD

Watch Variables
- None

43.6.1.14 Ethernet RevMII Example RevMII side - CM

FILE: ethernet_ex9_revmii_example_revmii_side.c

This example demonstrates the steps to be followed in using the Ethernet of the Communication Manager Subsystem to initialize the Ethernet module and Configure the module in RevMII mode. Before running this Communication Manager code the C28x cpu1 code has to be run to configure the clocks to Communication manager and required IO pads for Ethernet module This Example has been validated on an TI internal board and will not run on Control Card. This has been run with RevMII Example Remote MAC side. Once the RevMII mode is configured this appears like a PHY to the external MAC which is connected over MDIO. Even though there is no physical PHY the RevMII mode lets the remote MAC see this side as a PHY.

External Connections
TBD

Watch Variables
- None

43.6.1.15 Ethernet Low Latency Interrupt - CM

FILE: ethernet_ex10_lowlatency_interrupt.c

This example demonstrates the steps to be followed in using the Ethernet of the Communication Manager Subsystem to handle the Ethernet transmit,receive with minimum latency interrupts. The example interrupt handlers provided in the Ethernet driver help to achieve user friendly buffer management and the generic interrupt handler handles different interrupt sources, these factors might be more cycle consuming. This example demonstrates how to achieve lowest possible latency with interrupts and buffer management with the Ethernet

Driver. Before running this Communication Manager code the C28x cpu1 code has to be run to configure the clocks to Communication manager and required IO pads for Ethernet module

External Connections
This example programs the Ethernet module in Internal Loop back mode and hence needs no external connection on the MII Data lines. But it needs the MII Tx and Rx clocks to be input on those pins Refer to the C28x CPU1 code of ethernet_config_c28x project for which GPIOs are used for connecting to the PHY

Watch Variables
- genericISRCount
- transmitISRCount
- receiveISRCount

43.7 Ethernet Registers

This section describes the Ethernet Module Registers.

43.7.1 Ethernet Base Addresses

Table 43-76. EMAC Base Address Table (CM)

DriverLib Name	Base Address	uDMA Access	Ethernet DMA Access
EMAC_BASE	0x400C_0000	-	-
EMAC_SS_BASE	0x400C_2000	-	-

43.7.2 ETHERNETSS_REGS Registers

Table 43-77 lists the memory-mapped registers for the ETHERNETSS_REGS registers. All register offset addresses not listed in Table 43-77 should be considered as reserved locations and the register contents should not be modified.

Table 43-77. ETHERNETSS_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	ETHERNETSS_IPREVNUM	IP Revision Number		Go
4h	ETHERNETSS_CTRLSTS	Control Register	LOCK	Go
8h	ETHERNETSS_PTPTSTRIGSEL0	PTP Trigger-0 select	LOCK	Go
Ch	ETHERNETSS_PTPTSTRIGSEL1	PTP Trigger-1 select	LOCK	Go
10h	ETHERNETSS_PTPTSSWTRIG0	PTP SW Trigger-0		Go
14h	ETHERNETSS_PTPTSSWTRIG1	PTP SW Trigger-1		Go
18h	ETHERNETSS_PTPPPSR0	PTP PPS-0 Read		Go
1Ch	ETHERNETSS_PTPPPSR1	PTP PPS-1 Read		Go
20h	ETHERNETSS_PTP_TSRL	PTP timestamp read lower 32 bits		Go
24h	ETHERNETSS_PTP_TSRH	PTP timestamp read upper 32 bits		Go
28h	ETHERNETSS_PTP_TSWL	External Timestamp write lower 32 bits		Go
2Ch	ETHERNETSS_PTP_TSWH	External Timestamp write upper 32 bits		Go
30h	ETHERNETSS_REVMII_CTRL	RevMII Phy Address controls	LOCK	Go

Complex bit access types are encoded to fit into small table cells. Table 43-78 shows the codes that are used for access types in this section.

Table 43-78. ETHERNETSS_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
RC	R C	Read to Clear
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 43-78. ETHERNETSS_REGS Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

43.7.2.1 ETHERNETSS_IPREVNUM Register (Offset = 0h) [Reset = 0h]

ETHERNETSS_IPREVNUM is shown in Figure 43-27 and described in Table 43-79.

Return to the Summary Table.

IP Revision number showing the Major & Minor IP versions 4 bit each

Figure 43-27. ETHERNETSS_IPREVNUM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								IP_REV_MAJOR				IP_REV_MINOR			
R-0h								R-0h				R-0h			

Table 43-79. ETHERNETSS_IPREVNUM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-4	IP_REV_MAJOR	R	0h	Major IP Type increment is hardcoded reset value which increments to signify major change in IP behavior in terms of data/control flow or new feature addition. Reset type: CM.SYSRESETn
3-0	IP_REV_MINOR	R	0h	Reset value for this register is hardcoded and increments with minor changes to the IP those will not increment IP Type, but the bug fixes and changes impact behavior or software control than previous silicon version. Reset type: CM.SYSRESETn

43.7.2.2 ETHERNETSS_CTRLSTS Register (Offset = 4h) [Reset = 0h]

ETHERNETSS_CTRLSTS is shown in Figure 43-28 and described in Table 43-80.

Return to the Summary Table.

PHY Type, clock source type select

Figure 43-28. ETHERNETSS_CTRLSTS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
WRITE_KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED						FLOW_CTRL_EN	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLK_SRC_SEL	RESERVED		CLK_LM	RESERVED	PHY_INTF_SEL		
R/W-0h	R-0h		R-0-0h	R-0h	R/W-0h		

Table 43-80. ETHERNETSS_CTRLSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	WRITE_KEY	R-0/W	0h	The key value should be 0xa5 for the writes to this register to take effect. Writes to other controls in register with value in this field other than specified will be ignored. Reset type: XRSn
15-10	RESERVED	R	0h	Reserved
9-8	FLOW_CTRL_EN	R/W	0h	Hardware flow control enable per Rx Queue 8 : HW flow control enable For Queue-0 9 : HW flow control enable For Queue-1 If 0: HW Flow control is not enabled. 1: HW Flow control is enabled with Pause Packet transmission in Full-duplex & back-pressure in half duplex mode. Reset type: XRSn
7	CLK_SRC_SEL	R/W	0h	To select internal clock source in RMII mode and to enable clocking in REVMII mode. 0: External clock source drives clock through pin. 1: Internal source drives the clock Reset type: XRSn
6-5	RESERVED	R	0h	Reserved
4	CLK_LM	R-0	0h	Clock Select from internal source for internal loopback in MII/RMII mode without PHY 0: Normal Mode 1: Clocks enabled with internal source Reset type: XRSn
3	RESERVED	R	0h	Reserved

Table 43-80. ETHERNETSS_CTRLSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	PHY_INTF_SEL	R/W	0h	PHY Interface Selection control sampled by EQOS only out of reset. Controls the clocking, activated IO paths etc. Following are the options. 000-GMII/MII, 001-RGMII (Reserved), 010-SGMII (Reserved), 011-TBI (Reserved), 100-RMII, 101-RTBI(Reserved), 110-SMII(Reserevd). 111-RevMII Configuring any reserved configuration shall default to "000" MII selection. Reset type: XRSn

43.7.2.3 ETHERNETSS_PTPTSTRIGSEL0 Register (Offset = 8h) [Reset = 0h]

ETHERNETSS_PTPTSTRIGSEL0 is shown in Figure 43-29 and described in Table 43-81.

Return to the Summary Table.

PTP Timestamp triggers, Trigger selects and PPS control

Figure 43-29. ETHERNETSS_PTPTSTRIGSEL0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
WRITE_KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			PTP_AUX_TS_TRIG_SEL0				
R-0h			R/W-0h				

Table 43-81. ETHERNETSS_PTPTSTRIGSEL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	WRITE_KEY	R-0/W	0h	The key value should be 0xa5 for the writes to this register to take effect. Writes to other controls in register with value in this field other than specified will be ignored. Reset type: CM.SYSRESETn
15-5	RESERVED	R	0h	Reserved
4-0	PTP_AUX_TS_TRIG_SEL0	R/W	0h	Acts as mux select for the trigger sources of the timestamp capture-0. 32 Configurations are allowed. Refer device spec. for the mux select options. Reset type: CM.SYSRESETn

43.7.2.4 ETHERNETSS_PTPTSTRIGSEL1 Register (Offset = Ch) [Reset = 0h]

ETHERNETSS_PTPTSTRIGSEL1 is shown in Figure 43-30 and described in Table 43-82.

Return to the Summary Table.

PTP Timestamp triggers, Trigger selects and PPS control

Figure 43-30. ETHERNETSS_PTPTSTRIGSEL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
WRITE_KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			PTP_AUX_TS_TRIG_SEL1				
R-0h			R/W-0h				

Table 43-82. ETHERNETSS_PTPTSTRIGSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	WRITE_KEY	R-0/W	0h	The key value should be 0xa5 for the writes to this register to take effect. Writes to other controls in register with value in this field other than specified will be ignored. Reset type: CM.SYSRESETn
15-5	RESERVED	R	0h	Reserved
4-0	PTP_AUX_TS_TRIG_SEL1	R/W	0h	Acts as mux select for the trigger sources of the timestamp capture-1. 32 Configurations are allowed. Refer device spec. for the mux select options. Reset type: CM.SYSRESETn

43.7.2.5 ETHERNETSS_PTPTSSWTRIG0 Register (Offset = 10h) [Reset = 0h]

ETHERNETSS_PTPTSSWTRIG0 is shown in Figure 43-31 and described in Table 43-83.

Return to the Summary Table.

PTP Timestamp triggers, Trigger selects and PPS control

Figure 43-31. ETHERNETSS_PTPTSSWTRIG0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							PTP_AUX_TS_SW_TRIG0
R-0h							R-0/W1S-0h

Table 43-83. ETHERNETSS_PTPTSSWTRIG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	PTP_AUX_TS_SW_TRIG0	R-0/W1S	0h	Software controlled independent trigger-0 for capturing Auxillary timestamps in Timestamp FIFO. These signals are muxed with O/p Cross-bar output, Mux-Select is controlled through PTP_AUX_TS_TRIG_SEL0. When not programmed software trigger is default selection. Register always reads 0 and when written 1 creates a trigger pulse to the time-stamp capture logic. Writing 0 has no effect. Reset type: CM.SYSRESETn

43.7.2.6 ETHERNETSS_PTPTSSWTRIG1 Register (Offset = 14h) [Reset = 0h]

ETHERNETSS_PTPTSSWTRIG1 is shown in Figure 43-32 and described in Table 43-84.

Return to the Summary Table.

PTP Timestamp triggers, Trigger selects and PPS control

Figure 43-32. ETHERNETSS_PTPTSSWTRIG1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							PTP_AUX_TS_SW_TRIG1
R-0h							R-0/W1S-0h

Table 43-84. ETHERNETSS_PTPTSSWTRIG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	PTP_AUX_TS_SW_TRIG1	R-0/W1S	0h	Software controlled independent trigger-1 for capturing Auxillary timestamps in Timestamp FIFO. These signals are muxed with O/p Cross-bar output, Mux-Select is controlled through PTP_AUX_TS_TRIG_SEL1. When not programmed software trigger is default selection. Register always reads 0 and when written 1 creates a trigger pulse to the time-stamp capture logic. Writing 0 has no effect. Reset type: CM.SYSRESETn

43.7.2.7 ETHERNETSS_PTPPPSR0 Register (Offset = 18h) [Reset = 0h]

ETHERNETSS_PTPPPSR0 is shown in Figure 43-33 and described in Table 43-85.

Return to the Summary Table.

PTP Timestamp triggers, Trigger selects and PPS control

Figure 43-33. ETHERNETSS_PTPPPSR0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							PTP_PPS_R0
R-0h							RC-0h

Table 43-85. ETHERNETSS_PTPPPSR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	PTP_PPS_R0	RC	0h	Registered value of pulse per second-0, the register is cleared upon read and rearmed for next pulse. SW can track if the PPS is elapsed by reading this register. Setting of this register from internal has higher priority over the clear from read if the events were to happen together, so that next event is not lost. Reset type: CM.SYSRESETn

43.7.2.8 ETHERNETSS_PTPPPSR1 Register (Offset = 1Ch) [Reset = 0h]

ETHERNETSS_PTPPPSR1 is shown in Figure 43-34 and described in Table 43-86.

Return to the Summary Table.

PTP Timestamp triggers, Trigger selects and PPS control

Figure 43-34. ETHERNETSS_PTPPPSR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							PTP_PPS_R1
R-0h							RC-0h

Table 43-86. ETHERNETSS_PTPPPSR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	PTP_PPS_R1	RC	0h	Registered value of pulse per second-1, the register is cleared upon read and rearmed for next pulse. SW can track if the PPS is elapsed by reading this register. Setting of this register from internal has higher priority over the clear from read if the events were to happen together, so that next event is not lost. Reset type: CM.SYSRESETn

43.7.2.9 ETHERNETSS_PTP_TSRL Register (Offset = 20h) [Reset = 0h]

ETHERNETSS_PTP_TSRL is shown in Figure 43-35 and described in Table 43-87.

Return to the Summary Table.

PTP timestamp read lower 32 bits

Figure 43-35. ETHERNETSS_PTP_TSRL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSR_L																															
R-0h																															

Table 43-87. ETHERNETSS_PTP_TSRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TSR_L	R	0h	Lower 32 bit time stamp value as captured from the ptp_timestamp_o[31:0] output of EQOS. The value synchronsed to hclk_i domain. The SW needs to ensure correctness between High & low value by reading high value first followed by low and high again to check for rollover. Reset type: CM.SYSRESETn

43.7.2.10 ETHERNETSS_PTP_TSRH Register (Offset = 24h) [Reset = 0h]

ETHERNETSS_PTP_TSRH is shown in Figure 43-36 and described in Table 43-88.

Return to the Summary Table.

PTP timestamp read upper 32 bits

Figure 43-36. ETHERNETSS_PTP_TSRH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSR_H																															
R-0h																															

Table 43-88. ETHERNETSS_PTP_TSRH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TSR_H	R	0h	Lower 32 bit time stamp value as captured from the ptp_timestamp_o[63:32] output of EQOS. The value synchronsed to hclk_i domain. The SW needs to ensure correctness between High & low value by reading high value first followed by low and high again to check for rollover. Reset type: CM.SYSRESETn

43.7.2.11 ETHERNETSS_PTP_TSWL Register (Offset = 28h) [Reset = 0h]

ETHERNETSS_PTP_TSWL is shown in Figure 43-37 and described in Table 43-89.

Return to the Summary Table.

External PTP Timestamp write lower 32 bits

Figure 43-37. ETHERNETSS_PTP_TSWL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSW_L																															
R/W-0h																															

Table 43-89. ETHERNETSS_PTP_TSWL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TSW_L	R/W	0h	Lower 32 bit time stamp value input to ptp_timestamp_i[31:0] to be embedded in next packet. This could be value read from external source or in systems with RTC could be connected to RTC counter directly. Reset type: CM.SYSRESETn

43.7.2.12 ETHERNETSS_PTP_TSWH Register (Offset = 2Ch) [Reset = 0h]

ETHERNETSS_PTP_TSWH is shown in Figure 43-38 and described in Table 43-90.

Return to the Summary Table.

External PTP Timestamp write upper 32 bits

Figure 43-38. ETHERNETSS_PTP_TSWH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSW_H																															
R/W-0h																															

Table 43-90. ETHERNETSS_PTP_TSWH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TSW_H	R/W	0h	Higher 32 bit time stamp value input to ptp_timestamp_i[63:32] to be embedded in next packet. This could be value read from external source or in systems with RTC could be connected to RTC counter directly. Reset type: CM.SYSRESETn

43.7.2.13 ETHERNETSS_REVMII_CTRL Register (Offset = 30h) [Reset = 0h]

ETHERNETSS_REVMII_CTRL is shown in Figure 43-39 and described in Table 43-91.

Return to the Summary Table.

REVMII PHY addresses input values.

Figure 43-39. ETHERNETSS_REVMII_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
WRITE_KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED			REVMII_REMOTE_PHY_ADDR				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
RESERVED			REVMII_CORE_PHY_ADDR				
R-0h			R/W-0h				

Table 43-91. ETHERNETSS_REVMII_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	WRITE_KEY	R-0/W	0h	The key value should be 0xa5 for the writes to this register to take effect. Writes to other controls in register with value in this field other than specified will be ignored. Reset type: CM.SYSRESETn
15-13	RESERVED	R	0h	Reserved
12-8	REVMII_REMOTE_PHY_ADDR	R/W	0h	Address offset to access the control-status related to PHY related status/controls for the Remote MAC side. (RevMII does not have PHY) Reset type: CM.SYSRESETn
7-5	RESERVED	R	0h	Reserved
4-0	REVMII_CORE_PHY_ADDR	R/W	0h	Address offset to access the control-status related to PHY related status/controls for the core side. (RevMII does not have PHY) Reset type: CM.SYSRESETn

43.7.3 EMAC_REGS Registers

Table 43-92 lists the memory-mapped registers for the EMAC_REGS registers. All register offset addresses not listed in Table 43-92 should be considered as reserved locations and the register contents should not be modified.

Table 43-92. EMAC_REGS Registers

Offset	Acronym	Register Name	Section
0h	MAC_Configuration	The MAC Configuration Register establishes the operating mode of the MAC.	Go
4h	MAC_Ext_Configuration	The MAC Extended Configuration Register establishes the operating mode of the MAC.	Go
8h	MAC_Packet_Filter	The MAC Packet Filter register contains the filter controls for receiving packets. Some of the controls from this register go to the address check block of the MAC which performs the first level of address filtering. The second level of filtering is performed on the incoming packet based on other controls such as Pass Bad Packets and Pass Control Packets.	Go
Ch	MAC_Watchdog_Timeout	The Watchdog Timeout register controls the watchdog timeout for received packets.	Go
10h	MAC_Hash_Table_Reg0	The Hash Table Register 0 contains the first 32 bits of the hash table, when the width of the hash table is 128 or 256 bits. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant. The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5. The hash value of the destination address is calculated in the following way: - Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32). - Perform bitwise reversal for the value obtained in Step 1. - Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2. If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values. If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written. If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
14h	MAC_Hash_Table_Reg1	The Hash Table Register 1 contains the second 32 bits of the hash table. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant. The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5. The hash value of the destination address is calculated in the following way: - Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32). - Perform bitwise reversal for the value obtained in Step 1. - Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2. If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values. If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written. If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.	Go
50h	MAC_VLAN_Tag_Ctrl	This register is the redefined format of the MAC VLAN Tag Register. It is used for indirect addressing. It contains the address offset, command type and Busy Bit for CSR access of the Per VLAN Tag registers.	Go
54h	MAC_VLAN_Tag_Data	This register holds the read/write data for Indirect Access of the Per VLAN Tag registers. During the read access, this field contains valid read data only after the OB bit is reset. During the write access, this field should be valid prior to setting the OB bit in the MAC_VLAN_Tag_Ctrl Register.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
58h	MAC_VLAN_Hash_Table	When VTHM bit of the MAC_VLAN_Tag register is set, the 16-bit VLAN Hash Table register is used for group address filtering based on the VLAN tag. For hash filtering, the content of the 16-bit VLAN tag or 12-bit VLAN ID (based on the ETV bit of MAC_VLAN_Tag Register) in the incoming packet is passed through the CRC logic. The upper four bits of the calculated CRC are used to index the contents of the VLAN Hash table. For example, a hash value of 4b'1000 selects Bit 8 of the VLAN Hash table. The hash value of the destination address is calculated in the following way: - Calculate the 32-bit CRC for the VLAN tag or ID (For steps to calculate CRC32, see Section 3.2.8 of IEEE 802.3). - Perform bitwise reversal for the value obtained in step 1. - Take the upper four bits from the value obtained in step 2. If the VLAN hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[15:8] (in little-endian mode) or Bits[7:0] (in big-endian mode) of this register are written. - If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.	Go
60h	MAC_VLAN_Incl	The VLAN Tag Inclusion or Replacement register contains the VLAN tag for insertion or replacement in the Transmit packets. It also contains the VLAN tag insertion controls.	Go
64h	MAC_Inner_VLAN_Incl	The Inner VLAN Tag Inclusion or Replacement register contains the inner VLAN tag to be inserted or replaced in the Transmit packet. It also contains the inner VLAN tag insertion controls.	Go
70h	MAC_Q0_Tx_Flow_Ctrl	The Flow Control register controls the generation and reception of the Control (Pause Command) packets by the Flow control module of the MAC. A Write to a register with the Busy bit set to 1 triggers the Flow Control block to generate a Pause packet. The fields of the control packet are selected as specified in the 802.3x specification, and the Pause Time value from this register is used in the Pause Time field of the control packet. The Busy bit remains set until the control packet is transferred onto the cable. The application must make sure that the Busy bit is cleared before writing to the register. When the PFCE bit in the MAC_Rx_Flow_Ctrl register is enabled, this register controls the generation of Priority Flow Control (PFC) frames with priorities mapped according to PSRQ0 in the MAC_RxQ_Ctrl2 register.	Go
90h	MAC_Rx_Flow_Ctrl	The Receive Flow Control register controls the pausing of MAC Transmit based on the received Pause packet.	Go
94h	MAC_RxQ_Ctrl4	The Receive Queue Control 4 register controls the routing of unicast and multicast packets that fail the Destination or Source address filter to the Rx queues.	Go
A0h	MAC_RxQ_Ctrl0	The Receive Queue Control 0 register controls the queue management in the MAC Receiver. Note: In multiple Rx queues configuration, all the queues are disabled by default. Enable the Rx queue by programming the corresponding field in this register.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
A4h	MAC_RxQ_Ctrl1	The Receive Queue Control 1 register controls the routing of multicast, broadcast, AV, DCB, and untagged packets to the Rx queues.	Go
A8h	MAC_RxQ_Ctrl2	This register controls the routing of tagged packets based on the USP (user Priority) field of the received packets to the RxQueues 0 to 3.	Go
B0h	MAC_Interrupt_Status	The Interrupt Status register contains the status of interrupts.	Go
B4h	MAC_Interrupt_Enable	The Interrupt Enable register contains the masks for generating the interrupts.	Go
B8h	MAC_Rx_Tx_Status	The Receive Transmit Status register contains the Receive and Transmit Error status.	Go
C0h	MAC_PMT_Control_Status	The PMT Control and Status Register.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
C4h	MAC_RWK_Packet_Filter	The wkuppktfilter_reg register at address 0C4H loads the Wake-up Packet Filter register. To load values in a Wake-up Packet Filter register, the entire register (wkuppktfilter_reg) must be written. The wkuppktfilter_reg register is loaded by sequentially loading the eight, sixteen or thirty two register values in address (0C4H) for wkuppktfilter_reg0, wkuppktfilter_reg1,.. wkuppktfilter_reg31, respectively. The wkuppktfilter_reg register is read in a similar way. The DWC_ether_qos updates the wkuppktfilter_reg register current pointer value in Bits[26:24] of MAC_PMT_Control_Status register. Filter i Byte Mask: The filter i byte mask register defines the bytes of the packet that are examined by filter i (0, 1, 2, 3,..,15) to determine whether or not a packet is a wake-up packet. - The MSB (31st bit) must be zero. - Bit j[30:0] is the byte mask. - If Bit j (byte number) of the byte mask is set, the CRC block processes the Filter i Offset + j of the incoming packet; otherwise Filter i Offset + j is ignored. Filter i Command: The 4-bit filter i command controls the filter i operation. - Bit 3 specifies the address type, defining the destination address type of the pattern. When the bit is set, the pattern applies to only multicast packets; when the bit is reset, the pattern applies only to unicast packet. - Bit 2 (Inverse Mode), when set, reverses the logic of the CRC16 hash function signal, to reject a packet with matching CRC_16 value. - Bit 2, along with Bit 1, allows a MAC to reject a subset of remote wake-up packets by creating filter logic such as "Pattern 1 AND NOT Pattern 2". - Bit 1 (And_Previous) implements the Boolean logic. When set, the result of the current entry is logically ANDed with the result of the previous filter. This AND logic allows a filter pattern longer than 32 bytes by splitting the mask among two, three, or four filters. This depends on the number of filters that have the And_Previous bit set. - Bit 0 is the enable for filter i. If Bit 0 is not set, filter i is disabled. Filter i Offset: This filter i offset register defines the offset (within the packet) from which the filter i examines the packets. - This 8-bit pattern-offset is the offset for the filter i first byte to be examined. - The minimum allowed offset is 12, which refers to the 13th byte of the packet. - The offset value 0 refers to the first byte of the packet. Filter i CRC-16: This filter i CRC-16 register contains the CRC_16 value calculated from the pattern and also the byte mask programmed to the wake-up filter register block. - The 16-bit CRC calculation uses the following polynomial: G(x) = x^16 + x^15 + x^2 + 1 Each mask, used in the hash function calculation, is compared with a 16-bit value associated with that mask. Each filter has the following: - 32-bit Mask: Each bit in this mask corresponds to one byte in the detected packet.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
		If the bit is 1', the corresponding byte is taken into the CRC16 calculation. - 8-bit Offset Pointer: Specifies the byte to start the CRC16 computation. The pointer and the mask are used together to locate the bytes to be used in the CRC16 calculations. - Note: If you are accessing these registers in byte or half-word mode, the internal counter to access the appropriate wkuppktfilter_reg is incremented when CPU accesses Lane 3 (or Lane 0 in big-endian mode). - Note: When any Register content is being transferred to a different clock domain after a write operation, there should not be any further writes to the same location until the first write is updated. Otherwise, the second write operation does not get updated to the destination clock domain. Therefore, the delay between two writes to the same register location should be at least 4 cycles of the destination clock (PHY receive clock, PHY transmit clock, or PTP clock). Notes on And_Previous bit setting The And_Previous bit setting is applicable within a set of 4 filters. - Setting of And_Previous bit of filter that is not enabled has no effect. In other words, setting And_Previous bit of lowest number filter in the set of 4 filters has no effect. For example, setting of And_Previous bit of Filter 0 has no effect. - If And_Previous bit is set for filter to form AND chained filter, the AND chain breaks at the point any filter is not enabled. For example: If Filter 2 And_Previous bit is set (bit 1 of Filter 2 command is set) but Filter 1 is not enabled (bit 0 of in Filter 1 command is reset), then only Filter 2 result is considered. If Filter 2 And_Previous bit is set (bit 1 of Filter 2 command is set), Filter 3 And_Previous bit is set (bit 1 of Filter 3 command is set), but Filter 1 is not enabled (bit 0 of in Filter 1 command is reset), then only Filter 2 result ANDed with Filter 3 result is considered. If Filter 2 And_Previous bit is set (bit 1 of Filter 2 command is set), Filter 3 And_Previous bit is set (bit 1 of Filter 3 command is set), but Filter 2 is not enabled (bit 0 of in Filter 2 command is reset), then since setting of Filter 2 And_Previous bit has no effect only Filter 1 result ORed with Filter 3 result is considered. - If filters chained by And_Previous bit setting have complementary programming, then a frame may never pass the AND chained filter. For example, if Filter 2 And_Previous bit is set (bit 1 of Filter 2 command is set), Filter 1 Address_Type bit is set (bit 3 of Filter 1 command is set) indicating multicast detection and Filter 2 Address_Type bit is reset (bit 3 of Filter 2 command is reset) indicating unicast detection or vice versa, a remote wakeup frame does not pass the AND chained filter as a remote wakeup frame cannot be of both unicast and multicast address type.	
D0h	MAC_LPI_Control_Status	The LPI Control and Status Register controls the LPI functions and provides the LPI interrupt status. The status bits are cleared when this register is read.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
D4h	MAC_LPI_Timers_Control	The LPI Timers Control register controls the timeout values in the LPI states. It specifies the time for which the MAC transmits the LPI pattern and also the time for which the MAC waits before resuming the normal transmission.	Go
D8h	MAC_LPI_Entry_Timer	This register controls the Tx LPI entry timer. This counter is enabled only when bit[20](LPITE) bit of MAC_LPI_Control_Status is set to 1.	Go
DCh	MAC_1US_Tic_Counter	This register controls the generation of the Reference time (1 microsecond tic) for all the LPI timers. This timer has to be programmed by the software initially.	Go
110h	MAC_Version	The version register identifies the version of the DWC_ether_qos. This register contains two bytes: one that Synopsys uses to identify the core release number, and the other that you set while configuring the core.	Go
114h	MAC_Debug	The Debug register provides the debug status of various MAC blocks.	Go
11Ch	MAC_HW_Feature0	This register indicates the presence of first set of the optional features or functions of the DWC_ether_qos. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks. Note: All bits are set or reset according to the features selected while configuring the core in coreConsultant.	Go
120h	MAC_HW_Feature1	This register indicates the presence of second set of the optional features or functions of the DWC_ether_qos. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks. Note: All bits are set or reset according to the features selected while configuring the core in coreConsultant.	Go
124h	MAC_HW_Feature2	This register indicates the presence of third set of the optional features or functions of the DWC_ether_qos. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks.	Go
128h	MAC_HW_Feature3	This register indicates the presence of fourth set the optional features or functions of the DWC_ether_qos. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks.	Go
200h	MAC_MDIO_Address	The MDIO Address register controls the management cycles to external PHY through a management interface.	Go
204h	MAC_MDIO_Data	The MDIO Data register stores the Write data to be written to the PHY register located at the address specified in MAC_MDIO_Address. This register also stores the Read data from the PHY register located at the address specified by MDIO Address register.	Go
210h	MAC_ARP_Address	The ARP Address register contains the IPv4 Destination Address of the MAC. Note: IP address should be written to this register in host byte order format.	Go
230h	MAC_CSR_SW_Ctrl	This register contains SW programmable controls for changing the CSR access response and status bits clearing.	Go
238h	MAC_Ext_Cfg1	This register contains Split mode control field and offset field for Split Header feature.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
300h	MAC_Address0_High	The MAC Address0 High register holds the upper 16 bits of the first 6-byte MAC address of the station. The first DA byte that is received on the (G)MII interface corresponds to the LS byte (Bits [7:0]) of the MAC Address Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (G)MII as the destination address, then the MacAddress0 Register [47:0] is compared with 0x665544332211. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address0 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.	Go
304h	MAC_Address0_Low	The MAC Address0 Low register holds the lower 32 bits of the 6-byte first MAC address of the station.	Go
308h	MAC_Address1_High	The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.	Go
30Ch	MAC_Address1_Low	The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.	Go
310h	MAC_Address2_High	The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.	Go
314h	MAC_Address2_Low	The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.	Go
318h	MAC_Address3_High	The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.	Go
31Ch	MAC_Address3_Low	The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
320h	MAC_Address4_High	The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.	Go
324h	MAC_Address4_Low	The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.	Go
328h	MAC_Address5_High	The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.	Go
32Ch	MAC_Address5_Low	The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.	Go
330h	MAC_Address6_High	The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.	Go
334h	MAC_Address6_Low	The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.	Go
338h	MAC_Address7_High	The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.	Go
33Ch	MAC_Address7_Low	The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.	Go
700h	MMC_Control	This register establishes the operating mode of MMC.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
704h	MMC_Rx_Interrupt	This register maintains the interrupts generated from all Receive statistics counters. The MMC Receive Interrupt register maintains the interrupts that are generated when the following occur: - Receive statistic counters reach half of their maximum values (0x8000_0000 for 32 bit counter and 0x8000 for 16 bit counter). - Receive statistic counters cross their maximum values (0xFFFF_FFFF for 32 bit counter and 0xFFFF for 16 bit counter). When the Counter Stop Rollover is set, interrupts are set but the counter remains at all-ones. The MMC Receive Interrupt register is a 32 bit register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (Bits[7:0]) of the respective counter must be read to clear the interrupt bit. Note: R_SS_RC means that this register bit is set internally, and it is cleared when the Counter register is read.	Go
708h	MMC_Tx_Interrupt	This register maintains the interrupts generated from all Transmit statistics counters. The MMC Transmit Interrupt register maintains the interrupts generated when transmit statistic counters reach half their maximum values (0x8000_0000 for 32 bit counter and 0x8000 for 16 bit counter), and when they cross their maximum values (0xFFFF_FFFF for 32-bit counter and 0xFFFF for 16-bit counter). When Counter Stop Rollover is set, the interrupts are set but the counter remains at all-ones. The MMC Transmit Interrupt register is a 32 bit register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (Bits[7:0]) of the respective counter must be read to clear the interrupt bit.	Go
70Ch	MMC_Rx_Interrupt_Mask	This register maintains the masks for interrupts generated from all Receive statistics counters. The MMC Receive Interrupt Mask register maintains the masks for the interrupts generated when receive statistic counters reach half of their maximum value or the maximum values. This register is 32 bit wide.	Go
710h	MMC_Tx_Interrupt_Mask	This register maintains the masks for interrupts generated from all Transmit statistics counters. The MMC Transmit Interrupt Mask register maintains the masks for the interrupts generated when the transmit statistic counters reach half of their maximum value or the maximum values. This register is 32 bit wide.	Go
714h	Tx_Octet_Count_Good_Bad	This register provides the number of bytes transmitted by the DWC_ether_qos, exclusive of preamble and retried bytes, in good and bad packets.	Go
718h	Tx_Packet_Count_Good_Bad	This register provides the number of good and bad packets transmitted by DWC_ether_qos, exclusive of retried packets.	Go
71Ch	Tx_Broadcast_Packets_Good	This register provides the number of good broadcast packets transmitted by DWC_ether_qos.	Go
720h	Tx_Multicast_Packets_Good	This register provides the number of good multicast packets transmitted by DWC_ether_qos.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
724h	Tx_64Octets_Packets_Good_Bad	This register provides the number of good and bad packets transmitted by DWC_ether_qos with length 64 bytes, exclusive of preamble and retried packets.	Go
728h	Tx_65To127Octets_Packets_Good_Bad	This register provides the number of good and bad packets transmitted by DWC_ether_qos with length between 65 and 127 (inclusive) bytes, exclusive of preamble and retried packets.	Go
72Ch	Tx_128To255Octets_Packets_Good_Bad	This register provides the number of good and bad packets transmitted by DWC_ether_qos with length between 128 to 255 (inclusive) bytes, exclusive of preamble and retried packets.	Go
730h	Tx_256To511Octets_Packets_Good_Bad	This register provides the number of good and bad packets transmitted by DWC_ether_qos with length between 256 to 511 (inclusive) bytes, exclusive of preamble and retried packets.	Go
734h	Tx_512To1023Octets_Packets_Good_Bad	This register provides the number of good and bad packets transmitted by DWC_ether_qos with length 512 to 1023 (inclusive) bytes, exclusive of preamble and retried packets.	Go
738h	Tx_1024ToMaxOctets_Packets_Good_Bad	This register provides the number of good and bad packets transmitted by DWC_ether_qos with length 1024 to maxsize (inclusive) bytes, exclusive of preamble and retried packets.	Go
73Ch	Tx_Unicast_Packets_Good_Bad	This register provides the number of good and bad unicast packets transmitted by DWC_ether_qos.	Go
740h	Tx_Multicast_Packets_Good_Bad	This register provides the number of good and bad multicast packets transmitted by DWC_ether_qos.	Go
744h	Tx_Broadcast_Packets_Good_Bad	This register provides the number of good and bad broadcast packets transmitted by DWC_ether_qos.	Go
748h	Tx_Underflow_Error_Packets	This register provides the number of packets aborted by DWC_ether_qos because of packets underflow error.	Go
74Ch	Tx_Single_Collision_Good_Packets	This register provides the number of successfully transmitted packets by DWC_ether_qos after a single collision in the half-duplex mode.	Go
750h	Tx_Multiple_Collision_Good_Packets	This register provides the number of successfully transmitted packets by DWC_ether_qos after multiple collisions in the half-duplex mode.	Go
754h	Tx_Deferred_Packets	This register provides the number of successfully transmitted by DWC_ether_qos after a deferral in the half-duplex mode.	Go
758h	Tx_Late_Collision_Packets	This register provides the number of packets aborted by DWC_ether_qos because of late collision error.	Go
75Ch	Tx_Excessive_Collision_Packets	This register provides the number of packets aborted by DWC_ether_qos because of excessive (16) collision errors.	Go
760h	Tx_Carrier_Error_Packets	This register provides the number of packets aborted by DWC_ether_qos because of carrier sense error (no carrier or loss of carrier).	Go
764h	Tx_Octet_Count_Good	This register provides the number of bytes transmitted by DWC_ether_qos, exclusive of preamble, only in good packets.	Go
768h	Tx_Packet_Count_Good	This register provides the number of good packets transmitted by DWC_ether_qos.	Go
76Ch	Tx_Excessive_Deferral_Error	This register provides the number of packets aborted by DWC_ether_qos because of excessive deferral error (deferred for more than two max-sized packet times).	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
770h	Tx_Pause_Packets	This register provides the number of good Pause packets transmitted by DWC_ether_qos.	Go
774h	Tx_VLAN_Packets_Good	This register provides the number of good VLAN packets transmitted by DWC_ether_qos.	Go
778h	Tx_OSize_Packets_Good	This register provides the number of packets transmitted by DWC_ether_qos without errors and with length greater than the maxsize (1,518 or 1,522 bytes for VLAN tagged packets; 2000 bytes if enabled in S2KP bit of the MAC_Configuration register).	Go
780h	Rx_Packets_Count_Good_Bad	This register provides the number of good and bad packets received by DWC_ether_qos.	Go
784h	Rx_Octet_Count_Good_Bad	This register provides the number of bytes received by DWC_ther_qos, exclusive of preamble, in good and bad packets.	Go
788h	Rx_Octet_Count_Good	This register provides the number of bytes received by DWC_ether_qos, exclusive of preamble, only in good packets.	Go
78Ch	Rx_Broadcast_Packets_Good	This register provides the number of good broadcast packets received by DWC_ether_qos.	Go
790h	Rx_Multicast_Packets_Good	This register provides the number of good multicast packets received by DWC_ether_qos.	Go
794h	Rx_CRC_Error_Packets	This register provides the number of packets received by DWC_ether_qos with CRC error.	Go
798h	Rx_Alignment_Error_Packets	This register provides the number of packets received by DWC_ether_qos with alignment (dribble) error. It is valid only in 10/100 mode.	Go
79Ch	Rx_Runt_Error_Packets	This register provides the number of packets received by DWC_ether_qos with runt (length less than 64 bytes and CRC error) error.	Go
7A0h	Rx_Jabber_Error_Packets	This register provides the number of giant packets received by DWC_ether_qos with length (including CRC) greater than 1,518 bytes (1,522 bytes for VLAN tagged) and with CRC error. If Jumbo Packet mode is enabled, packets of length greater than 9,018 bytes (9,022 bytes for VLAN tagged) are considered as giant packets.	Go
7A4h	Rx_Undersize_Packets_Good	This register provides the number of packets received by DWC_ether_qos with length less than 64 bytes, without any errors.	Go
7A8h	Rx_Oversize_Packets_Good	This register provides the number of packets received by DWC_ether_qos without errors, with length greater than the maxsize (1,518 bytes or 1,522 bytes for VLAN tagged packets; 2000 bytes if enabled in the S2KP bit of the MAC_Configuration register).	Go
7ACh	Rx_64Octets_Packets_Good_Bad	This register provides the number of good and bad packets received by DWC_ether_qos with length 64 bytes, exclusive of the preamble.	Go
7B0h	Rx_65To127Octets_Packets_Good_Bad	This register provides the number of good and bad packets received by DWC_ether_qos with length between 65 and 127 (inclusive) bytes, exclusive of the preamble.	Go
7B4h	Rx_128To255Octets_Packets_Good_Bad	This register provides the number of good and bad packets received by DWC_ether_qos with length between 128 and 255 (inclusive) bytes, exclusive of the preamble.	Go
7B8h	Rx_256To511Octets_Packets_Good_Bad	This register provides the number of good and bad packets received by DWC_ether_qos with length between 256 and 511 (inclusive) bytes, exclusive of the preamble.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
7BCh	Rx_512To1023Octets_Packets_Good_Bad	This register provides the number of good and bad packets received by DWC_ether_qos with length between 512 and 1023 (inclusive) bytes, exclusive of the preamble.	Go
7C0h	Rx_1024ToMaxOctets_Packets_Good_Bad	This register provides the number of good and bad packets received by DWC_ether_qos with length between 1024 and maxsize (inclusive) bytes, exclusive of the preamble.	Go
7C4h	Rx_Unicast_Packets_Good	This register provides the number of good unicast packets received by DWC_ether_qos.	Go
7C8h	Rx_Length_Error_Packets	This register provides the number of packets received by DWC_ether_qos with length error (Length Type field not equal to packet size), for all packets with valid length field.	Go
7CCh	Rx_Out_Of_Range_Type_Packets	This register provides the number of packets received by DWC_ether_qos with length field not equal to the valid packet size (greater than 1,500 but less than 1,536).	Go
7D0h	Rx_Pause_Packets	This register provides the number of good and valid Pause packets received by DWC_ether_qos.	Go
7D4h	Rx_FIFO_Overflow_Packets	This register provides the number of missed received packets because of FIFO overflow in DWC_ether_qos.	Go
7D8h	Rx_VLAN_Packets_Good_Bad	This register provides the number of good and bad VLAN packets received by DWC_ether_qos.	Go
7DCh	Rx_Watchdog_Error_Packets	This register provides the number of packets received by DWC_ether_qos with error because of watchdog timeout error (packets with a data load larger than 2,048 bytes (when JE and WD bits are reset in MAC_Configuration register), 10,240 bytes (when JE bit is set and WD bit is reset in MAC_Configuration register), 16,384 bytes (when WD bit is set in MAC_Configuration register) or the value programmed in the MAC_Watchdog_Timeout register).	Go
7E0h	Rx_Receive_Error_Packets	This register provides the number of packets received by DWC_ether_qos with Receive error or Packet Extension error on the GMII or MII interface.	Go
7E4h	Rx_Control_Packets_Good	This register provides the number of good control packets received by DWC_ether_qos.	Go
7ECh	Tx_LPI_USEC_Cntr	This register provides the number of microseconds Tx LPI is asserted by DWC_ether_qos.	Go
7F0h	Tx_LPI_Tran_Cntr	This register provides the number of times DWC_ether_qos has entered Tx LPI.	Go
7F4h	Rx_LPI_USEC_Cntr	This register provides the number of microseconds Rx LPI is sampled by DWC_ether_qos.	Go
7F8h	Rx_LPI_Tran_Cntr	This register provides the number of times DWC_ether_qos has entered Rx LPI.	Go
800h	MMC_IPC_Rx_Interrupt_Mask	This register maintains the mask for the interrupt generated from the receive IPC statistic counters. The MMC Receive Checksum Off load Interrupt Mask register maintains the masks for the interrupts generated when the receive IPC (Checksum Off load) statistic counters reach half their maximum value, and when they reach their maximum values. This register is 32 bits wide.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
808h	MMC_IPC_Rx_Interrupt	This register maintains the interrupt that the receive IPC statistic counters generate. The MMC Receive Checksum Offload Interrupt register maintains the interrupts generated when receive IPC statistic counters reach half their maximum values (0x8000_0000 for 32 bit counter and 0x8000 for 16 bit counter), and when they cross their maximum values (0xFFFF_FFFF for 32 bit counter and 0xFFFF for 16 bit counter). When Counter Stop Rollover is set, the interrupts are set but the counter remains at all-ones. The MMC Receive Checksum Offload Interrupt register is 32 bit wide. When the MMC IPC counter that caused the interrupt is read, its corresponding interrupt bit is cleared. The counter's least-significant byte lane (Bits[7:0]) must be read to clear the interrupt bit.	Go
810h	RxIPv4_Good_Packets	This register provides the number of good IPv4 datagrams received by DWC_ether_qos with the TCP, UDP, or ICMP payload.	Go
814h	RxIPv4_Header_Error_Packets	RxIPv4 Header Error Packets This register provides the number of IPv4 datagrams received by DWC_ether_qos with header (checksum, length, or version mismatch) errors.	Go
818h	RxIPv4_No_Payload_Packets	This register provides the number of IPv4 datagram packets received by DWC_ether_qos that did not have a TCP, UDP, or ICMP payload.	Go
81Ch	RxIPv4_Fragmented_Packets	This register provides the number of good IPv4 datagrams received by DWC_ether_qos with fragmentation.	Go
820h	RxIPv4_UDP_Checksum_Disabled_Packets	This register provides the number of good IPv4 datagrams received by DWC_ether_qos that had a UDP payload with checksum disabled.	Go
824h	RxIPv6_Good_Packets	This register provides the number of good IPv6 datagrams received by DWC_ether_qos with the TCP, UDP, or ICMP payload.	Go
828h	RxIPv6_Header_Error_Packets	This register provides the number of IPv6 datagrams received by DWC_ether_qos with header (length or version mismatch) errors.	Go
82Ch	RxIPv6_No_Payload_Packets	This register provides the number of IPv6 datagram packets received by DWC_ether_qos that did not have a TCP, UDP, or ICMP payload. This includes all IPv6 datagrams with fragmentation or security extension headers.	Go
830h	RxUDP_Good_Packets	This register provides the number of good IP datagrams received by DWC_ether_qos with a good UDP payload. This counter is not updated when the RxIPv4_UDP_Checksum_Disabled_Packets counter is incremented.	Go
834h	RxUDP_Error_Packets	This register provides the number of good IP datagrams received by DWC_ether_qos whose UDP payload has a checksum error.	Go
838h	RxTCP_Good_Packets	This register provides the number of good IP datagrams received by DWC_ether_qos with a good TCP payload.	Go
83Ch	RxTCP_Error_Packets	This register provides the number of good IP datagrams received by DWC_ether_qos whose TCP payload has a checksum error.	Go
840h	RxICMP_Good_Packets	This register provides the number of good IP datagrams received by DWC_ether_qos with a good ICMP payload.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
844h	RxICMP_Error_Packets	This register provides the number of good IP datagrams received by DWC_ether_qos whose ICMP payload has a checksum error.	Go
850h	RxIPv4_Good_Octets	This register provides the number of bytes received by DWC_ether_qos in good IPv4 datagrams encapsulating TCP, UDP, or ICMP data. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.	Go
854h	RxIPv4_Header_Error_Octets	This register provides the number of bytes received by DWC_ether_qos in IPv4 datagrams with header errors (checksum, length, version mismatch). The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.	Go
858h	RxIPv4_No_Payload_Octets	This register provides the number of bytes received by DWC_ether_qos in IPv4 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.	Go
85Ch	RxIPv4_Fragmented_Octets	This register provides the number of bytes received by DWC_ether_qos in fragmented IPv4 datagrams. The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.	Go
860h	RxIPv4_UDP_Checksum_Disable_Octets	This register provides the number of bytes received by DWC_ether_qos in a UDP segment that had the UDP checksum disabled. This counter does not count IP Header bytes. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.	Go
864h	RxIPv6_Good_Octets	This register provides the number of bytes received by DWC_ether_qos in good IPv6 datagrams encapsulating TCP, UDP, or ICMP data. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.	Go
868h	RxIPv6_Header_Error_Octets	This register provides the number of bytes received by DWC_ether_qos in IPv6 datagrams with header errors (length, version mismatch). The value in the Length field of IPv6 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.	Go
86Ch	RxIPv6_No_Payload_Octets	This register provides the number of bytes received by DWC_ether_qos in IPv6 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the Length field of IPv6 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.	Go
870h	RxUDP_Good_Octets	This register provides the number of bytes received by DWC_ether_qos in a good UDP segment. This counter does not count IP header bytes.	Go
874h	RxUDP_Error_Octets	This register provides the number of bytes received by DWC_ether_qos in a UDP segment that had checksum errors. This counter does not count IP header bytes.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
878h	RxTCP_Good_Octets	This register provides the number of bytes received by DWC_ether_qos in a good TCP segment. This counter does not count IP header bytes.	Go
87Ch	RxTCP_Error_Octets	This register provides the number of bytes received by DWC_ether_qos in a TCP segment that had checksum errors. This counter does not count IP header bytes.	Go
880h	RxICMP_Good_Octets	This register provides the number of bytes received by DWC_ether_qos in a good ICMP segment. This counter does not count IP header bytes.	Go
884h	RxICMP_Error_Octets	This register provides the number of bytes received by DWC_ether_qos in a ICMP segment that had checksum errors. This counter does not count IP header bytes.	Go
900h	MAC_L3_L4_Control0	The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.	Go
904h	MAC_Layer4_Address0	The Layer 4 Address 0 register and registers 580 through 667 are reserved (RO with default value) if Enable Layer 3 and Layer 4 Packet Filter option is not selected while configuring the core. You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option while configuring the core. When you select this option, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform consecutive writes to same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.	Go
910h	MAC_Layer3_Addr0_Reg0	For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.	Go
914h	MAC_Layer3_Addr1_Reg0	For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.	Go
918h	MAC_Layer3_Addr2_Reg0	The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.	Go
91Ch	MAC_Layer3_Addr3_Reg0	The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.	Go
930h	MAC_L3_L4_Control1	The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
934h	MAC_Layer4_Address1	The Layer 4 Address 0 register and registers 580 through 667 are reserved (RO with default value) if Enable Layer 3 and Layer 4 Packet Filter option is not selected while configuring the core. You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option while configuring the core. When you select this option, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform consecutive writes to same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.	Go
940h	MAC_Layer3_Addr0_Reg1	For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.	Go
944h	MAC_Layer3_Addr1_Reg1	For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.	Go
948h	MAC_Layer3_Addr2_Reg1	The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.	Go
94Ch	MAC_Layer3_Addr3_Reg1	The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.	Go
960h	MAC_L3_L4_Control2	The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.	Go
964h	MAC_Layer4_Address2	The Layer 4 Address 0 register and registers 580 through 667 are reserved (RO with default value) if Enable Layer 3 and Layer 4 Packet Filter option is not selected while configuring the core. You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option while configuring the core. When you select this option, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform consecutive writes to same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.	Go
970h	MAC_Layer3_Addr0_Reg2	For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.	Go
974h	MAC_Layer3_Addr1_Reg2	For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
978h	MAC_Layer3_Addr2_Reg2	The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.	Go
97Ch	MAC_Layer3_Addr3_Reg2	The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.	Go
990h	MAC_L3_L4_Control3	The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.	Go
994h	MAC_Layer4_Address3	The Layer 4 Address 0 register and registers 580 through 667 are reserved (RO with default value) if Enable Layer 3 and Layer 4 Packet Filter option is not selected while configuring the core. You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option while configuring the core. When you select this option, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform consecutive writes to same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.	Go
9A0h	MAC_Layer3_Addr0_Reg3	For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.	Go
9A4h	MAC_Layer3_Addr1_Reg3	For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.	Go
9A8h	MAC_Layer3_Addr2_Reg3	The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.	Go
9ACh	MAC_Layer3_Addr3_Reg3	The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.	Go
B00h	MAC_Timestamp_Control	This register controls the operation of the System Time generator and processing of PTP packets for timestamping in the Receiver.	Go
B04h	MAC_Sub_Second_Increment	This register specifies the value to be added to the internal system time register every cycle of clk_ptp_ref_i clock.	Go
B08h	MAC_System_Time_Seconds	The System Time Seconds register, along with System Time Nanoseconds register, indicates the current value of the system time maintained by the MAC. Though it is updated on a continuous basis, there is some delay from the actual time because of clock domain transfer latencies (from clk_ptp_ref_i to CSR clock).	Go
B0Ch	MAC_System_Time_Nanoseconds	The System Time Nanoseconds register, along with System Time Seconds register, indicates the current value of the system time maintained by the MAC.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
B10h	MAC_System_Time_Seconds_Update	The System Time Seconds Update register, along with the System Time Nanoseconds Update register, initializes or updates the system time maintained by the MAC. You must write both registers before setting the TSINIT or TSUPDT bits in EMAC_REGS/EQOS_MAC/ MAC_Timestamp_Control.	Go
B14h	MAC_System_Time_Nanoseconds_Update	MAC System Time Nanoseconds Update register.	Go
B18h	MAC_Timestamp_Addend	Timestamp Addend register. This register value is used only when the system time is configured for Fine Update mode (TSCFUPDT bit in the MAC_Timestamp_Control register). The content of this register is added to a 32-bit accumulator in every clock cycle (of clk_ptp_ref_i) and the system time is updated whenever the accumulator overflows.	Go
B1Ch	MAC_System_Time_Higher_Word_Seconds	System Time - Higher Word Seconds register.	Go
B20h	MAC_Timestamp_Status	Timestamp Status register. All bits except Bits[27:25] gets cleared when the application reads this register.	Go
B30h	MAC_Tx_Timestamp_Status_Nanoseconds	This register contains the nanosecond part of timestamp captured for Transmit packets when Tx status is disabled. The MAC_Tx_Timestamp_Status_Nanoseconds register, along with MAC_Tx_Timestamp_Status_Seconds, gives the 64-bit timestamp captured for the PTP packet successfully transmitted by the MAC. This value is considered to be read by the application when the last byte of MAC_Tx_Timestamp_Status_Nanoseconds is read. In the little-endian mode, this means when bits[31:24] are read; in big-endian mode, bits[7:0] are read. If the application does not read these registers and timestamp of another packet is captured, then either the current timestamp is lost (overwritten) or the new timestamp is lost (dropped), depending on the setting of the TXTSSTSM bit of the MAC_Timestamp_Control register. The status bit TXTSC bit [15] in MAC_Timestamp_Status register is set whenever the MAC transmitter captures the timestamp.	Go
B34h	MAC_Tx_Timestamp_Status_Seconds	The register contains the higher 32 bits of the timestamp (in seconds) captured when a PTP packet is transmitted.	Go
B40h	MAC_Auxiliary_Control	The Auxiliary Timestamp Control register controls the Auxiliary Timestamp snapshot.	Go
B48h	MAC_Auxiliary_Timestamp_Nanoseconds	The Auxiliary Timestamp Nanoseconds register, along with MAC_Auxiliary_Timestamp_Seconds, gives the 64-bit timestamp stored as auxiliary snapshot. These two registers form the read port of a 64-bit wide FIFO with a depth of 4, 8, or 16 as selected while configuring the core. You can store multiple snapshots in this FIFO. Bits[29:25] in MAC_Timestamp_Status indicate the fill-level of the FIFO. The top of the FIFO is removed only when the last byte of MAC Register 91 (Auxiliary Timestamp - Seconds Register) is read. In the little-endian mode, this means when Bits[31:24] are read and in big-endian mode, Bits[7:0] are read.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
B4Ch	MAC_Auxiliary_Timestamp_Seconds	The Auxiliary Timestamp - Seconds register contains the lower 32 bits of the Seconds field of the auxiliary timestamp register.	Go
B50h	MAC_Timestamp_Ingress_Asym_Corr	The MAC Timestamp Ingress Asymmetry Correction register contains the Ingress Asymmetry Correction value to be used while updating correction field in PDelay_Resp PTP messages.	Go
B54h	MAC_Timestamp_Egress_Asym_Corr	The MAC Timestamp Egress Asymmetry Correction register contains the Egress Asymmetry Correction value to be used while updating the correction field in PDelay_Req PTP messages.	Go
B58h	MAC_Timestamp_Ingress_Corr_Nanosecond	This register contains the correction value in nanoseconds to be used with the captured timestamp value in the ingress path.	Go
B5Ch	MAC_Timestamp_Egress_Corr_Nanosecond	This register contains the correction value in nanoseconds to be used with the captured timestamp value in the egress path.	Go
B60h	MAC_Timestamp_Ingress_Corr_Subnanosec	This register contains the sub-nanosecond part of the correction value to be used with the captured timestamp value, for ingress direction.	Go
B64h	MAC_Timestamp_Egress_Corr_Subnanosec	This register contains the sub-nanosecond part of the correction value to be used with the captured timestamp value, for egress direction.	Go
B70h	MAC_PPS_Control	PPS Control register. Bits[30:24] of this register are valid only when four Flexible PPS outputs are selected. Bits[22:16] are valid only when three or more Flexible PPS outputs are selected. Bits[14:8] are valid only when two or more Flexible PPS outputs are selected. Bits[6:4] are valid only when Flexible PPS feature is selected.	Go
B80h	MAC_PPS0_Target_Time_Seconds	The PPS Target Time Seconds register, along with PPS Target Time Nanoseconds register, is used to schedule an interrupt event [Bit 1 of MAC_Timestamp_Status] when the system time exceeds the value programmed in these registers.	Go
B84h	MAC_PPS0_Target_Time_Nanoseconds	PPS0 Target Time Nanoseconds register.	Go
B88h	MAC_PPS0_Interval	The PPS0 Interval register contains the number of units of sub-second increment value between the rising edges of PPS0 signal output (ptp_pps_o[0]).	Go
B8Ch	MAC_PPS0_Width	The PPS0 Width register contains the number of units of sub-second increment value between the rising and corresponding falling edges of PPS0 signal output (ptp_pps_o[0]).	Go
B90h	MAC_PPS1_Target_Time_Seconds	The PPS Target Time Seconds register, along with PPS Target Time Nanoseconds register, is used to schedule an interrupt event [Bit 1 of MAC_Timestamp_Status] when the system time exceeds the value programmed in these registers.	Go
B94h	MAC_PPS1_Target_Time_Nanoseconds	PPS0 Target Time Nanoseconds register.	Go
B98h	MAC_PPS1_Interval	The PPS0 Interval register contains the number of units of sub-second increment value between the rising edges of PPS0 signal output (ptp_pps_o[0]).	Go
B9Ch	MAC_PPS1_Width	The PPS0 Width register contains the number of units of sub-second increment value between the rising and corresponding falling edges of PPS0 signal output (ptp_pps_o[0]).	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
BC0h	MAC_PTO_Control	This register controls the PTP Offload Engine operation. This register is available only when the Enable PTP Timestamp Offload feature is selected.	Go
BC4h	MAC_Source_Port_Identity0	This register contains Bits[31:0] of the 80-bit Source Port Identity of the PTP node. This register is available only when the Enable PTP Timestamp Offload feature is selected.	Go
BC8h	MAC_Source_Port_Identity1	This register contains Bits[63:32] of the 80-bit Source Port Identity of the PTP node. This register is available only when the Enable PTP Timestamp Offload feature is selected.	Go
BCCh	MAC_Source_Port_Identity2	This register contains Bits[79:64] of the 80-bit Source Port Identity of the PTP node. This register is available only when the Enable PTP Timestamp Offload feature is selected.	Go
BD0h	MAC_Log_Message_Interval	This register contains the periodic intervals for automatic PTP packet generation. This register is available only when the Enable PTP Timestamp Offload feature is selected.	Go
C00h	MTL_Operation_Mode	The Operation Mode register establishes the Transmit and Receive operating modes and commands.	Go
C08h	MTL_DBG_CTL	The FIFO Debug Access Control and Status register controls the operation mode of FIFO debug access.	Go
C0Ch	MTL_DBG_STS	The FIFO Debug Status register contains the status of FIFO debug access.	Go
C10h	MTL_FIFO_Debug_Data	The FIFO Debug Data register contains the data to be written to or read from the FIFOs.	Go
C20h	MTL_Interrupt_Status	The software driver (application) reads this register during interrupt service routine or polling to determine the interrupt status of MTL queues and the MAC.	Go
C30h	MTL_RxQ_DMA_Map0	The Receive Queue and DMA Channel Mapping 0 register is reserved in EQOS-CORE and EQOS-MTL configurations.	Go
D00h	MTL_TxQ0_Operation_Mode	The Queue 0 Transmit Operation Mode register establishes the Transmit queue operating modes and commands.	Go
D04h	MTL_TxQ0_Underflow	The Queue 0 Underflow Counter register contains the counter for packets aborted because of Transmit queue underflow and packets missed because of Receive queue packet flush	Go
D08h	MTL_TxQ0_Debug	The Queue 0 Transmit Debug register gives the debug status of various blocks related to the Transmit queue.	Go
D14h	MTL_TxQ0_ETS_Status	The Queue 0 ETS Status register provides the average traffic transmitted in Queue 0.	Go
D18h	MTL_TxQ0_Quantum_Weight	The Queue 0 Quantum or Weights register contains the quantum value for Deficit Weighted Round Robin (DWRR), weights for the Weighted Round Robin (WRR), and Weighted Fair Queuing (WFQ) for Queue 0.	Go
D2Ch	MTL_Q0_Interrupt_Control_Status	This register contains the interrupt enable and status bits for the queue 0 interrupts.	Go
D30h	MTL_RxQ0_Operation_Mode	The Queue 0 Receive Operation Mode register establishes the Receive queue operating modes and command. The RFA and RFD fields are not backward compatible with the RFA and RFD fields of 4.00a release	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
D34h	MTL_RxQ0_Missed_Packet_Overflow_Cnt	The Queue 0 Missed Packet and Overflow Counter register contains the counter for packets missed because of Receive queue packet flush and packets discarded because of Receive queue overflow.	Go
D38h	MTL_RxQ0_Debug	The Queue 0 Receive Debug register gives the debug status of various blocks related to the Receive queue.	Go
D3Ch	MTL_RxQ0_Control	The Queue Receive Control register controls the receive arbitration and passing of received packets to the application.	Go
D40h	MTL_TxQ1_Operation_Mode	The Queue 1 Transmit Operation Mode register establishes the Transmit queue operating modes and commands.	Go
D44h	MTL_TxQ1_Underflow	The Queue 1 Underflow Counter register contains the counter for packets aborted because of Transmit queue underflow and packets missed because of Receive queue packet flush	Go
D48h	MTL_TxQ1_Debug	The Queue 1 Transmit Debug register gives the debug status of various blocks related to the Transmit queue.	Go
D54h	MTL_TxQ1_ETS_Status	The Queue 1 ETS Status register provides the average traffic transmitted in Queue 1.	Go
D58h	MTL_TxQ1_Quantum_Weight	The Queue 1 idleSlopeCredit, Quantum or Weights register provides the average traffic transmitted in Queue 1.	Go
D6Ch	MTL_Q1_Interrupt_Control_Status	This register contains the interrupt enable and status bits for the queue 1 interrupts.	Go
D70h	MTL_RxQ1_Operation_Mode	The Queue 1 Receive Operation Mode register establishes the Receive queue operating modes and command. The RFA and RFD fields are not backward compatible with the RFA and RFD fields of 4.00a release	Go
D74h	MTL_RxQ1_Missed_Packet_Overflow_Cnt	The Queue 1 Missed Packet and Overflow Counter register contains the counter for packets missed because of Receive queue packet flush and packets discarded because of Receive queue overflow.	Go
D78h	MTL_RxQ1_Debug	The Queue 1 Receive Debug register gives the debug status of various blocks related to the Receive queue.	Go
D7Ch	MTL_RxQ1_Control	The Queue Receive Control register controls the receive arbitration and passing of received packets to the application.	Go
1000h	DMA_Mode	The Bus Mode register establishes the bus operating modes for the DMA.	Go
1004h	DMA_SysBus_Mode	The System Bus mode register controls the behavior of the AHB or AXI master. It mainly controls burst splitting and number of outstanding requests.	Go
1008h	DMA_Interrupt_Status	The application reads this Interrupt Status register during interrupt service routine or polling to determine the interrupt status of DMA channels, MTL queues, and the MAC.	Go
100Ch	DMA_Debug_Status0	The Debug Status 0 register gives the Receive and Transmit process status for DMA Channel 0-Channel 2 for debugging purpose.	Go
1100h	DMA_CH0_Control	The DMA Channeli Control register specifies the MSS value for segmentation, length to skip between two descriptors, and also the features such as header splitting and 8xPBL mode.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
1104h	DMA_CH0_Tx_Control	The DMA Channeli Transmit Control register controls the Tx features such as PBL, TCP segmentation, and Tx Channel weights.	Go
1108h	DMA_CH0_Rx_Control	The DMA Channeli Receive Control register controls the Rx features such as PBL, buffer size, and extended status.	Go
1114h	DMA_CH0_TxDesc_List_Address	The Channeli Tx Descriptor List Address register points the DMA to the start of Transmit descriptor list. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LSB to low. You can write to this register only when the Tx DMA has stopped, that is, the ST bit is set to zero in DMA_CH0_Tx_Control register. When stopped, this register can be written with a new descriptor list address. When you set the ST bit to 1, the DMA takes the newly-programmed descriptor base address. If this register is not changed when the ST bit is set to 0, the DMA takes the descriptor address where it was stopped earlier.	Go
111Ch	DMA_CH0_RxDesc_List_Address	The Channeli Rx Descriptor List Address register points the DMA to the start of Receive descriptor list. This register points to the start of the Receive Descriptor List. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LS bits low. Writing to this register is permitted only when reception is stopped. When stopped, this register must be written to before the receive Start command is given. You can write to this register only when Rx DMA has stopped, that is, SR bit is set to zero in DMA_CH0_Rx_Control register. When stopped, this register can be written with a new descriptor list address. When you set the SR bit to 1, the DMA takes the newly programmed descriptor base address.	Go
1120h	DMA_CH0_TxDesc_Tail_Pointer	The Channeli Tx Descriptor Tail Pointer register points to an offset from the base and indicates the location of the last valid descriptor.	Go
1128h	DMA_CH0_RxDesc_Tail_Pointer	The Channeli Rx Descriptor Tail Pointer Points to an offset from the base and indicates the location of the last valid descriptor.	Go
112Ch	DMA_CH0_TxDesc_Ring_Length	The Tx Descriptor Ring Length register contains the length of the Transmit descriptor ring.	Go
1130h	DMA_CH0_RxDesc_Ring_Length	The Channeli Rx Descriptor Ring Length register contains the length of the Receive descriptor circular ring.	Go
1134h	DMA_CH0_Interrupt_Enable	The Channeli Interrupt Enable register enables the interrupts reported by the Status register.	Go
1138h	DMA_CH0_Rx_Interrupt_Watchdog_Timer	The Receive Interrupt Watchdog Timer register indicates the watchdog timeout for Receive Interrupt (RI) from the DMA. When this register is written with a non-zero value, it enables the watchdog timer for the RI bit of the DMA_CHi_Status register.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
1144h	DMA_CH0_Current_App_TxDesc	The Channeli Current Application Transmit Descriptor register points to the current Transmit descriptor read by the DMA.	Go
114Ch	DMA_CH0_Current_App_RxDesc	The Channeli Current Application Receive Descriptor register points to the current Receive descriptor read by the DMA.	Go
1154h	DMA_CH0_Current_App_TxBuffer	The Channeli Current Application Transmit Buffer Address register points to the current Tx buffer address read by the DMA.	Go
115Ch	DMA_CH0_Current_App_RxBuffer	The Channel 0 Current Application Receive Buffer Address register points to the current Rx buffer address read by the DMA.	Go
1160h	DMA_CH0_Status	The software driver (application) reads the Status register during interrupt service routine or polling to determine the status of the DMA. Note: The number of DMA_CH[n]_Status register in the configuration is the higher of number of Rx DMA Channels and Tx DMA Channels.	Go
1164h	DMA_CH0_Miss_Frame_Cnt	This register has the number of packet counter that got dropped by the DMA either due to Bus Error or due to programming RPF field in DMA_CH${i}_Rx_Control register.	Go
116Ch	DMA_CH0_RX_ERI_Cnt		Go
1180h	DMA_CH1_Control	The DMA Channeli Control register specifies the MSS value for segmentation, length to skip between two descriptors, and also the features such as header splitting and 8xPBL mode.	Go
1184h	DMA_CH1_Tx_Control	The DMA Channeli Transmit Control register controls the Tx features such as PBL, TCP segmentation, and Tx Channel weights.	Go
1188h	DMA_CH1_Rx_Control	The DMA Channeli Receive Control register controls the Rx features such as PBL, buffer size, and extended status.	Go
1194h	DMA_CH1_TxDesc_List_Address	The Channeli Tx Descriptor List Address register points the DMA to the start of Transmit descriptor list. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LSB to low. You can write to this register only when the Tx DMA has stopped, that is, the ST bit is set to zero in DMA_CH0_Tx_Control register. When stopped, this register can be written with a new descriptor list address. When you set the ST bit to 1, the DMA takes the newly-programmed descriptor base address. If this register is not changed when the ST bit is set to 0, the DMA takes the descriptor address where it was stopped earlier.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
119Ch	DMA_CH1_RxDesc_List_Address	The Channeli Rx Descriptor List Address register points the DMA to the start of Receive descriptor list. This register points to the start of the Receive Descriptor List. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LS bits low. Writing to this register is permitted only when reception is stopped. When stopped, this register must be written to before the receive Start command is given. You can write to this register only when Rx DMA has stopped, that is, SR bit is set to zero in DMA_CH0_Rx_Control register. When stopped, this register can be written with a new descriptor list address. When you set the SR bit to 1, the DMA takes the newly programmed descriptor base address.	Go
11A0h	DMA_CH1_TxDesc_Tail_Pointer	The Channeli Tx Descriptor Tail Pointer register points to an offset from the base and indicates the location of the last valid descriptor.	Go
11A8h	DMA_CH1_RxDesc_Tail_Pointer	The Channeli Rx Descriptor Tail Pointer Points to an offset from the base and indicates the location of the last valid descriptor.	Go
11ACh	DMA_CH1_TxDesc_Ring_Length	The Tx Descriptor Ring Length register contains the length of the Transmit descriptor ring.	Go
11B0h	DMA_CH1_RxDesc_Ring_Length	The Channeli Rx Descriptor Ring Length register contains the length of the Receive descriptor circular ring.	Go
11B4h	DMA_CH1_Interrupt_Enable	The Channeli Interrupt Enable register enables the interrupts reported by the Status register.	Go
11B8h	DMA_CH1_Rx_Interrupt_Watchdog_Timer	The Receive Interrupt Watchdog Timer register indicates the watchdog timeout for Receive Interrupt (RI) from the DMA. When this register is written with a non-zero value, it enables the watchdog timer for the RI bit of the DMA_CHi_Status register.	Go
11C4h	DMA_CH1_Current_App_TxDesc	The Channeli Current Application Transmit Descriptor register points to the current Transmit descriptor read by the DMA.	Go
11CCh	DMA_CH1_Current_App_RxDesc	The Channeli Current Application Receive Descriptor register points to the current Receive descriptor read by the DMA.	Go
11D4h	DMA_CH1_Current_App_TxBuffer	The Channeli Current Application Transmit Buffer Address register points to the current Tx buffer address read by the DMA.	Go
11DCh	DMA_CH1_Current_App_RxBuffer	The Channel 0 Current Application Receive Buffer Address register points to the current Rx buffer address read by the DMA.	Go
11E0h	DMA_CH1_Status	The software driver (application) reads the Status register during interrupt service routine or polling to determine the status of the DMA. Note: The number of DMA_CH[n]_Status register in the configuration is the higher of number of Rx DMA Channels and Tx DMA Channels.	Go
11E4h	DMA_CH1_Miss_Frame_Cnt	This register has the number of packet counter that got dropped by the DMA either due to Bus Error or due to programming RPF field in DMA_CH${i}_Rx_Control register.	Go

Table 43-92. EMAC_REGS Registers (continued)

Offset	Acronym	Register Name	Section
11ECh	DMA_CH1_RX_ERI_Cnt		Go

Complex bit access types are encoded to fit into small table cells. Table 43-93 shows the codes that are used for access types in this section.

Table 43-93. EMAC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

43.7.3.1 MAC_Configuration Register (Offset = 0h) [Reset = 8000h]

MAC_Configuration is shown in Figure 43-40 and described in Table 43-94.

Return to the Summary Table.

The MAC Configuration Register establishes the operating mode of the MAC.

Figure 43-40. MAC_Configuration Register

31	30	29	28	27	26	25	24
ARPEN	SARC			IPC	IPG		
R/W-0h	R/W-0h			R/W-0h	R/W-0h		
23	22	21	20	19	18	17	16
GPSLCE	S2KP	CST	ACS	WD	RESERVED	JD	JE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
PS	FES	DM	LM	ECRSFD	DO	DCRS	DR
R-1h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	BL		DC	PRELEN		TE	RE
R-0h	R/W-0h		R/W-0h	R/W-0h		R/W-0h	R/W-0h

Table 43-94. MAC_Configuration Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ARPEN	R/W	0h	ARP Offload Enable When this bit is set, the MAC can recognize an incoming ARP request packet and schedules the ARP packet for transmission. It forwards the ARP packet to the application and also indicate the events in the RxStatus. When this bit is reset, the MAC receiver does not recognize any ARP packet and indicates them as Type frame in the RxStatus. This bit is available only when the Enable IPv4 ARP Offload is selected. 0h = ARP Offload is disabled : 0x0 1h = ARP Offload is enabled : 0x1

Table 43-94. MAC_Configuration Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
30-28	SARC	R/W	0h	Source Address Insertion or Replacement Control This field controls the source address insertion or replacement for all transmitted packets. Bit 30 specifies which MAC Address register (0 or 1) is used for source address insertion or replacement based on the values of Bits[29:28]: 2'b0x: - The SA Insertion control is to be programmed in Bits[25:23] of TDES3 in first Transmit Descriptor of the packet. 2'b10: - If Bit 30 is set to 0, the MAC inserts the content of the MAC Address 0 registers (MAC registers 192 and 193) in the SA field of all transmitted packets. - If Bit 30 is set to 1 and the Enable MAC Address Register 1 option is selected while configuring the core, the MAC inserts the content of the MAC Address 1 registers (MAC registers 194 and 195) in the SA field of all transmitted packets. 2'b11: - If Bit 30 is set to 0, the MAC replaces the content of the MAC Address 0 registers (MAC registers 192 and 193) in the SA field of all transmitted packets. - If Bit 30 is set to 1 and the MAC Address Register 1 is enabled, the MAC replaces the content of the MAC Address 1 registers (MAC registers 194 and 195) in the SA field of all transmitted packets. Note: - Changes to this field take effect only on the start of a packet. If you write to this register field when a packet is being transmitted, only the subsequent packet can use the updated value, that is, the current packet does not use the updated value. 0h = mti_sa_ctrl_i and ati_sa_ctrl_i input signals control the SA field generation : 0x0 2h = Contents of MAC Addr-0 inserted in SA field : 0x2 3h = Contents of MAC Addr-0 replaces SA field : 0x3 6h = Contents of MAC Addr-1 inserted in SA field : 0x6 7h = Contents of MAC Addr-1 replaces SA field : 0x7
27	IPC	R/W	0h	Checksum Offload When set, this bit enables the IPv4 header checksum checking and IPv4 or IPv6 TCP, UDP, or ICMP payload checksum checking. When this bit is reset, the COE function in the receiver is disabled. The Layer 3 and Layer 4 Packet Filter and Enable Split Header features automatically selects the IPC Full Checksum Offload Engine on the Receive side. When any of these features are enabled, you must set the IPC bit. 0h = IP header/payload checksum checking is disabled : 0x0 1h = IP header/payload checksum checking is enabled : 0x1
26-24	IPG	R/W	0h	Inter-Packet Gap These bits control the minimum IPG between packets during transmission. This range of minimum IPG is valid in full-duplex mode. In the half-duplex mode, the minimum IPG can be configured only for 64-bit times (IPG = 100). Lower values are not considered. When a JAM pattern is being transmitted because of backpressure activation, the MAC does not consider the minimum IPG. The above function (IPG less than 96 bit times) is valid only when EIPGEN bit in MAC_Ext_Configuration register is reset. When EIPGEN is set, then the minimum IPG (greater than 96 bit times) is controlled as per the description given in EIPG field in MAC_Ext_Configuration register. 0h = 96 bit times IPG : 0x0 1h = 88 bit times IPG : 0x1 2h = 80 bit times IPG : 0x2 3h = 72 bit times IPG : 0x3 4h = 64 bit times IPG : 0x4 5h = 56 bit times IPG : 0x5 6h = 48 bit times IPG : 0x6 7h = 40 bit times IPG : 0x7

Table 43-94. MAC_Configuration Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	GPSLCE	R/W	0h	Giant Packet Size Limit Control Enable When this bit is set, the MAC considers the value in GPSL field in MAC_Ext_Configuration register to declare a received packet as Giant packet. This field must be programmed to more than 1,518 bytes. Otherwise, the MAC considers 1,518 bytes as giant packet limit. When this bit is reset, the MAC considers a received packet as Giant packet when its size is greater than 1,518 bytes (1522 bytes for tagged packet). The watchdog timeout limit, Jumbo Packet Enable and 2K Packet Enable have higher precedence over this bit, that is the MAC considers a received packet as Giant packet when its size is greater than 9,018 bytes (9,022 bytes for tagged packet) with Jumbo Packet Enabled and greater than 2,000 bytes with 2K Packet Enabled. The watchdog timeout, if enabled, terminates the received packet when watchdog limit is reached. Therefore, the programmed giant packet limit should be less than the watchdog limit to get the giant packet status. 0h = Giant Packet Size Limit Control is disabled : 0x0 1h = Giant Packet Size Limit Control is enabled : 0x1
22	S2KP	R/W	0h	IEEE 802.3as Support for 2K Packets When this bit is set, the MAC considers all packets with up to 2,000 bytes length as normal packets. When the JE bit is not set, the MAC considers all received packets of size more than 2K bytes as Giant packets. When this bit is reset and the JE bit is not set, the MAC considers all received packets of size more than 1,518 bytes (1,522 bytes for tagged) as giant packets. For more information about how the setting of this bit and the JE bit impact the Giant packet status, see the Table, Gaint Packet Status based on S2KP and JE Bits. Note: When the JE bit is set, setting this bit has no effect on the giant packet status. 0h = Support upto 2K packet is disabled : 0x0 1h = Support upto 2K packet is Enabled : 0x1
21	CST	R/W	0h	CRC stripping for Type packets When this bit is set, the last four bytes (FCS) of all packets of Ether type (type field greater than 1,536) are stripped and dropped before forwarding the packet to the application. Note: For information about how the settings of the ACS bit and this bit impact the packet length, see the Table, Packet Length based on the CST and ACS Bits. 0h = CRC stripping for Type packets is disabled : 0x0 1h = CRC stripping for Type packets is enabled : 0x1
20	ACS	R/W	0h	Automatic Pad or CRC Stripping When this bit is set, the MAC strips the Pad or FCS field on the incoming packets only if the value of the length field is less than 1,536 bytes. All received packets with length field greater than or equal to 1,536 bytes are passed to the application without stripping the Pad or FCS field. When this bit is reset, the MAC passes all incoming packets to the application, without any modification. Note: For information about how the settings of CST bit and this bit impact the packet length, see the Table, Packet Length based on the CST and ACS Bit . 0h = Automatic Pad or CRC Stripping is disabled : 0x0 1h = Automatic Pad or CRC Stripping is enabled : 0x1
19	WD	R/W	0h	Watchdog Disable When this bit is set, the MAC disables the watchdog timer on the receiver. The MAC can receive packets of up to 16,383 bytes. When this bit is reset, the MAC does not allow more than 2,048 bytes (10,240 if JE is set high) of the packet being received. The MAC cuts off any bytes received after 2,048 bytes. 0h = Watchdog is enabled : 0x0 1h = Watchdog is disabled : 0x1

Table 43-94. MAC_Configuration Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	RESERVED	R	0h	Reserved.
17	JD	R/W	0h	Jabber Disable When this bit is set, the MAC disables the jabber timer on the transmitter. The MAC can transfer packets of up to 16,383 bytes. When this bit is reset, if the application sends more than 2,048 bytes of data (10,240 if JE is set high) during transmission, the MAC does not send rest of the bytes in that packet. 0h = Jabber is enabled : 0x0 1h = Jabber is disabled : 0x1
16	JE	R/W	0h	Jumbo Packet Enable When this bit is set, the MAC allows jumbo packets of 9,018 bytes (9,022 bytes for VLAN tagged packets) without reporting a giant packet error in the Rx packet status. 0h = Jumbo packet is disabled : 0x0 1h = Jumbo packet is enabled : 0x1
15	PS	R	1h	Port Select This bit selects the Ethernet line speed. This bit, along with Bit 14, selects the exact line speed. In the 10/100 Mbps-only (always 1) or 1000 Mbps-only (always 0) configurations, this bit is read-only (RO) with appropriate value. In default 10/100/1000 Mbps configurations, this bit is read-write (R/W). The mac_speed_o[1] signal reflects the value of this bit. 0h = For 1000 or 2500 Mbps operations : 0x0 1h = For 10 or 100 Mbps operations : 0x1
14	FES	R/W	0h	Speed This bit selects the speed mode. The mac_speed_o[0] signal reflects the value of this bit. 0h = 10 Mbps when PS bit is 1 and 1 Gbps when PS bit is 0 : 0x0 1h = 100 Mbps when PS bit is 1 and 2.5 Gbps when PS bit is 0 : 0x1
13	DM	R/W	0h	Duplex Mode When this bit is set, the MAC operates in the full-duplex mode in which it can transmit and receive simultaneously. This bit is RO with default value of 1'b1 in the full-duplex-only configurations. 0h = Half-duplex mode : 0x0 1h = Full-duplex mode : 0x1
12	LM	R/W	0h	Loopback Mode When this bit is set, the MAC operates in the loopback mode at GMII or MII. The (G)MII Rx clock input (clk_rx_i) is required for the loopback to work properly. This is because the Tx clock is not internally looped back. 0h = Loopback is disabled : 0x0 1h = Loopback is enabled : 0x1
11	ECRSFD	R/W	0h	Enable Carrier Sense Before Transmission in Full-Duplex Mode When this bit is set, the MAC transmitter checks the CRS signal before packet transmission in the full-duplex mode. The MAC starts the transmission only when the CRS signal is low. When this bit is reset, the MAC transmitter ignores the status of the CRS signal. 0h = ECRSFD is disabled : 0x0 1h = ECRSFD is enabled : 0x1
10	DO	R/W	0h	Disable Receive Own When this bit is set, the MAC disables the reception of packets when the gmii_txen_o is asserted in the half-duplex mode. When this bit is reset, the MAC receives all packets given by the PHY. This bit is not applicable in the full-duplex mode. 0h = Enable Receive Own : 0x0 1h = Disable Receive Own : 0x1

Table 43-94. MAC_Configuration Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	DCRS	R/W	0h	Disable Carrier Sense During Transmission When this bit is set, the MAC transmitter ignores the (G)MII CRS signal during packet transmission in the half-duplex mode. As a result, no errors are generated because of Loss of Carrier or No Carrier during transmission. When this bit is reset, the MAC transmitter generates errors because of Carrier Sense. The MAC can even abort the transmission. 0h = Enable Carrier Sense During Transmission : 0x0 1h = Disable Carrier Sense During Transmission : 0x1
8	DR	R/W	0h	Disable Retry When this bit is set, the MAC attempts only one transmission. When a collision occurs on the GMII or MII interface, the MAC ignores the current packet transmission and reports a Packet Abort with excessive collision error in the Tx packet status. When this bit is reset, the MAC retries based on the settings of the BL field. This bit is applicable only in the half-duplex mode. 0h = Enable Retry : 0x0 1h = Disable Retry : 0x1
7	RESERVED	R	0h	Reserved.
6-5	BL	R/W	0h	Back-Off Limit The back-off limit determines the random integer number (r) of slot time delays (4,096 bit times for 1000/2500 Mbps 512 bit times for 10/100 Mbps) for which the MAC waits before rescheduling a transmission attempt during retries after a collision. n = retransmission attempt. The random integer r takes the value in the range 0 <= r < 2^k This bit is applicable only in the half-duplex mode. 0h = k = min(n,10) : 0x0 1h = k = min(n,8) : 0x1 2h = k = min(n,4) : 0x2 3h = k = min(n,1) : 0x3
4	DC	R/W	0h	Deferral Check When this bit is set, the deferral check function is enabled in the MAC. The MAC issues a Packet Abort status, along with the excessive deferral error bit set in the Tx packet status, when the Tx state machine is deferred for more than 24,288 bit times in 10 or 100 Mbps mode. If the MAC is configured for 1000/2500 Mbps operation, the threshold for deferral is 155,680 bits times. Deferral begins when the transmitter is ready to transmit, but it is prevented because of an active carrier sense signal (CRS) on GMII or MII. The defer time is not cumulative. For example, if the transmitter defers for 10,000 bit times because the CRS signal is active and the CRS signal becomes inactive, the transmitter transmits and collision happens. Because of collision, the transmitter needs to back off and then defer again after back off completion. In such a scenario, the deferral timer is reset to 0, and it is restarted. When this bit is reset, the deferral check function is disabled and the MAC defers until the CRS signal goes inactive. This bit is applicable only in the half-duplex mode. 0h = Deferral check function is disabled : 0x0 1h = Deferral check function is enabled : 0x1
3-2	PRELEN	R/W	0h	Preamble Length for Transmit packets These bits control the number of preamble bytes that are added to the beginning of every Tx packet. The preamble reduction occurs only when the MAC is operating in the full-duplex mode. 0h = 7 bytes of preamble : 0x0 1h = 5 bytes of preamble : 0x1 2h = 3 bytes of preamble : 0x2 3h = Reserved : 0x3

Table 43-94. MAC_Configuration Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	TE	R/W	0h	Transmitter Enable When this bit is set, the Tx state machine of the MAC is enabled for transmission on the GMII or MII interface. When this bit is reset, the MAC Tx state machine is disabled after it completes the transmission of the current packet. The Tx state machine does not transmit any more packets. 0h = Transmitter is disabled : 0x0 1h = Transmitter is enabled : 0x1
0	RE	R/W	0h	Receiver Enable When this bit is set, the Rx state machine of the MAC is enabled for receiving packets from the GMII or MII interface. When this bit is reset, the MAC Rx state machine is disabled after it completes the reception of the current packet. The Rx state machine does not receive any more packets from the GMII or MII interface. 0h = Receiver is disabled : 0x0 1h = Receiver is enabled : 0x1

43.7.3.2 MAC_Ext_Configuration Register (Offset = 4h) [Reset = 0h]

MAC_Ext_Configuration is shown in Figure 43-41 and described in Table 43-95.

Return to the Summary Table.

The MAC Extended Configuration Register establishes the operating mode of the MAC.

Figure 43-41. MAC_Ext_Configuration Register

31	30	29	28	27	26	25	24
RESERVED		EIPG					EIPGEN
R-0h		R/W-0h					R/W-0h
23	22	21	20	19	18	17	16
RESERVED	HDSMS			RESERVED	USP	SPEN	DCRCC
R-0h	R/W-0h			R-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED		GPSL					
R-0h		R/W-0h					
7	6	5	4	3	2	1	0
GPSL							
R/W-0h							

Table 43-95. MAC_Ext_Configuration Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved.
29-25	EIPG	R/W	0h	Extended Inter-Packet Gap The value in this field is applicable when the EIPGEN bit is set. This field (as Most Significant bits), along with IPG field in MAC_Configuration register, gives the minimum IPG greater than 96 bit times in steps of 8 bit times: {EIPG, IPG} 8'h00 - 104 bit times 8'h01 - 112 bit times 8'h02 - 120 bit times ---------- 8'hFF - 2144 bit times
24	EIPGEN	R/W	0h	Extended Inter-Packet Gap Enable When this bit is set, the MAC interprets EIPG field and IPG field in MAC_Configuration register together as minimum IPG greater than 96 bit times in steps of 8 bit times. When this bit is reset, the MAC ignores EIPG field and interprets IPG field in MAC_Configuration register as minimum IPG less than or equal to 96 bit times in steps of 8 bit times. Note: The extended Inter-Packet Gap feature must be enabled when operating in Full-Duplex mode only. There may be undesirable effects on back-pressure function and frame transmission if it is enabled in Half-Duplex mode. 0h = Extended Inter-Packet Gap is disabled : 0x0 1h = Extended Inter-Packet Gap is enabled : 0x1
23	RESERVED	R	0h	Reserved.
22-20	HDSMS	R/W	0h	Maximum Size for Splitting the Header Data These bits indicate the maximum header size allowed for splitting the header data in the received packet. 0h = Maximum Size for Splitting the Header Data is 64 bytes : 0x0 1h = Maximum Size for Splitting the Header Data is 128 bytes : 0x1 2h = Maximum Size for Splitting the Header Data is 256 bytes : 0x2 3h = Maximum Size for Splitting the Header Data is 512 bytes : 0x3 4h = Maximum Size for Splitting the Header Data is 1024 bytes : 0x4 5h = Reserved : 0x5

Table 43-95. MAC_Ext_Configuration Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	RESERVED	R	0h	Reserved.
18	USP	R/W	0h	Unicast Slow Protocol Packet Detect When this bit is set, the MAC detects the Slow Protocol packets with unicast address of the station specified in the MAC_Address0_High and MAC_Address0_Low registers. The MAC also detects the Slow Protocol packets with the Slow Protocols multicast address (01-80-C2-00-00-02). When this bit is reset, the MAC detects only Slow Protocol packets with the Slow Protocol multicast address specified in the IEEE 802.3-2008, Section 5. 0h = Unicast Slow Protocol Packet Detection is disabled : 0x0 1h = Unicast Slow Protocol Packet Detection is enabled : 0x1
17	SPEN	R/W	0h	Slow Protocol Detection Enable When this bit is set, MAC processes the Slow Protocol packets (Ether Type 0x8809) and provides the Rx status. The MAC discards the Slow Protocol packets with invalid sub-types. When this bit is reset, the MAC forwards all error-free Slow Protocol packets to the application. The MAC considers such packets as normal Type packets. 0h = Slow Protocol Detection is disabled : 0x0 1h = Slow Protocol Detection is enabled : 0x1
16	DCRCC	R/W	0h	Disable CRC Checking for Received Packets When this bit is set, the MAC receiver does not check the CRC field in the received packets. When this bit is reset, the MAC receiver always checks the CRC field in the received packets. 0h = CRC Checking is enabled : 0x0 1h = CRC Checking is disabled : 0x1
15-14	RESERVED	R	0h	Reserved.
13-0	GPSL	R/W	0h	Giant Packet Size Limit If the received packet size is greater than the value programmed in this field in units of bytes, the MAC declares the received packet as Giant packet. The value programmed in this field must be greater than or equal to 1,518 bytes. Any other programmed value is considered as 1,518 bytes. For VLAN tagged packets, the MAC adds 4 bytes to the programmed value. When the Enable Double VLAN Processing option is selected, the MAC adds 8 bytes to the programmed value for double VLAN tagged packets. The value in this field is applicable when the GPSLCE bit is set in MAC_Configuration register.

43.7.3.3 MAC_Packet_Filter Register (Offset = 8h) [Reset = 0h]

MAC_Packet_Filter is shown in Figure 43-42 and described in Table 43-96.

Return to the Summary Table.

The MAC Packet Filter register contains the filter controls for receiving packets. Some of the controls from this register go to the address check block of the MAC which performs the first level of address filtering. The second level of filtering is performed on the incoming packet based on other controls such as Pass Bad Packets and Pass Control Packets.

Figure 43-42. MAC_Packet_Filter Register

31	30	29	28	27	26	25	24
RA	RESERVED						
R/W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED		DNTU	IPFE	RESERVED			VTFE
R-0h		R/W-0h	R/W-0h	R-0h			R/W-0h
15	14	13	12	11	10	9	8
RESERVED					HPF	SAF	SAIF
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
PCF		DBF	PM	DAIF	HMC	HUC	PR
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 43-96. MAC_Packet_Filter Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RA	R/W	0h	Receive All When this bit is set, the MAC Receiver module passes all received packets to the application, irrespective of whether they pass the address filter or not. The result of the SA or DA filtering is updated (pass or fail) in the corresponding bit in the Rx Status Word. When this bit is reset, the Receiver module passes only those packets to the application that pass the SA or DA address filter. 0h = Receive All is disabled : 0x0 1h = Receive All is enabled : 0x1
30-22	RESERVED	R	0h	Reserved.
21	DNTU	R/W	0h	Drop Non-TCP/UDP over IP Packets When this bit is set, the MAC drops the non-TCP or UDP over IP packets. The MAC forward only those packets that are processed by the Layer 4 filter. When this bit is reset, the MAC forwards all non-TCP or UDP over IP packets. 0h = Forward Non-TCP/UDP over IP Packets : 0x0 1h = Drop Non-TCP/UDP over IP Packets : 0x1
20	IPFE	R/W	0h	Layer 3 and Layer 4 Filter Enable When this bit is set, the MAC drops packets that do not match the enabled Layer 3 and Layer 4 filters. If Layer 3 or Layer 4 filters are not enabled for matching, this bit does not have any effect. When this bit is reset, the MAC forwards all packets irrespective of the match status of the Layer 3 and Layer 4 fields. 0h = Layer 3 and Layer 4 Filters are disabled : 0x0 1h = Layer 3 and Layer 4 Filters are enabled : 0x1
19-17	RESERVED	R	0h	Reserved.

Table 43-96. MAC_Packet_Filter Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	VTFE	R/W	0h	VLAN Tag Filter Enable When this bit is set, the MAC drops the VLAN tagged packets that do not match the VLAN Tag. When this bit is reset, the MAC forwards all packets irrespective of the match status of the VLAN Tag. 0h = VLAN Tag Filter is disabled : 0x0 1h = VLAN Tag Filter is enabled : 0x1
15-11	RESERVED	R	0h	Reserved.
10	HPF	R/W	0h	Hash or Perfect Filter When this bit is set, the address filter passes a packet if it matches either the perfect filtering or hash filtering as set by the HMC or HUC bit. When this bit is reset and the HUC or HMC bit is set, the packet is passed only if it matches the Hash filter. 0h = Hash or Perfect Filter is disabled : 0x0 1h = Hash or Perfect Filter is enabled : 0x1
9	SAF	R/W	0h	Source Address Filter Enable When this bit is set, the MAC compares the SA field of the received packets with the values programmed in the enabled SA registers. If the comparison fails, the MAC drops the packet. When this bit is reset, the MAC forwards the received packet to the application with updated SAF bit of the Rx Status depending on the SA address comparison. Note: According to the IEEE specification, Bit 47 of the SA is reserved. However, in DWC_ether_qos, the MAC compares all 48 bits. The software driver should take this into consideration while programming the MAC address registers for SA. 0h = SA Filtering is disabled : 0x0 1h = SA Filtering is enabled : 0x1
8	SAIF	R/W	0h	SA Inverse Filtering When this bit is set, the Address Check block operates in the inverse filtering mode for SA address comparison. If the SA of a packet matches the values programmed in the SA registers, it is marked as failing the SA Address filter. When this bit is reset, if the SA of a packet does not match the values programmed in the SA registers, it is marked as failing the SA Address filter. 0h = SA Inverse Filtering is disabled : 0x0 1h = SA Inverse Filtering is enabled : 0x1
7-6	PCF	R/W	0h	Pass Control Packets These bits control the forwarding of all control packets (including unicast and multicast Pause packets). 0h = MAC filters all control packets from reaching the application : 0x0 1h = MAC forwards all control packets except Pause packets to the application even if they fail the Address filter : 0x1 2h = MAC forwards all control packets to the application even if they fail the Address filter : 0x2 3h = MAC forwards the control packets that pass the Address filter : 0x3
5	DBF	R/W	0h	Disable Broadcast Packets When this bit is set, the AFM module blocks all incoming broadcast packets. In addition, it overrides all other filter settings. When this bit is reset, the AFM module passes all received broadcast packets. 0h = Enable Broadcast Packets : 0x0 1h = Disable Broadcast Packets : 0x1

Table 43-96. MAC_Packet_Filter Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	PM	R/W	0h	Pass All Multicast When this bit is set, it indicates that all received packets with a multicast destination address (first bit in the destination address field is '1') are passed. When this bit is reset, filtering of multicast packet depends on HMC bit. 0h = Pass All Multicast is disabled : 0x0 1h = Pass All Multicast is enabled : 0x1
3	DAIF	R/W	0h	DA Inverse Filtering When this bit is set, the Address Check block operates in inverse filtering mode for the DA address comparison for both unicast and multicast packets. When this bit is reset, normal filtering of packets is performed. 0h = DA Inverse Filtering is disabled : 0x0 1h = DA Inverse Filtering is enabled : 0x1
2	HMC	R/W	0h	Hash Multicast When this bit is set, the MAC performs the destination address filtering of received multicast packets according to the hash table. When this bit is reset, the MAC performs the perfect destination address filtering for multicast packets, that is, it compares the DA field with the values programmed in DA registers. 0h = Hash Multicast is disabled : 0x0 1h = Hash Multicast is enabled : 0x1
1	HUC	R/W	0h	Hash Unicast When this bit is set, the MAC performs the destination address filtering of unicast packets according to the hash table. When this bit is reset, the MAC performs a perfect destination address filtering for unicast packets, that is, it compares the DA field with the values programmed in DA registers. 0h = Hash Unicast is disabled : 0x0 1h = Hash Unicast is enabled : 0x1
0	PR	R/W	0h	Promiscuous Mode When this bit is set, the Address Filtering module passes all incoming packets irrespective of the destination or source address. The SA or DA Filter Fails status bits of the Rx Status Word are always cleared when PR is set. 0h = Promiscuous Mode is disabled : 0x0 1h = Promiscuous Mode is enabled : 0x1

43.7.3.4 MAC_Watchdog_Timeout Register (Offset = Ch) [Reset = 0h]

MAC_Watchdog_Timeout is shown in Figure 43-43 and described in Table 43-97.

Return to the Summary Table.

The Watchdog Timeout register controls the watchdog timeout for received packets.

Figure 43-43. MAC_Watchdog_Timeout Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED							PWE	RESERVED				WTO			
R-0h							R/W-0h	R-0h				R/W-0h			

Table 43-97. MAC_Watchdog_Timeout Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved.
8	PWE	R/W	0h	Programmable Watchdog Enable When this bit is set and the WD bit of the MAC_Configuration register is reset, the WTO field is used as watchdog timeout for a received packet. When this bit is cleared, the watchdog timeout for a received packet is controlled by setting of WD and JE bits in MAC_Configuration register. 0h = Programmable Watchdog is disabled : 0x0 1h = Programmable Watchdog is enabled : 0x1
7-4	RESERVED	R	0h	Reserved.
3-0	WTO	R/W	0h	Watchdog Timeout When the PWE bit is set and the WD bit of the MAC_Configuration register is reset, this field is used as watchdog timeout for a received packet. If the length of a received packet exceeds the value of this field, such packet is terminated and declared as an error packet. Note: When the PWE bit is set, the value in this field should be more than 1,522 (0x05F2). Otherwise, the IEEE 802.3-specified valid tagged packets are declared as error packets and then dropped. 0h = 2 KB : 0x0 1h = 3 KB : 0x1 2h = 4 KB : 0x2 3h = 5 KB : 0x3 4h = 6 KB : 0x4 5h = 7 KB : 0x5 6h = 8 KB : 0x6 7h = 9 KB : 0x7 8h = 10 KB : 0x8 9h = 11 KB : 0x9 Ah = 12 KB : 0xa Bh = 13 KB : 0xb Ch = 14 KB : 0xc Dh = 15 KB : 0xd Eh = 16383 Bytes : 0xe Fh = Reserved : 0xf

43.7.3.5 MAC_Hash_Table_Reg0 Register (Offset = 10h) [Reset = 0h]

MAC_Hash_Table_Reg0 is shown in Figure 43-44 and described in Table 43-98.

Return to the Summary Table.

The Hash Table Register 0 contains the first 32 bits of the hash table, when the width of the hash table is 128 or 256 bits. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant. The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.
The hash value of the destination address is calculated in the following way:
- Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).
- Perform bitwise reversal for the value obtained in Step 1.
- Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2.

If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values.
If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written.
If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.

Figure 43-44. MAC_Hash_Table_Reg0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HT31T0																															
R/W-0h																															

Table 43-98. MAC_Hash_Table_Reg0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HT31T0	R/W	0h	MAC Hash Table First 32 Bits This field contains the first 32 Bits [31:0] of the Hash table.

43.7.3.6 MAC_Hash_Table_Reg1 Register (Offset = 14h) [Reset = 0h]

MAC_Hash_Table_Reg1 is shown in Figure 43-45 and described in Table 43-99.

Return to the Summary Table.

The Hash Table Register 1 contains the second 32 bits of the hash table. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant.
The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.
The hash value of the destination address is calculated in the following way:
- Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).
- Perform bitwise reversal for the value obtained in Step 1.
- Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2.
If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values.
If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written.
If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.

Figure 43-45. MAC_Hash_Table_Reg1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HT63T32																															
R/W-0h																															

Table 43-99. MAC_Hash_Table_Reg1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HT63T32	R/W	0h	MAC Hash Table Second 32 Bits This field contains the second 32 Bits [63:32] of the Hash table.

43.7.3.7 MAC_VLAN_Tag_Ctrl Register (Offset = 50h) [Reset = 0h]

MAC_VLAN_Tag_Ctrl is shown in Figure 43-46 and described in Table 43-100.

Return to the Summary Table.

This register is the redefined format of the MAC VLAN Tag Register. It is used for indirect addressing. It contains the address offset, command type and Busy Bit for CSR access of the Per VLAN Tag registers.

Figure 43-46. MAC_VLAN_Tag_Ctrl Register

31	30	29	28	27	26	25	24
EIVLRXS	RESERVED	EIVLS		ERIVLT	EDVLP	VTHM	EVLRXS
R/W-0h	R-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	EVLS		RESERVED		ESVL	VTIM	RESERVED
R-0h	R/W-0h		R-0h		R/W-0h	R/W-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	OFS					CT	OB
R-0h	R/W-0h					R/W-0h	R/W-0h

Table 43-100. MAC_VLAN_Tag_Ctrl Register Field Descriptions

Bit	Field	Type	Reset	Description
31	EIVLRXS	R/W	0h	Enable Inner VLAN Tag in Rx Status When this bit is set, the MAC provides the inner VLAN Tag in the Rx status. When this bit is reset, the MAC does not provide the inner VLAN Tag in Rx status. 0h = Inner VLAN Tag in Rx status is disabled : 0x0 1h = Inner VLAN Tag in Rx status is enabled : 0x1
30	RESERVED	R	0h	Reserved.
29-28	EIVLS	R/W	0h	Enable Inner VLAN Tag Stripping on Receive This field indicates the stripping operation on inner VLAN Tag in received packet. 0h = Do not strip : 0x0 1h = Strip if VLAN filter passes : 0x1 2h = Strip if VLAN filter fails : 0x2 3h = Always strip : 0x3
27	ERIVLT	R/W	0h	
26	EDVLP	R/W	0h	Enable Double VLAN Processing When this bit is set, the MAC enables processing of up to two VLAN Tags on Tx and Rx (if present). When this bit is reset, the MAC enables processing of up to one VLAN Tag on Tx and Rx (if present). 0h = Double VLAN Processing is disabled : 0x0 1h = Double VLAN Processing is enabled : 0x1
25	VTHM	R/W	0h	VLAN Tag Hash Table Match Enable When this bit is set, the most significant four bits of CRC of VLAN Tag are used to index the content of the MAC_VLAN_Hash_Table register. A value of 1 in the VLAN Hash Table register, corresponding to the index, indicates that the packet matched the VLAN hash table. When the ETV bit is set, the CRC of the 12-bit VLAN Identifier (VID) is used for comparison. When the ETV bit is reset, the CRC of the 16-bit VLAN tag is used for comparison. When this bit is reset, the VLAN Hash Match operation is not performed. 0h = VLAN Tag Hash Table Match is disabled : 0x0 1h = VLAN Tag Hash Table Match is enabled : 0x1

Table 43-100. MAC_VLAN_Tag_Ctrl Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24	EVLRXS	R/W	0h	Enable VLAN Tag in Rx status When this bit is set, MAC provides the outer VLAN Tag in the Rx status. When this bit is reset, the MAC does not provide the outer VLAN Tag in Rx status. 0h = VLAN Tag in Rx status is disabled : 0x0 1h = VLAN Tag in Rx status is enabled : 0x1
23	RESERVED	R	0h	Reserved.
22-21	EVLS	R/W	0h	Enable VLAN Tag Stripping on Receive This field indicates the stripping operation on the outer VLAN Tag in received packet. 0h = Do not strip : 0x0 1h = Strip if VLAN filter passes : 0x1 2h = Strip if VLAN filter fails : 0x2 3h = Always strip : 0x3
20-19	RESERVED	R	0h	Reserved.
18	ESVL	R/W	0h	Enable S-VLAN When this bit is set, the MAC transmitter and receiver consider the S-VLAN packets (Type = 0x88A8) as valid VLAN tagged packets. 0h = S-VLAN is disabled : 0x0 1h = S-VLAN is enabled : 0x1
17	VTIM	R/W	0h	VLAN Tag Inverse Match Enable When this bit is set, this bit enables the VLAN Tag inverse matching. The packets without matching VLAN Tag are marked as matched. When reset, this bit enables the VLAN Tag perfect matching. The packets with matched VLAN Tag are marked as matched. 0h = VLAN Tag Inverse Match is disabled : 0x0 1h = VLAN Tag Inverse Match is enabled : 0x1
16-7	RESERVED	R	0h	Reserved.
6-2	OFS	R/W	0h	Offset This field holds the address offset of the MAC VLAN Tag Filter Register which the application is trying to access. The width of the field depends on the number of MAC VLAN Tag Registers enabled.
1	CT	R/W	0h	Command Type This bit indicates if the current register access is a read or a write. When set, it indicate a read operation. When reset, it indicates a write operation. 0h = Write operation : 0x0 1h = Read operation : 0x1
0	OB	R/W	0h	Operation Busy This bit is set along with a read or write command for initiating the indirect access to per VLAN Tag Filter register. This bit is reset when the read or write command to per VLAN Tag Filter indirect access register is complete. The next indirect register access can be initiated only after this bit is reset. During a write operation, the bit is reset only after the data has been written into the Per VLAN Tag register. During a read operation, the data should be read from the MAC_VLAN_Tag_Data register only after this bit is reset. 0h = Operation Busy is disabled : 0x0 1h = Operation Busy is enabled : 0x1

43.7.3.8 MAC_VLAN_Tag_Data Register (Offset = 54h) [Reset = 0h]

MAC_VLAN_Tag_Data is shown in Figure 43-47 and described in Table 43-101.

Return to the Summary Table.

This register holds the read/write data for Indirect Access of the Per VLAN Tag registers. During the read access, this field contains valid read data only after the OB bit is reset.
During the write access, this field should be valid prior to setting the OB bit in the MAC_VLAN_Tag_Ctrl Register.

Figure 43-47. MAC_VLAN_Tag_Data Register

31	30	29	28	27	26	25	24
RESERVED						DMACHN	DMACHEN
R-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED			ERIVLT	ERSVLM	DOVLTC	ETV	VEN
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
VID							
R/W-0h							
7	6	5	4	3	2	1	0
VID							
R/W-0h							

Table 43-101. MAC_VLAN_Tag_Data Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved.
25	DMACHN	R/W	0h	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed.
24	DMACHEN	R/W	0h	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. 0h = DMA Channel Number is disabled : 0x0 1h = DMA Channel Number is enabled : 0x1
23-21	RESERVED	R	0h	Reserved.
20	ERIVLT	R/W	0h	Enable Inner VLAN Tag Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). 0h = Inner VLAN tag comparision is disabled : 0x0 1h = Inner VLAN tag comparision is enabled : 0x1
19	ERSVLM	R/W	0h	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets. 0h = Receive S-VLAN Match is disabled : 0x0 1h = Receive S-VLAN Match is enabled : 0x1

Table 43-101. MAC_VLAN_Tag_Data Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	DOVLTC	R/W	0h	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. 0h = VLAN type comparision is enabled : 0x0 1h = VLAN type comparision is disabled : 0x1
17	ETV	R/W	0h	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. 0h = 16 bit VLAN comparision : 0x0 1h = 12 bit VLAN comparision : 0x1
16	VEN	R/W	0h	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. 0h = VLAN Tag is disabled : 0x0 1h = VLAN Tag is enabled : 0x1
15-0	VID	R/W	0h	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set.

43.7.3.9 MAC_VLAN_Hash_Table Register (Offset = 58h) [Reset = 0h]

MAC_VLAN_Hash_Table is shown in Figure 43-48 and described in Table 43-102.

Return to the Summary Table.

When VTHM bit of the MAC_VLAN_Tag register is set, the 16-bit VLAN Hash Table register is used for group address filtering based on the VLAN tag. For hash filtering, the content of the 16-bit VLAN tag or 12-bit VLAN ID (based on the ETV bit of MAC_VLAN_Tag Register) in the incoming packet is passed through the CRC logic. The upper four bits of the calculated CRC are used to index the contents of the VLAN Hash table. For example, a hash value of 4b'1000 selects Bit 8 of the VLAN Hash table.
The hash value of the destination address is calculated in the following way:
- Calculate the 32-bit CRC for the VLAN tag or ID (For steps to calculate CRC32, see Section 3.2.8 of IEEE 802.3).
- Perform bitwise reversal for the value obtained in step 1.
- Take the upper four bits from the value obtained in step 2.
If the VLAN hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[15:8] (in little-endian mode) or Bits[7:0] (in big-endian mode) of this register are written.
- If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.

Figure 43-48. MAC_VLAN_Hash_Table Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																VLHT															
R-0h																R/W-0h															

Table 43-102. MAC_VLAN_Hash_Table Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved.
15-0	VLHT	R/W	0h	VLAN Hash Table This field contains the 16-bit VLAN Hash Table.

43.7.3.10 MAC_VLAN_Incl Register (Offset = 60h) [Reset = 0h]

MAC_VLAN_Incl is shown in Figure 43-49 and described in Table 43-103.

Return to the Summary Table.

The VLAN Tag Inclusion or Replacement register contains the VLAN tag for insertion or replacement in the Transmit packets. It also contains the VLAN tag insertion controls.

Figure 43-49. MAC_VLAN_Incl Register

31	30	29	28	27	26	25	24
BUSY	RDWR	RESERVED					ADDR
R-0h	R/W-0h	R-0h					R/W-0h
23	22	21	20	19	18	17	16
RESERVED		CBTI	VLTI	CSVL	VLP	VLC	
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
VLT							
R/W-0h							
7	6	5	4	3	2	1	0
VLT							
R/W-0h							

Table 43-103. MAC_VLAN_Incl Register Field Descriptions

Bit	Field	Type	Reset	Description
31	BUSY	R	0h	Busy This bit indicates the status of the read/write operation of indirect access to the queue/channel specific VLAN inclusion register. For write operation write to a register is complete when this bit is reset. For read operation the read data is valid when the bit is reset. The application must make sure that this bit is reset before attempting subsequent access to this register. 0h = Busy status not detected : 0x0 1h = Busy status detected : 0x1
30	RDWR	R/W	0h	Read write control This bit controls the read or write operation for indirectly accessing the queue/channel specific VLAN Inclusion register. When set indicates write operation and when reset indicates read operation. This does not have any effect when CBTI is reset. 0h = Read operation of indirect access : 0x0 1h = Write operation of indirect access : 0x1
29-25	RESERVED	R	0h	Reserved.
24	ADDR	R/W	0h	Address This field selects one of the queue/channel specific VLAN Inclusion register for read/write access. This does not have any effect when CBTI is reset.
23-22	RESERVED	R	0h	Reserved.

Table 43-103. MAC_VLAN_Incl Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	CBTI	R/W	0h	Channel based tag insertion When this bit is set, outer VLAN tag is inserted for every packets transmitted by the MAC. The tag value is taken from the queue/ channel specific VLAN tag register. The VLTI, VLP, VLC, and VLT fields of this register are ignored when this bit is set. When this bit is set, a write operation to byte 3 of this register initiates the read/write access to the indirect register. When reset, outer VLAN operation is based on the setting of VLTI, VLP, VLC and VLT fields of this register. 0h = Channel based tag insertion is disabled : 0x0 1h = Channel based tag insertion is enabled : 0x1
20	VLTI	R/W	0h	VLAN Tag Input When this bit is set, it indicates that the VLAN tag to be inserted or replaced in Tx packet should be taken from: - The Tx descriptor 0h = VLAN Tag Input is disabled : 0x0 1h = VLAN Tag Input is enabled : 0x1
19	CSVL	R/W	0h	C-VLAN or S-VLAN When this bit is set, S-VLAN type (0x88A8) is inserted or replaced in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted or replaced in the 13th and 14th bytes of transmitted packets. 0h = C-VLAN type (0x8100) is inserted or replaced : 0x0 1h = S-VLAN type (0x88A8) is inserted or replaced : 0x1
18	VLP	R/W	0h	VLAN Priority Control When this bit is set, the control bits[17:16] are used for VLAN deletion, insertion, or replacement. When this bit is reset, the mti_vlan_ctrl_i control input is used and bits[17:16] are ignored. 0h = VLAN Priority Control is disabled : 0x0 1h = VLAN Priority Control is enabled : 0x1
17-16	VLC	R/W	0h	VLAN Tag Control in Transmit Packets - 2'b00: No VLAN tag deletion, insertion, or replacement - 2'b01: VLAN tag deletion The MAC removes the VLAN type (bytes 13 and 14) and VLAN tag (bytes 15 and 16) of all transmitted packets with VLAN tags. - 2'b10: VLAN tag insertion The MAC inserts VLT in bytes 15 and 16 of the packet after inserting the Type value (0x8100 or 0x88a8) in bytes 13 and 14. This operation is performed on all transmitted packets, irrespective of whether they already have a VLAN tag. - 2'b11: VLAN tag replacement The MAC replaces VLT in bytes 15 and 16 of all VLAN-type transmitted packets (Bytes 13 and 14 are 0x8100 or 0x88a8). Note: Changes to this field take effect only on the start of a packet. If you write this register field when a packet is being transmitted, only the subsequent packet can use the updated value, that is, the current packet does not use the updated value. 0h = No VLAN tag deletion, insertion, or replacement : 0x0 1h = VLAN tag deletion : 0x1 2h = VLAN tag insertion : 0x2 3h = VLAN tag replacement : 0x3
15-0	VLT	R/W	0h	VLAN Tag for Transmit Packets This field contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

43.7.3.11 MAC_Inner_VLAN_Incl Register (Offset = 64h) [Reset = 0h]

MAC_Inner_VLAN_Incl is shown in Figure 43-50 and described in Table 43-104.

Return to the Summary Table.

The Inner VLAN Tag Inclusion or Replacement register contains the inner VLAN tag to be inserted or replaced in the Transmit packet. It also contains the inner VLAN tag insertion controls.

Figure 43-50. MAC_Inner_VLAN_Incl Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED			VLTI	CSVL	VLP	VLC	
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
VLT							
R/W-0h							
7	6	5	4	3	2	1	0
VLT							
R/W-0h							

Table 43-104. MAC_Inner_VLAN_Incl Register Field Descriptions

Bit	Field	Type	Reset	Description
31-21	RESERVED	R	0h	Reserved.
20	VLTI	R/W	0h	VLAN Tag Input When this bit is set, it indicates that the VLAN tag to be inserted or replaced in Tx packet should be taken from: - The Tx descriptor 0h = VLAN Tag Input is disabled : 0x0 1h = VLAN Tag Input is enabled : 0x1
19	CSVL	R/W	0h	C-VLAN or S-VLAN When this bit is set, S-VLAN type (0x88A8) is inserted or replaced in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted or replaced in the 13th and 14th bytes of transmitted packets. 0h = C-VLAN type (0x8100) is inserted : 0x0 1h = S-VLAN type (0x88A8) is inserted : 0x1
18	VLP	R/W	0h	VLAN Priority Control When this bit is set, the VLC field is used for VLAN deletion, insertion, or replacement. When this bit is reset, the mti_vlan_ctrl_i control input is used and the VLC field is ignored. 0h = VLAN Priority Control is disabled : 0x0 1h = VLAN Priority Control is enabled : 0x1

Table 43-104. MAC_Inner_VLAN_Incl Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17-16	VLC	R/W	0h	VLAN Tag Control in Transmit Packets - 2'b00: No VLAN tag deletion, insertion, or replacement - 2'b01: VLAN tag deletion The MAC removes the VLAN type (bytes 17 and 18) and VLAN tag (bytes 19 and 20) of all transmitted packets with VLAN tags. - 2'b10: VLAN tag insertion The MAC inserts VLT in bytes 19 and 20 of the packet after inserting the Type value (0x8100 or 0x88a8) in bytes 17 and 18. This operation is performed on all transmitted packets, irrespective of whether they already have a VLAN tag. - 2'b11: VLAN tag replacement The MAC replaces VLT in bytes 19 and 20 of all VLAN-type transmitted packets (Bytes 17 and 18 are 0x8100 or 0x88a8). Note: Changes to this field take effect only on the start of a packet. If you write this register field when a packet is being transmitted, only the subsequent packet can use the updated value, that is, the current packet does not use the updated value. 0h = No VLAN tag deletion, insertion, or replacement : 0x0 1h = VLAN tag deletion : 0x1 2h = VLAN tag insertion : 0x2 3h = VLAN tag replacement : 0x3
15-0	VLT	R/W	0h	VLAN Tag for Transmit Packets This field contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: - Bits[15:13]: User Priority - Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) - Bits[11:0]: VLAN Identifier (VID) field of VLAN tag

43.7.3.12 MAC_Q0_Tx_Flow_Ctrl Register (Offset = 70h) [Reset = 0h]

MAC_Q0_Tx_Flow_Ctrl is shown in Figure 43-51 and described in Table 43-105.

Return to the Summary Table.

The Flow Control register controls the generation and reception of the Control (Pause Command) packets by the Flow control module of the MAC. A Write to a register with the Busy bit set to 1 triggers the Flow Control block to generate a Pause packet. The fields of the control packet are selected as specified in the 802.3x specification, and the Pause Time value from this register is used in the Pause Time field of the control packet. The Busy bit remains set until the control packet is transferred onto the cable. The application must make sure that the Busy bit is cleared before writing to the register.
When the PFCE bit in the MAC_Rx_Flow_Ctrl register is enabled, this register controls the generation of Priority Flow Control (PFC) frames with priorities mapped according to PSRQ0 in the MAC_RxQ_Ctrl2 register.

Figure 43-51. MAC_Q0_Tx_Flow_Ctrl Register

31	30	29	28	27	26	25	24
PT							
R/W-0h							
23	22	21	20	19	18	17	16
PT							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
DZPQ	PLT			RESERVED		TFE	FCB_BPA
R/W-0h	R/W-0h			R-0h		R/W-0h	R/W-0h

Table 43-105. MAC_Q0_Tx_Flow_Ctrl Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	PT	R/W	0h	Pause Time This field holds the value to be used in the Pause Time field in the Tx control packet. If the Pause Time bits are configured to be double-synchronized to the (G)MII clock domain, consecutive writes to this register should be performed only after at least four clock cycles in the destination clock domain.
15-8	RESERVED	R	0h	Reserved.
7	DZPQ	R/W	0h	Disable Zero-Quanta Pause When this bit is set, it disables the automatic generation of the zero-quanta Pause packets on de-assertion of the flow-control signal from the FIFO layer (MTL or external sideband flow control signal sbd_flowctrl_i or mti_flowctrl_i). When this bit is reset, normal operation with automatic zero-quanta Pause packet generation is enabled. 0h = Zero-Quanta Pause packet generation is enabled : 0x0 1h = Zero-Quanta Pause packet generation is disabled : 0x1

Table 43-105. MAC_Q0_Tx_Flow_Ctrl Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-4	PLT	R/W	0h	Pause Low Threshold This field configures the threshold of the Pause timer at which the input flow control signal mti_flowctrl_i (or sbd_flowctrl_i) is checked for automatic retransmission of the Pause packet. The threshold values should be always less than the Pause Time configured in Bits[31:16]. For example, if PT = 100H (256 slot times), and PLT = 001, a second Pause packet is automatically transmitted if the mti_flowctrl_i signal is asserted at 228 (256-28) slot times after the first Pause packet is transmitted. The following list provides the threshold values for different values. The slot time is defined as the time taken to transmit 512 bits (64 bytes) on the GMII or MII interface. This (approximate) computation is based on the packet size (64, 1518, 2000, 9018, 16384, or 32768) + 2 Pause Packet Size + IPG in Slot Times. 0h = Pause Time minus 4 Slot Times (PT -4 slot times) : 0x0 1h = Pause Time minus 28 Slot Times (PT -28 slot times) : 0x1 2h = Pause Time minus 36 Slot Times (PT -36 slot times) : 0x2 3h = Pause Time minus 144 Slot Times (PT -144 slot times) : 0x3 4h = Pause Time minus 256 Slot Times (PT -256 slot times) : 0x4 5h = Pause Time minus 512 Slot Times (PT -512 slot times) : 0x5 6h = Reserved : 0x6
3-2	RESERVED	R	0h	Reserved.
1	TFE	R/W	0h	Transmit Flow Control Enable Full-Duplex Mode: In the full-duplex mode, when this bit is set, the MAC enables the flow control operation to Tx Pause packets. When this bit is reset, the flow control operation in the MAC is disabled, and the MAC does not transmit any Pause packets. Half-Duplex Mode: In the half-duplex mode, when this bit is set, the MAC enables the backpressure operation. When this bit is reset, the backpressure feature is disabled. 0h = Transmit Flow Control is disabled : 0x0 1h = Transmit Flow Control is enabled : 0x1
0	FCB_BPA	R/W	0h	Flow Control Busy or Backpressure Activate This bit initiates a Pause packet in the full-duplex mode and activates the backpressure function in the half-duplex mode if the TFE bit is set. Full-Duplex Mode: In the full-duplex mode, this bit should be read as 1'b0 before writing to this register. To initiate a Pause packet, the application must set this bit to 1'b1. During Control packet transfer, this bit continues to be set to indicate that a packet transmission is in progress. When Pause packet transmission is complete, the MAC resets this bit to 1'b0. You should not write to this register until this bit is cleared. Half-Duplex Mode: When this bit is set (and TFE bit is set) in the half-duplex mode, the MAC asserts the backpressure. During backpressure, when the MAC receives a new packet, the transmitter starts sending a JAM pattern resulting in a collision. This control register bit is logically ORed with the mti_flowctrl_i input signal for the backpressure function. When the MAC is configured for the full-duplex mode, the BPA is automatically disabled. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0h = Flow Control Busy or Backpressure Activate is disabled : 0x0 1h = Flow Control Busy or Backpressure Activate is enabled : 0x1

43.7.3.13 MAC_Rx_Flow_Ctrl Register (Offset = 90h) [Reset = 0h]

MAC_Rx_Flow_Ctrl is shown in Figure 43-52 and described in Table 43-106.

Return to the Summary Table.

The Receive Flow Control register controls the pausing of MAC Transmit based on the received Pause packet.

Figure 43-52. MAC_Rx_Flow_Ctrl Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							RESERVED
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED						UP	RFE
R-0h						R/W-0h	R/W-0h

Table 43-106. MAC_Rx_Flow_Ctrl Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved.
8	RESERVED	R	0h	Reserved.
7-2	RESERVED	R	0h	Reserved.
1	UP	R/W	0h	Unicast Pause Packet Detect A pause packet is processed when it has the unique multicast address specified in the IEEE 802.3. When this bit is set, the MAC can also detect Pause packets with unicast address of the station. This unicast address should be as specified in MAC_Address0_High and MAC_Address0_Low. When this bit is reset, the MAC only detects Pause packets with unique multicast address. Note: The MAC does not process a Pause packet if the multicast address is different from the unique multicast address. This is also applicable to the received PFC packet when the Priority Flow Control (PFC) is enabled. The unique multicast address (0x01_80_C2_00_00_01) is as specified in IEEE 802.1 Qbb-2011. 0h = Unicast Pause Packet Detect disabled : 0x0 1h = Unicast Pause Packet Detect enabled : 0x1
0	RFE	R/W	0h	Receive Flow Control Enable When this bit is set and the MAC is operating in full-duplex mode, the MAC decodes the received Pause packet and disables its transmitter for a specified (Pause) time. When this bit is reset or the MAC is operating in half-duplex mode, the decode function of the Pause packet is disabled. When PFC is enabled, flow control is enabled for PFC packets. The MAC decodes the received PFC packet and disables the Transmit queue, with matching priorities, for a duration of received Pause time. 0h = Receive Flow Control is disabled : 0x0 1h = Receive Flow Control is enabled : 0x1

43.7.3.14 MAC_RxQ_Ctrl4 Register (Offset = 94h) [Reset = 0h]

MAC_RxQ_Ctrl4 is shown in Figure 43-53 and described in Table 43-107.

Return to the Summary Table.

The Receive Queue Control 4 register controls the routing of unicast and multicast packets that fail the Destination or Source address filter to the Rx queues.

Figure 43-53. MAC_RxQ_Ctrl4 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						VFFQ	VFFQE
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED						MFFQ	MFFQE
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED						UFFQ	UFFQE
R-0h						R/W-0h	R/W-0h

Table 43-107. MAC_RxQ_Ctrl4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved.
17	VFFQ	R/W	0h	VLAN Tag Filter Fail Packets Queue This field holds the Rx queue number to which the tagged packets failing the Destination or Source Address filter (and UFFQE/MFFQE not enabled) or failing the VLAN tag filter must be routed to. This field is valid only when the VFFQE bit is set.
16	VFFQE	R/W	0h	VLAN Tag Filter Fail Packets Queuing Enable When this bit is set, the tagged packets which fail the Destination or Source address filter or fail the VLAN tag filter, are routed to the Rx Queue Number programmed in the VFFQ. When this bit is reset, the tagged packets which fail the Destination or Source address filter or fail the VLAN tag filter are routed based on other routing options. This bit is valid only when the RA bit of the MAC_Packet_Filter register is set. 0h = VLAN tag Filter Fail Packets Queuing is disabled : 0x0 1h = VLAN tag Filter Fail Packets Queuing is enabled : 0x1
15-10	RESERVED	R	0h	Reserved.
9	MFFQ	R/W	0h	Multicast Address Filter Fail Packets Queue. This field holds the Rx queue number to which the Multicast packets failing the Destination or Source Address filter are routed to. This field is valid only when the MFFQE bit is set.
8	MFFQE	R/W	0h	Multicast Address Filter Fail Packets Queuing Enable. When this bit is set, the Multicast packets which fail the Destination or Source address filter is routed to the Rx Queue Number programmed in the MFFQ. When this bit is reset, the Multicast packets which fail the Destination or Source address filter is routed based on other routing options. This bit is valid only when the RA bit of the MAC_Packet_Filter register is set. 0h = Multicast Address Filter Fail Packets Queuing is disabled : 0x0 1h = Multicast Address Filter Fail Packets Queuing is enabled : 0x1
7-2	RESERVED	R	0h	Reserved.

Table 43-107. MAC_RxQ_Ctrl4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	UFFQ	R/W	0h	Unicast Address Filter Fail Packets Queue. This field holds the Rx queue number to which the Unicast packets failing the Destination or Source Address filter are routed to. This field is valid only when the UFFQE bit is set.
0	UFFQE	R/W	0h	Unicast Address Filter Fail Packets Queuing Enable. When this bit is set, the Unicast packets which fail the Destination or Source address filter is routed to the Rx Queue Number programmed in the UFFQ. When this bit is reset, the Unicast packets which fail the Destination or Source address filter is routed based on other routing options. This bit is valid only when the RA bit of the MAC_Packet_Filter register is set. 0h = Unicast Address Filter Fail Packets Queuing is disabled : 0x0 1h = Unicast Address Filter Fail Packets Queuing is enabled : 0x1

43.7.3.15 MAC_RxQ_Ctrl0 Register (Offset = A0h) [Reset = 0h]

MAC_RxQ_Ctrl0 is shown in Figure 43-54 and described in Table 43-108.

Return to the Summary Table.

The Receive Queue Control 0 register controls the queue management in the MAC Receiver.
Note: In multiple Rx queues configuration, all the queues are disabled by default. Enable the Rx queue by programming the corresponding field in this register.

Figure 43-54. MAC_RxQ_Ctrl0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		RESERVED		RESERVED		RESERVED	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
RESERVED		RESERVED		RXQ1EN		RXQ0EN	
R-0h		R-0h		R/W-0h		R/W-0h	

Table 43-108. MAC_RxQ_Ctrl0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved.
15-14	RESERVED	R	0h	Reserved.
13-12	RESERVED	R	0h	Reserved.
11-10	RESERVED	R	0h	Reserved.
9-8	RESERVED	R	0h	Reserved.
7-6	RESERVED	R	0h	Reserved.
5-4	RESERVED	R	0h	Reserved.
3-2	RXQ1EN	R/W	0h	Receive Queue 1 Enable This field is similar to the RXQ0EN field. 0h = Queue not enabled : 0x0 1h = Queue enabled for AV : 0x1 2h = Queue enabled for DCB/Generic : 0x2 3h = Reserved : 0x3
1-0	RXQ0EN	R/W	0h	Receive Queue 0 Enable This field indicates whether Rx Queue 0 is enabled for AV or DCB. 0h = Queue not enabled : 0x0 1h = Queue enabled for AV : 0x1 2h = Queue enabled for DCB/Generic : 0x2 3h = Reserved : 0x3

43.7.3.16 MAC_RxQ_Ctrl1 Register (Offset = A4h) [Reset = 0h]

MAC_RxQ_Ctrl1 is shown in Figure 43-55 and described in Table 43-109.

Return to the Summary Table.

The Receive Queue Control 1 register controls the routing of multicast, broadcast, AV, DCB, and untagged packets to the Rx queues.

Figure 43-55. MAC_RxQ_Ctrl1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED	TPQC	RESERVED	MCBCQEN	RESERVED	MCBCQ		
R-0h	R/W-0h	R-0h	R/W-0h	R-0h	R/W-0h		
15	14	13	12	11	10	9	8
RESERVED	UPQ			RESERVED	RESERVED		
R-0h	R/W-0h			R-0h	R-0h		
7	6	5	4	3	2	1	0
RESERVED	PTPQ			RESERVED	RESERVED		
R-0h	R/W-0h			R-0h	R-0h		

Table 43-109. MAC_RxQ_Ctrl1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved.
23	RESERVED	R	0h	Reserved.
22	TPQC	R/W	0h	Tagged PTP over Ethernet Packets Queuing Control. This field controls the routing of the VLAN Tagged PTPoE packets. If DWC_EQOS_AV_ENABLE is selected in the configuration, the following programmable options are allowed. - 2'b00: VLAN Tagged PTPoE packets are routed as generic VLAN Tagged packet (based on PSRQ for only non-AV enabled Rx Queues). - 2'b01: VLAN Tagged PTPoE packets are routed to Rx Queue specified by PTPQ field (That Rx Queue can be enabled for AV or non-AV traffic). - 2'b10: VLAN Tagged PTPoE packets are routed to only AV enabled Rx Queues based on PSRQ. - 2'b11: Reserved If DWC_EQOS_AV_ENABLE is not selected in the configuration, the following programmable options are allowed. - 1'b0: VLAN Tagged PTPoE packets are routed as generic VLAN Tagged packet (based on PSRQ for DCB/Generic enabled Rx Queues). - 1'b1: VLAN Tagged PTPoE packets are routed to Rx Queues specified by PTPQ field.
21	RESERVED	R	0h	Reserved.
20	MCBCQEN	R/W	0h	Multicast and Broadcast Queue Enable This bit specifies that Multicast or Broadcast packets routing to the Rx Queue is enabled and the Multicast or Broadcast packets must be routed to Rx Queue specified in MCBCQ field. 0h = Multicast and Broadcast Queue is disabled : 0x0 1h = Multicast and Broadcast Queue is enabled : 0x1
19	RESERVED	R	0h	Reserved.

Table 43-109. MAC_RxQ_Ctrl1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-16	MCBCQ	R/W	0h	Multicast and Broadcast Queue This field specifies the Rx Queue onto which Multicast or Broadcast Packets are routed. Any Rx Queue enabled for Generic/DCB/AV traffic can be used to route the Multicast or Broadcast Packets. 0h = Receive Queue 0 : 0x0 1h = Receive Queue 1 : 0x1 2h = Receive Queue 2 : 0x2 3h = Receive Queue 3 : 0x3 4h = Receive Queue 4 : 0x4 5h = Receive Queue 5 : 0x5 6h = Receive Queue 6 : 0x6 7h = Receive Queue 7 : 0x7
15	RESERVED	R	0h	Reserved.
14-12	UPQ	R/W	0h	Untagged Packet Queue This field indicates the Rx Queue to which Untagged Packets are to be routed. Any Rx Queue enabled for Generic/DCB/AV traffic can be used to route the Untagged Packets. 0h = Receive Queue 0 : 0x0 1h = Receive Queue 1 : 0x1 2h = Receive Queue 2 : 0x2 3h = Receive Queue 3 : 0x3 4h = Receive Queue 4 : 0x4 5h = Receive Queue 5 : 0x5 6h = Receive Queue 6 : 0x6 7h = Receive Queue 7 : 0x7
11	RESERVED	R	0h	Reserved.
10-8	RESERVED	R	0h	Reserved.
7	RESERVED	R	0h	Reserved.
6-4	PTPQ	R/W	0h	PTP Packets Queue This field specifies the Rx queue on which the PTP packets sent over the Ethernet payload (not over IPv4 or IPv6) are routed. When the AV8021ASMEN bit of MAC_Timestamp_Control register is set, only untagged PTP over Ethernet packets are routed on an Rx Queue. If the bit is not set, then based on programming of TPQC field, both tagged and untagged PTPoE packets can be routed to this Rx Queue. 0h = Receive Queue 0 : 0x0 1h = Receive Queue 1 : 0x1 2h = Receive Queue 2 : 0x2 3h = Receive Queue 3 : 0x3 4h = Receive Queue 4 : 0x4 5h = Receive Queue 5 : 0x5 6h = Receive Queue 6 : 0x6 7h = Receive Queue 7 : 0x7
3	RESERVED	R	0h	Reserved.
2-0	RESERVED	R	0h	Reserved.

43.7.3.17 MAC_RxQ_Ctrl2 Register (Offset = A8h) [Reset = 0h]

MAC_RxQ_Ctrl2 is shown in Figure 43-56 and described in Table 43-110.

Return to the Summary Table.

This register controls the routing of tagged packets based on the USP (user Priority) field of the received packets to the RxQueues 0 to 3.

Figure 43-56. MAC_RxQ_Ctrl2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RESERVED								PSRQ1								PSRQ0							
R-0h								R-0h								R/W-0h								R/W-0h							

Table 43-110. MAC_RxQ_Ctrl2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved.
23-16	RESERVED	R	0h	Reserved.
15-8	PSRQ1	R/W	0h	Priorities Selected in the Receive Queue 1 This field decides the priorities assigned to Rx Queue 1. All packets with priorities that match the values set in this field are routed to Rx Queue 1. For example, if PSRQ1[4] is set, packets with USP field equal to 4 are routed to Rx Queue 1. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues.
7-0	PSRQ0	R/W	0h	Priorities Selected in the Receive Queue 0 This field decides the priorities assigned to Rx Queue 0. All packets with priorities that match the values set in this field are routed to Rx Queue 0. For example, if PSRQ0[5] is set, packets with USP field equal to 5 are routed to Rx Queue 0. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues.

43.7.3.18 MAC_Interrupt_Status Register (Offset = B0h) [Reset = 0h]

MAC_Interrupt_Status is shown in Figure 43-57 and described in Table 43-111.

Return to the Summary Table.

The Interrupt Status register contains the status of interrupts.

Figure 43-57. MAC_Interrupt_Status Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED			RESERVED	RESERVED	MDIOIS	RESERVED	RESERVED
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED	RXSTSIS	TXSTSIS	TSIS	MMCRXIPIS	MMCTXIS	MMCRXIS	MMCIS
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED		LPIIS	PMTIS	PHYIS	RESERVED	RESERVED	RESERVED
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 43-111. MAC_Interrupt_Status Register Field Descriptions

Bit	Field	Type	Reset	Description
31-21	RESERVED	R	0h	Reserved.
20	RESERVED	R	0h	Reserved.
19	RESERVED	R	0h	Reserved.
18	MDIOIS	R	0h	MDIO Interrupt Status This bit indicates an interrupt event after the completion of MDIO operation. To reset this bit, the application has to read this bit/Write 1 to this bit when RCWE bit of MAC_CSR_SW_Ctrl register is set. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = MDIO Interrupt status not active : 0x0 1h = MDIO Interrupt status active : 0x1
17	RESERVED	R	0h	Reserved.
16	RESERVED	R	0h	Reserved.
15	RESERVED	R	0h	Reserved.
14	RXSTSIS	R	0h	Receive Status Interrupt This bit indicates the status of received packets. This bit is set when the RWT bit is set in the MAC_Rx_Tx_Status register. This bit is cleared when the corresponding interrupt source bit is read (or corresponding interrupt source bit is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set) in the MAC_Rx_Tx_Status register. 0h = Receive Interrupt status not active : 0x0 1h = Receive Interrupt status active : 0x1

Table 43-111. MAC_Interrupt_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	TXSTSIS	R	0h	Transmit Status Interrupt This bit indicates the status of transmitted packets. This bit is set when any of the following bits is set in the MAC_Rx_Tx_Status register: - Excessive Collision (EXCOL) - Late Collision (LCOL) - Excessive Deferral (EXDEF) - Loss of Carrier (LCARR) - No Carrier (NCARR) - Jabber Timeout (TJT) This bit is cleared when the corresponding interrupt source bit is read (or corresponding interrupt source bit is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set) in the MAC_Rx_Tx_Status register. 0h = Transmit Interrupt status not active : 0x0 1h = Transmit Interrupt status active : 0x1
12	TSIS	R	0h	Timestamp Interrupt Status If the Timestamp feature is enabled, this bit is set when any of the following conditions is true: - The system time value is equal to or exceeds the value specified in the Target Time High and Low registers. - There is an overflow in the Seconds register. - The Target Time Error occurred, that is, programmed target time already elapsed. If the Auxiliary Snapshot feature is enabled, this bit is set when the auxiliary snapshot trigger is asserted. In configurations other than EQOS_CORE, when drop transmit status is enabled in MTL, this bit is set when the captured transmit timestamp is updated in the MAC_Tx_Timestamp_Status_Nanoseconds and Mac_TxTimestamp_Status_Seconds registers. When PTP offload feature is enabled, this bit is set when the captured transmit timestamp is updated in the MAC_Tx_Timestamp_Status_Nanoseconds and MAC_Tx_Timestamp_Status_Seconds registers, for PTO generated Delay Request and Pdelay request packets. This bit is cleared when the corresponding interrupt source bit is read (or corresponding interrupt source bit is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set) in the MAC_Timestamp_Status register. 0h = Timestamp Interrupt status not active : 0x0 1h = Timestamp Interrupt status active : 0x1
11	MMCRXIPIS	R	0h	MMC Receive Checksum Offload Interrupt Status This bit is set high when an interrupt is generated in the MMC Receive Checksum Offload Interrupt Register. This bit is cleared when all bits in this interrupt register are cleared. This bit is valid only when you select the Enable MAC Management Counters (MMC) and Enable Receive TCP/IP Checksum Check options. 0h = MMC Receive Checksum Offload Interrupt status not active : 0x0 1h = MMC Receive Checksum Offload Interrupt status active : 0x1
10	MMCTXIS	R	0h	MMC Transmit Interrupt Status This bit is set high when an interrupt is generated in the MMC Transmit Interrupt Register. This bit is cleared when all bits in this interrupt register are cleared. This bit is valid only when you select the Enable MAC Management Counters (MMC) option. 0h = MMC Transmit Interrupt status not active : 0x0 1h = MMC Transmit Interrupt status active : 0x1

Table 43-111. MAC_Interrupt_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	MMCRXIS	R	0h	MMC Receive Interrupt Status This bit is set high when an interrupt is generated in the MMC Receive Interrupt Register. This bit is cleared when all bits in this interrupt register are cleared. This bit is valid only when you select the Enable MAC Management Counters (MMC) option. 0h = MMC Receive Interrupt status not active : 0x0 1h = MMC Receive Interrupt status active : 0x1
8	MMCIS	R	0h	MMC Interrupt Status This bit is set high when Bit 11, Bit 10, or Bit 9 is set high. This bit is cleared only when all these bits are low. This bit is valid only when you select the Enable MAC Management Counters (MMC) option. 0h = MMC Interrupt status not active : 0x0 1h = MMC Interrupt status active : 0x1
7-6	RESERVED	R	0h	Reserved.
5	LPIIS	R	0h	LPI Interrupt Status When the Energy Efficient Ethernet feature is enabled, this bit is set for any LPI state entry or exit in the MAC Transmitter or Receiver. This bit is cleared when the corresponding interrupt source bit of MAC_LPI_Control_Status register is read (or corresponding interrupt source bit of MAC_LPI_Control_Status register is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set). 0h = LPI Interrupt status not active : 0x0 1h = LPI Interrupt status active : 0x1
4	PMTIS	R	0h	PMT Interrupt Status This bit is set when a Magic packet or Wake-on-LAN packet is received in the power-down mode (RWKPRCVD and MGKPRCVD bits in MAC_PMT_Control_Status register). This bit is cleared when corresponding interrupt source bit are cleared because of a Read operation to the MAC_PMT_Control_Status register (or corresponding interrupt source bit of MAC_PMT_Control_Status register is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set). This bit is valid only when you select the Enable Power Management option. 0h = PMT Interrupt status not active : 0x0 1h = PMT Interrupt status active : 0x1
3	PHYIS	R	0h	PHY Interrupt This bit is set when rising edge is detected on the phy_intr_i input. This bit is cleared when this register is read (or this bit is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set). 0h = PHY Interrupt not detected : 0x0 1h = PHY Interrupt detected : 0x1
2	RESERVED	R	0h	Reserved.
1	RESERVED	R	0h	Reserved.
0	RESERVED	R	0h	Reserved.

43.7.3.19 MAC_Interrupt_Enable Register (Offset = B4h) [Reset = 0h]

MAC_Interrupt_Enable is shown in Figure 43-58 and described in Table 43-112.

Return to the Summary Table.

The Interrupt Enable register contains the masks for generating the interrupts.

Figure 43-58. MAC_Interrupt_Enable Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED					MDIOIE	RESERVED	RESERVED
R-0h					R/W-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED	RXSTSIE	TXSTSIE	TSIE	RESERVED			
R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h			
7	6	5	4	3	2	1	0
RESERVED		LPIIE	PMTIE	PHYIE	RESERVED	RESERVED	RESERVED
R-0h		R/W-0h	R/W-0h	R/W-0h	R-0h	R-0h	R-0h

Table 43-112. MAC_Interrupt_Enable Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved.
18	MDIOIE	R/W	0h	MDIO Interrupt Enable When this bit is set, it enables the assertion of the interrupt when MDIOIS field is set in the MAC_Interrupt_Status register. 0h = MDIO Interrupt is disabled : 0x0 1h = MDIO Interrupt is enabled : 0x1
17	RESERVED	R	0h	Reserved.
16	RESERVED	R	0h	Reserved.
15	RESERVED	R	0h	Reserved.
14	RXSTSIE	R/W	0h	Receive Status Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of RXSTSIS bit in the MAC_Interrupt_Status register. 0h = Receive Status Interrupt is disabled : 0x0 1h = Receive Status Interrupt is enabled : 0x1
13	TXSTSIE	R/W	0h	Transmit Status Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of TXSTSIS bit in the MAC_Interrupt_Status register. 0h = Timestamp Status Interrupt is disabled : 0x0 1h = Timestamp Status Interrupt is enabled : 0x1
12	TSIE	R/W	0h	Timestamp Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of TSIS bit in MAC_Interrupt_Status register. 0h = Timestamp Interrupt is disabled : 0x0 1h = Timestamp Interrupt is enabled : 0x1
11-6	RESERVED	R	0h	Reserved.
5	LPIIE	R/W	0h	LPI Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of LPIIS bit in MAC_Interrupt_Status register. 0h = LPI Interrupt is disabled : 0x0 1h = LPI Interrupt is enabled : 0x1

Table 43-112. MAC_Interrupt_Enable Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	PMTIE	R/W	0h	PMT Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of PMTIS bit in MAC_Interrupt_Status register. 0h = PMT Interrupt is disabled : 0x0 1h = PMT Interrupt is enabled : 0x1
3	PHYIE	R/W	0h	PHY Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of PHYIS bit in MAC_Interrupt_Status register. 0h = PHY Interrupt is disabled : 0x0 1h = PHY Interrupt is enabled : 0x1
2	RESERVED	R	0h	Reserved.
1	RESERVED	R	0h	Reserved.
0	RESERVED	R	0h	Reserved.

43.7.3.20 MAC_Rx_Tx_Status Register (Offset = B8h) [Reset = 0h]

MAC_Rx_Tx_Status is shown in Figure 43-59 and described in Table 43-113.

Return to the Summary Table.

The Receive Transmit Status register contains the Receive and Transmit Error status.

Figure 43-59. MAC_Rx_Tx_Status Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							RWT
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED		EXCOL	LCOL	EXDEF	LCARR	NCARR	TJT
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 43-113. MAC_Rx_Tx_Status Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved.
8	RWT	R	0h	Receive Watchdog Timeout This bit is set when a packet with length greater than 2,048 bytes is received (10, 240 bytes when Jumbo Packet mode is enabled) and the WD bit is reset in the MAC_Configuration register. This bit is set when a packet with length greater than 16,383 bytes is received and the WD bit is set in the MAC_Configuration register. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = No receive watchdog timeout : 0x0 1h = Receive watchdog timed out : 0x1
7-6	RESERVED	R	0h	Reserved.
5	EXCOL	R	0h	Excessive Collisions When the DTXSTS bit is set in the MTL_Operation_Mode register, this bit indicates that the transmission aborted after 16 successive collisions while attempting to transmit the current packet. If the DR bit is set in the MAC_Configuration register, this bit is set after the first collision and the packet transmission is aborted. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = No collision : 0x0 1h = Excessive collision is sensed : 0x1

Table 43-113. MAC_Rx_Tx_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	LCOL	R	0h	Late Collision When the DTXSTS bit is set in the MTL_Operation_Mode register, this bit indicates that the packet transmission aborted because a collision occurred after the collision window (64 bytes including Preamble in MII mode 512 bytes including Preamble and Carrier Extension in GMII mode). This bit is not valid if the Underflow error occurs. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = No collision : 0x0 1h = Late collision is sensed : 0x1
3	EXDEF	R	0h	Excessive Deferral When the DTXSTS bit is set in the MTL_Operation_Mode register and the DC bit is set in the MAC_Configuration register, this bit indicates that the transmission ended because of excessive deferral of over 24,288 bit times (155,680 in 1000/2500 Mbps mode or when Jumbo packet is enabled). Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = No Excessive deferral : 0x0 1h = Excessive deferral : 0x1
2	LCARR	R	0h	Loss of Carrier When the DTXSTS bit is set in the MTL_Operation_Mode register, this bit indicates that the loss of carrier occurred during packet transmission, that is, the phy_crs_i signal was inactive for one or more transmission clock periods during packet transmission. This bit is valid only for packets transmitted without collision. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = Carrier is present : 0x0 1h = Loss of carrier : 0x1
1	NCARR	R	0h	No Carrier When the DTXSTS bit is set in the MTL_Operation_Mode register, this bit indicates that the carrier signal from the PHY is not present at the end of preamble transmission. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = Carrier is present : 0x0 1h = No carrier : 0x1
0	TJT	R	0h	Transmit Jabber Timeout This bit indicates that the Transmit Jabber Timer expired which happens when the packet size exceeds 2,048 bytes (10,240 bytes when the Jumbo packet is enabled) and JD bit is reset in the MAC_Configuration register. This bit is set when the packet size exceeds 16,383 bytes and the JD bit is set in the MAC_Configuration register. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = No Transmit Jabber Timeout : 0x0 1h = Transmit Jabber Timeout occured : 0x1

43.7.3.21 MAC_PMT_Control_Status Register (Offset = C0h) [Reset = 0h]

MAC_PMT_Control_Status is shown in Figure 43-60 and described in Table 43-114.

Return to the Summary Table.

The PMT Control and Status Register.

Figure 43-60. MAC_PMT_Control_Status Register

31	30	29	28	27	26	25	24
RWKFILTRST	RESERVED				RWKPTR		
R/W-0h	R-0h				R-0h		
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					RWKPFE	GLBLUCAST	RESERVED
R-0h					R/W-0h	R/W-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	RWKPRCVD	MGKPRCVD	RESERVED		RWKPKTEN	MGKPKTEN	PWRDWN
R-0h	R-0h	R-0h	R-0h		R/W-0h	R/W-0h	R/W-0h

Table 43-114. MAC_PMT_Control_Status Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RWKFILTRST	R/W	0h	Remote Wake-Up Packet Filter Register Pointer Reset When this bit is set, the remote wake-up packet filter register pointer is reset to 3'b000. It is automatically cleared after 1 clock cycle. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0h = Remote Wake-Up Packet Filter Register Pointer is not Reset : 0x0 1h = Remote Wake-Up Packet Filter Register Pointer is Reset : 0x1
30-29	RESERVED	R	0h	Reserved.
28-24	RWKPTR	R	0h	Remote Wake-up FIFO Pointer This field gives the current value (0 to 7, 15, or 31 when 4, 8, or 16 Remote Wake-up Packet Filters are selected) of the Remote Wake-up Packet Filter register pointer. When the value of this pointer is equal to maximum for the selected number of Remote Wake-up Packet Filters, the contents of the Remote Wake-up Packet Filter Register are transferred to the clk_rx_i domain when a Write occurs to that register.
23-11	RESERVED	R	0h	Reserved.

Table 43-114. MAC_PMT_Control_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	RWKPFE	R/W	0h	Remote Wake-up Packet Forwarding Enable When this bit is set along with RWKPKTEN, the MAC receiver drops all received frames until it receives the expected Wake-up frame. All frames after that event including the received wake-up frame are forwarded to application. This bit is then self-cleared on receiving the wake-up packet. The application can also clear this bit before the expected wake-up frame is received. In such cases, the MAC reverts to the default behavior where packets received are forwarded to the application. This bit must only be set when RWKPKTEN is set high and PWRDWN is set low. The setting of this bit has no effect when PWRDWN is set high. Note: If Magic Packet Enable and Wake-Up Frame Enable are both set along with setting of this bit and Magic Packet is received prior to wake-up frame, this bit is self-cleared on receiving Magic Packet, the received Magic packet is dropped, and all frames after received Magic Packet are forwarded to application. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0h = Remote Wake-up Packet Forwarding is disabled : 0x0 1h = Remote Wake-up Packet Forwarding is enabled : 0x1
9	GLBLUCAST	R/W	0h	Global Unicast When this bit set, any unicast packet filtered by the MAC (DAF) address recognition is detected as a remote wake-up packet. 0h = Global unicast is disabled : 0x0 1h = Global unicast is enabled : 0x1
8-7	RESERVED	R	0h	Reserved.
6	RWKPRCVD	R	0h	Remote Wake-Up Packet Received When this bit is set, it indicates that the power management event is generated because of the reception of a remote wake-up packet. This bit is cleared when this register is read. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = Remote wake-up packet is received : 0x0 1h = Remote wake-up packet is received : 0x1
5	MGKPRCVD	R	0h	Magic Packet Received When this bit is set, it indicates that the power management event is generated because of the reception of a magic packet. This bit is cleared when this register is read. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = No Magic packet is received : 0x0 1h = Magic packet is received : 0x1
4-3	RESERVED	R	0h	Reserved.
2	RWKPKTEN	R/W	0h	Remote Wake-Up Packet Enable When this bit is set, a power management event is generated when the MAC receives a remote wake-up packet. 0h = Remote wake-up packet is disabled : 0x0 1h = Remote wake-up packet is enabled : 0x1
1	MGKPKTEN	R/W	0h	Magic Packet Enable When this bit is set, a power management event is generated when the MAC receives a magic packet. 0h = Magic Packet is disabled : 0x0 1h = Magic Packet is enabled : 0x1

Table 43-114. MAC_PMT_Control_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	PWRDWN	R/W	0h	Power Down When this bit is set, the MAC receiver drops all received packets until it receives the expected magic packet or remote wake-up packet. This bit is then self-cleared and the power-down mode is disabled. The software can clear this bit before the expected magic packet or remote wake-up packet is received. The packets received by the MAC after this bit is cleared are forwarded to the application. This bit must only be set when the Magic Packet Enable, Global Unicast, or Remote Wake-Up Packet Enable bit is set high. Note: You can gate-off the CSR clock during the power-down mode. However, when the CSR clock is gated-off, you cannot perform any read or write operations on this register. Therefore, the Software cannot clear this bit. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0h = Power down is disabled : 0x0 1h = Power down is enabled : 0x1

43.7.3.22 MAC_RWK_Packet_Filter Register (Offset = C4h) [Reset = 0h]

MAC_RWK_Packet_Filter is shown in Figure 43-61 and described in Table 43-115.

Return to the Summary Table.

The wkuppktfilter_reg register at address 0C4H loads the Wake-up Packet Filter register.
To load values in a Wake-up Packet Filter register, the entire register (wkuppktfilter_reg) must be written.
The wkuppktfilter_reg register is loaded by sequentially loading the eight, sixteen or thirty two register values in address (0C4H) for wkuppktfilter_reg0, wkuppktfilter_reg1,.. wkuppktfilter_reg31, respectively. The wkuppktfilter_reg register is read in a similar way. The DWC_ether_qos updates the wkuppktfilter_reg register current pointer value in Bits[26:24] of MAC_PMT_Control_Status register.
Filter i Byte Mask: The filter i byte mask register defines the bytes of the packet that are examined by filter i (0, 1, 2, 3,..,15) to determine whether or not a packet is a wake-up packet.
- The MSB (31st bit) must be zero.
- Bit j[30:0] is the byte mask.
- If Bit j (byte number) of the byte mask is set, the CRC block processes the Filter i Offset + j of the incoming packet
otherwise Filter i Offset + j is ignored.
Filter i Command: The 4-bit filter i command controls the filter i operation.
- Bit 3 specifies the address type, defining the destination address type of the pattern. When the bit is set, the pattern applies to only multicast packets
when the bit is reset, the pattern applies only to unicast packet.
- Bit 2 (Inverse Mode), when set, reverses the logic of the CRC16 hash function signal, to reject a packet with matching CRC_16 value.
- Bit 2, along with Bit 1, allows a MAC to reject a subset of remote wake-up packets by creating filter logic such as "Pattern 1 AND NOT Pattern 2".
- Bit 1 (And_Previous) implements the Boolean logic. When set, the result of the current entry is logically ANDed with the result of the previous filter. This AND logic allows a filter pattern longer than 32 bytes by splitting the mask among two, three, or four filters. This depends on the number of filters that have the And_Previous bit set.
- Bit 0 is the enable for filter i. If Bit 0 is not set, filter i is disabled.
Filter i Offset: This filter i offset register defines the offset (within the packet) from which the filter i examines the packets.
- This 8-bit pattern-offset is the offset for the filter i first byte to be examined.
- The minimum allowed offset is 12, which refers to the 13th byte of the packet.
- The offset value 0 refers to the first byte of the packet.
Filter i CRC-16: This filter i CRC-16 register contains the CRC_16 value calculated from the pattern and also the byte mask programmed to the wake-up filter register block.
- The 16-bit CRC calculation uses the following polynomial:
G(x) = x^16 + x^15 + x^2 + 1
Each mask, used in the hash function calculation, is compared with a 16-bit value associated with that mask.
Each filter has the following:
- 32-bit Mask: Each bit in this mask corresponds to one byte in the detected packet. If the bit is 1', the corresponding byte is taken into the CRC16 calculation.
- 8-bit Offset Pointer: Specifies the byte to start the CRC16 computation.
The pointer and the mask are used together to locate the bytes to be used in the CRC16 calculations.
- Note: If you are accessing these registers in byte or half-word mode, the internal counter to access the appropriate wkuppktfilter_reg is incremented when CPU accesses Lane 3 (or Lane 0 in big-endian mode).
- Note: When any Register content is being transferred to a different clock domain after a write operation, there should not be any further writes to the same location until the first write is updated. Otherwise, the second write operation does not get updated to the destination clock domain. Therefore, the delay between two writes to the same register location should be at least 4 cycles of the destination clock (PHY receive clock, PHY transmit clock, or PTP clock).
Notes on And_Previous bit setting
The And_Previous bit setting is applicable within a set of 4 filters.

- Setting of And_Previous bit of filter that is not enabled has no effect. In other words, setting And_Previous bit of lowest number filter in the set of 4 filters has no effect. For example, setting of And_Previous bit of Filter 0 has no effect.
- If And_Previous bit is set for filter to form AND chained filter, the AND chain breaks at the point any filter is not enabled. For example:
If Filter 2 And_Previous bit is set (bit 1 of Filter 2 command is set) but Filter 1 is not enabled (bit 0 of in Filter 1 command is reset), then only Filter 2 result is considered.
If Filter 2 And_Previous bit is set (bit 1 of Filter 2 command is set), Filter 3 And_Previous bit is set (bit 1 of Filter 3 command is set), but Filter 1 is not enabled (bit 0 of in Filter 1 command is reset), then only Filter 2 result ANDed with Filter 3
result is considered.
If Filter 2 And_Previous bit is set (bit 1 of Filter 2 command is set), Filter 3 And_Previous bit is set (bit 1 of Filter 3 command is set), but Filter 2 is not enabled (bit 0 of in Filter 2 command is reset), then since setting of Filter 2 And_Previous bit
has no effect only Filter 1 result ORed with Filter 3 result is considered.
- If filters chained by And_Previous bit setting have complementary programming, then a frame may never pass the AND chained filter. For example, if Filter 2 And_Previous bit is set (bit 1 of Filter 2 command is set), Filter 1 Address_Type bit is set (bit 3 of Filter 1 command is set) indicating multicast detection and Filter 2 Address_Type bit is reset (bit 3 of Filter 2 command is reset) indicating unicast detection or vice versa, a remote wakeup frame does not pass the AND chained filter as a remote wakeup frame cannot be of both unicast and multicast address type.

Figure 43-61. MAC_RWK_Packet_Filter Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WKUPFRMFTR																															
R/W-0h																															

Table 43-115. MAC_RWK_Packet_Filter Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WKUPFRMFTR	R/W	0h	RWK Packet Filter This field contains the various controls of RWK Packet filter.

43.7.3.23 MAC_LPI_Control_Status Register (Offset = D0h) [Reset = 0h]

MAC_LPI_Control_Status is shown in Figure 43-62 and described in Table 43-116.

Return to the Summary Table.

The LPI Control and Status Register controls the LPI functions and provides the LPI interrupt status. The status bits are cleared when this register is read.

Figure 43-62. MAC_LPI_Control_Status Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED		LPITCSE	LPIATE	LPITXA	RESERVED	PLS	LPIEN
R-0h		R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED						RLPIST	TLPIST
R-0h						R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED				RLPIEX	RLPIEN	TLPIEX	TLPIEN
R-0h				R-0h	R-0h	R-0h	R-0h

Table 43-116. MAC_LPI_Control_Status Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reserved.
21	LPITCSE	R/W	0h	LPI Tx Clock Stop Enable When this bit is set, the MAC asserts sbd_tx_clk_gating_ctrl_o signal high after it enters Tx LPI mode to indicate that the Tx clock to MAC can be stopped. When this bit is reset, the MAC does not assert sbd_tx_clk_gating_ctrl_o signal high after it enters Tx LPI mode. If RGMII Interface is selected, the Tx clock is required for transmitting the LPI pattern. The Tx Clock cannot be gated and so the LPITCSE bit cannot be programmed. 0h = LPI Tx Clock Stop is disabled : 0x0 1h = LPI Tx Clock Stop is enabled : 0x1
20	LPIATE	R/W	0h	LPI Timer Enable This bit controls the automatic entry of the MAC Transmitter into and exit out of the LPI state. When LPIATE, LPITXA and LPIEN bits are set, the MAC Transmitter enters LPI state only when the complete MAC TX data path is IDLE for a period indicated by the MAC_LPI_Entry_Timer register. After entering LPI state, if the data path becomes non-IDLE (due to a new packet being accepted for transmission), the Transmitter exits LPI state but does not clear LPIEN bit. This enables the re-entry into LPI state when it is IDLE again. When LPIATE is 0, the LPI Auto timer is disabled and MAC Transmitter enters LPI state based on the settings of LPITXA and LPIEN bit descriptions. 0h = LPI Timer is disabled : 0x0 1h = LPI Timer is enabled : 0x1

Table 43-116. MAC_LPI_Control_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	LPITXA	R/W	0h	LPI Tx Automate This bit controls the behavior of the MAC when it is entering or coming out of the LPI mode on the Transmit side. This bit is not functional in the EQOS-CORE configurations in which the Tx clock gating is done during the LPI mode. If the LPITXA and LPIEN bits are set to 1, the MAC enters the LPI mode only after all outstanding packets (in the core) and pending packets (in the application interface) have been transmitted. The MAC comes out of the LPI mode when the application sends any packet for transmission or the application issues a Tx FIFO Flush command. In addition, the MAC automatically clears the LPIEN bit when it exits the LPI state. If Tx FIFO Flush is set in the FTQ bit of MTL_TxQ0_Operation_Mode register, when the MAC is in the LPI mode, it exits the LPI mode. When this bit is 0, the LPIEN bit directly controls behavior of the MAC when it is entering or coming out of the LPI mode. 0h = LPI Tx Automate is disabled : 0x0 1h = LPI Tx Automate is enabled : 0x1
18	RESERVED	R	0h	Reserved.
17	PLS	R/W	0h	PHY Link Status This bit indicates the link status of the PHY. The MAC Transmitter asserts the LPI pattern only when the link status is up (OKAY) at least for the time indicated by the LPI LS TIMER. When this bit is set, the link is considered to be okay (UP) and when this bit is reset, the link is considered to be down. 0h = link is down : 0x0 1h = link is okay (UP) : 0x1
16	LPIEN	R/W	0h	LPI Enable When this bit is set, it instructs the MAC Transmitter to enter the LPI state. When this bit is reset, it instructs the MAC to exit the LPI state and resume normal transmission. This bit is cleared when the LPITXA bit is set and the MAC exits the LPI state because of the arrival of a new packet for transmission. 0h = LPI state is disabled : 0x0 1h = LPI state is enabled : 0x1
15-10	RESERVED	R	0h	Reserved.
9	RLPIST	R	0h	Receive LPI State When this bit is set, it indicates that the MAC is receiving the LPI pattern on the GMII or MII interface. 0h = Receive LPI state not detected : 0x0 1h = Receive LPI state detected : 0x1
8	TLPIST	R	0h	Transmit LPI State When this bit is set, it indicates that the MAC is transmitting the LPI pattern on the GMII or MII interface. 0h = Transmit LPI state not detected : 0x0 1h = Transmit LPI state detected : 0x1
7-4	RESERVED	R	0h	Reserved.
3	RLPIEX	R	0h	Receive LPI Exit When this bit is set, it indicates that the MAC Receiver has stopped receiving the LPI pattern on the GMII or MII interface, exited the LPI state, and resumed the normal reception. This bit is cleared by a read into this register (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Note: This bit may not be set if the MAC stops receiving the LPI pattern for a very short duration, such as, less than three clock cycles of CSR clock. 0h = Receive LPI exit not detected : 0x0 1h = Receive LPI exit detected : 0x1

Table 43-116. MAC_LPI_Control_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	RLPIEN	R	0h	Receive LPI Entry When this bit is set, it indicates that the MAC Receiver has received an LPI pattern and entered the LPI state. This bit is cleared by a read into this register (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Note: This bit may not be set if the MAC stops receiving the LPI pattern for a very short duration, such as, less than three clock cycles of CSR clock. 0h = Receive LPI entry not detected : 0x0 1h = Receive LPI entry detected : 0x1
1	TLPIEX	R	0h	Transmit LPI Exit When this bit is set, it indicates that the MAC transmitter exited the LPI state after the application cleared the LPIEN bit and the LPI TW Timer has expired. This bit is cleared by a read into this register (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). 0h = Transmit LPI exit not detected : 0x0 1h = Transmit LPI exit detected : 0x1
0	TLPIEN	R	0h	Transmit LPI Entry When this bit is set, it indicates that the MAC Transmitter has entered the LPI state because of the setting of the LPIEN bit. This bit is cleared by a read into this register (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). 0h = Transmit LPI entry not detected : 0x0 1h = Transmit LPI entry detected : 0x1

43.7.3.24 MAC_LPI_Timers_Control Register (Offset = D4h) [Reset = 03E80000h]

MAC_LPI_Timers_Control is shown in Figure 43-63 and described in Table 43-117.

Return to the Summary Table.

The LPI Timers Control register controls the timeout values in the LPI states. It specifies the time for which the MAC transmits the LPI pattern and also the time for which the MAC waits before resuming the normal transmission.

Figure 43-63. MAC_LPI_Timers_Control Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED						LST									
R-0h						R/W-3E8h									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TWT															
R/W-0h															

Table 43-117. MAC_LPI_Timers_Control Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved.
25-16	LST	R/W	3E8h	LPI LS Timer This field specifies the minimum time (in milliseconds) for which the link status from the PHY should be up (OKAY) before the LPI pattern can be transmitted to the PHY. The MAC does not transmit the LPI pattern even when the LPIEN bit is set unless the LPI LS Timer reaches the programmed terminal count. The default value of the LPI LS Timer is 1000 (1 sec) as defined in the IEEE standard.
15-0	TWT	R/W	0h	LPI TW Timer This field specifies the minimum time (in microseconds) for which the MAC waits after it stops transmitting the LPI pattern to the PHY and before it resumes the normal transmission. The TLPIEX status bit is set after the expiry of this timer.

43.7.3.25 MAC_LPI_Entry_Timer Register (Offset = D8h) [Reset = 0h]

MAC_LPI_Entry_Timer is shown in Figure 43-64 and described in Table 43-118.

Return to the Summary Table.

This register controls the Tx LPI entry timer. This counter is enabled only when bit[20](LPITE) bit of MAC_LPI_Control_Status is set to 1.

Figure 43-64. MAC_LPI_Entry_Timer Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED												LPIET			
R-0h												R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LPIET													RESERVED		
R/W-0h													R-0h		

Table 43-118. MAC_LPI_Entry_Timer Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved.
19-3	LPIET	R/W	0h	LPI Entry Timer This field specifies the time in microseconds the MAC waits to enter LPI mode, after it has transmitted all the frames. This field is valid and used only when LPITE and LPITXA are set to 1. Bits [2:0] are read-only so that the granularity of this timer is in steps of 8 micro-seconds.
2-0	RESERVED	R	0h	Reserved.

43.7.3.26 MAC_1US_Tic_Counter Register (Offset = DCh) [Reset = 63h]

MAC_1US_Tic_Counter is shown in Figure 43-65 and described in Table 43-119.

Return to the Summary Table.

This register controls the generation of the Reference time (1 microsecond tic) for all the LPI timers. This timer has to be programmed by the software initially.

Figure 43-65. MAC_1US_Tic_Counter Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				TIC_1US_CNTR											
R-0h																				R/W-63h											

Table 43-119. MAC_1US_Tic_Counter Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved.
11-0	TIC_1US_CNTR	R/W	63h	1US TIC Counter The application must program this counter so that the number of clock cycles of CSR clock is 1us. (Subtract 1 from the value before programming). For example if the CSR clock is 100MHz then this field needs to be programmed to value 100 - 1 = 99 (which is 0x63). This is required to generate the 1US events that are used to update some of the EEE related counters.

43.7.3.27 MAC_Version Register (Offset = 110h) [Reset = 50h]

MAC_Version is shown in Figure 43-66 and described in Table 43-120.

Return to the Summary Table.

The version register identifies the version of the DWC_ether_qos. This register contains two bytes: one that Synopsys uses to identify the core release number, and the other that you set while configuring the core.

Figure 43-66. MAC_Version Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																USERVER								SNPSVER							
R-0h																R-0h								R-50h							

Table 43-120. MAC_Version Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved.
15-8	USERVER	R	0h	User-defined Version (configured with coreConsultant)
7-0	SNPSVER	R	50h	Synopsys-defined Version

43.7.3.28 MAC_Debug Register (Offset = 114h) [Reset = 0h]

MAC_Debug is shown in Figure 43-67 and described in Table 43-121.

Return to the Summary Table.

The Debug register provides the debug status of various MAC blocks.

Figure 43-67. MAC_Debug Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED					TFCSTS		TPESTS
R-0h					R-0h		R-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					RFCFCSTS		RPESTS
R-0h					R-0h		R-0h

Table 43-121. MAC_Debug Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved.
18-17	TFCSTS	R	0h	MAC Transmit Packet Controller Status This field indicates the state of the MAC Transmit Packet Controller module. 0h = Idle state : 0x0 1h = Waiting for one of the following: Status of the previous packet OR IPG or backoff period to be over : 0x1 2h = Generating and transmitting a Pause control packet (in full-duplex mode) : 0x2 3h = Transferring input packet for transmission : 0x3
16	TPESTS	R	0h	MAC GMII or MII Transmit Protocol Engine Status When this bit is set, it indicates that the MAC GMII or MII transmit protocol engine is actively transmitting data, and it is not in the Idle state. 0h = MAC GMII or MII Transmit Protocol Engine Status not detected : 0x0 1h = MAC GMII or MII Transmit Protocol Engine Status detected : 0x1
15-3	RESERVED	R	0h	Reserved.
2-1	RFCFCSTS	R	0h	MAC Receive Packet Controller FIFO Status When this bit is set, this field indicates the active state of the small FIFO Read and Write controllers of the MAC Receive Packet Controller module.
0	RPESTS	R	0h	MAC GMII or MII Receive Protocol Engine Status When this bit is set, it indicates that the MAC GMII or MII receive protocol engine is actively receiving data, and it is not in the Idle state. 0h = MAC GMII or MII Receive Protocol Engine Status not detected : 0x0 1h = MAC GMII or MII Receive Protocol Engine Status detected : 0x1

43.7.3.29 MAC_HW_Feature0 Register (Offset = 11Ch) [Reset = 0E1D73F5h]

MAC_HW_Feature0 is shown in Figure 43-68 and described in Table 43-122.

Return to the Summary Table.

This register indicates the presence of first set of the optional features or functions of the DWC_ether_qos. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks. Note: All bits are set or reset according to the features selected while configuring the core in coreConsultant.

Figure 43-68. MAC_HW_Feature0 Register

31	30	29	28	27	26	25	24
RESERVED	ACTPHYSEL			SAVLANINS	TSSTSSEL		MACADR64SEL
R-0h	R-0h			R-1h	R-3h		R-0h
23	22	21	20	19	18	17	16
MACADR32SEL	ADDMACADRSEL					RESERVED	RXCOESEL
R-0h	R-7h					R-0h	R-1h
15	14	13	12	11	10	9	8
RESERVED	TXCOESEL	EEESEL	TSSEL	RESERVED		ARPOFFSEL	MMCSEL
R-0h	R-1h	R-1h	R-1h	R-0h		R-1h	R-1h
7	6	5	4	3	2	1	0
MGKSEL	RWKSEL	SMASEL	VLHASH	PCSSEL	HDSEL	GMIISEL	MIISEL
R-1h	R-1h	R-1h	R-1h	R-0h	R-1h	R-0h	R-1h

Table 43-122. MAC_HW_Feature0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved.
30-28	ACTPHYSEL	R	0h	Active PHY Selected When you have multiple PHY interfaces in your configuration, this field indicates the sampled value of phy_intf_sel_i during reset de-assertion. 0h = GMII or MII : 0x0 1h = RGMII : 0x1 2h = SGMII : 0x2 3h = TBI : 0x3 4h = RMII : 0x4 5h = RTBI : 0x5 6h = SMII : 0x6 7h = RevMII : 0x7
27	SAVLANINS	R	1h	Source Address or VLAN Insertion Enable This bit is set to 1 when the Enable SA and VLAN Insertion on Tx option is selected 0h = Source Address or VLAN Insertion Enable option is not selected : 0x0 1h = Source Address or VLAN Insertion Enable option is selected : 0x1
26-25	TSSTSSEL	R	3h	Timestamp System Time Source This bit indicates the source of the Timestamp system time: This bit is set to 1 when the Enable IEEE 1588 Timestamp Support option is selected 0h = Internal : 0x0 1h = External : 0x1 2h = Both : 0x2 3h = Reserved : 0x3

Table 43-122. MAC_HW_Feature0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24	MACADR64SEL	R	0h	MAC Addresses 64-127 Selected This bit is set to 1 when the Enable Additional 64 MAC Address Registers (64-127) option is selected 0h = MAC Addresses 64-127 Select option is not selected : 0x0 1h = MAC Addresses 64-127 Select option is selected : 0x1
23	MACADR32SEL	R	0h	MAC Addresses 32-63 Selected This bit is set to 1 when the Enable Additional 32 MAC Address Registers (32-63) option is selected 0h = MAC Addresses 32-63 Select option is not selected : 0x0 1h = MAC Addresses 32-63 Select option is selected : 0x1
22-18	ADDMACADRSEL	R	7h	MAC Addresses 1-31 Selected This bit is set to 1 when the Enable Additional 1-31 MAC Address Registers option is selected
17	RESERVED	R	0h	Reserved.
16	RXCOESEL	R	1h	Receive Checksum Offload Enabled This bit is set to 1 when the Enable Receive TCP/IP Checksum Check option is selected 0h = Receive Checksum Offload Enable option is not selected : 0x0 1h = Receive Checksum Offload Enable option is selected : 0x1
15	RESERVED	R	0h	Reserved.
14	TXCOESEL	R	1h	Transmit Checksum Offload Enabled This bit is set to 1 when the Enable Transmit TCP/IP Checksum Insertion option is selected 0h = Transmit Checksum Offload Enable option is not selected : 0x0 1h = Transmit Checksum Offload Enable option is selected : 0x1
13	EEESEL	R	1h	Energy Efficient Ethernet Enabled This bit is set to 1 when the Enable Energy Efficient Ethernet (EEE) option is selected 0h = Energy Efficient Ethernet Enable option is not selected : 0x0 1h = Energy Efficient Ethernet Enable option is selected : 0x1
12	TSSEL	R	1h	IEEE 1588-2008 Timestamp Enabled This bit is set to 1 when the Enable IEEE 1588 Timestamp Support option is selected 0h = IEEE 1588-2008 Timestamp Enable option is not selected : 0x0 1h = IEEE 1588-2008 Timestamp Enable option is selected : 0x1
11-10	RESERVED	R	0h	Reserved.
9	ARPOFFSEL	R	1h	ARP Offload Enabled This bit is set to 1 when the Enable IPv4 ARP Offload option is selected 0h = ARP Offload Enable option is not selected : 0x0 1h = ARP Offload Enable option is selected : 0x1
8	MMCSEL	R	1h	RMON Module Enable This bit is set to 1 when the Enable MAC Management Counters (MMC) option is selected 0h = RMON Module Enable option is not selected : 0x0 1h = RMON Module Enable option is selected : 0x1
7	MGKSEL	R	1h	PMT Magic Packet Enable This bit is set to 1 when the Enable Magic Packet Detection option is selected 0h = PMT Magic Packet Enable option is not selected : 0x0 1h = PMT Magic Packet Enable option is selected : 0x1
6	RWKSEL	R	1h	PMT Remote Wake-up Packet Enable This bit is set to 1 when the Enable Remote Wake-Up Packet Detection option is selected 0h = PMT Remote Wake-up Packet Enable option is not selected : 0x0 1h = PMT Remote Wake-up Packet Enable option is selected : 0x1

Table 43-122. MAC_HW_Feature0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	SMASEL	R	1h	SMA (MDIO) Interface This bit is set to 1 when the Enable Station Management (MDIO Interface) option is selected 0h = SMA (MDIO) Interface not selected : 0x0 1h = SMA (MDIO) Interface selected : 0x1
4	VLHASH	R	1h	VLAN Hash Filter Selected This bit is set to 1 when the Enable VLAN Hash Table Based Filtering option is selected 0h = VLAN Hash Filter not selected : 0x0 1h = VLAN Hash Filter selected : 0x1
3	PCSSEL	R	0h	PCS Registers (TBI, SGMII, or RTBI PHY interface) This bit is set to 1 when the TBI, SGMII, or RTBI PHY interface option is selected 0h = No PCS Registers (TBI, SGMII, or RTBI PHY interface) : 0x0 1h = PCS Registers (TBI, SGMII, or RTBI PHY interface) : 0x1
2	HDSEL	R	1h	Half-duplex Support This bit is set to 1 when the half-duplex mode is selected 0h = No Half-duplex support : 0x0 1h = Half-duplex support : 0x1
1	GMIISEL	R	0h	1000 Mbps Support This bit is set to 1 when 1000 Mbps is selected as the Mode of Operation 0h = No 1000 Mbps support : 0x0 1h = 1000 Mbps support : 0x1
0	MIISEL	R	1h	10 or 100 Mbps Support This bit is set to 1 when 10/100 Mbps is selected as the Mode of Operation 0h = No 10 or 100 Mbps support : 0x0 1h = 10 or 100 Mbps support : 0x1

43.7.3.30 MAC_HW_Feature1 Register (Offset = 120h) [Reset = 218E3965h]

MAC_HW_Feature1 is shown in Figure 43-69 and described in Table 43-123.

Return to the Summary Table.

This register indicates the presence of second set of the optional features or functions of the DWC_ether_qos. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks.
Note: All bits are set or reset according to the features selected while configuring the core in coreConsultant.

Figure 43-69. MAC_HW_Feature1 Register

31	30	29	28	27	26	25	24
RESERVED	L3L4FNUM				RESERVED	HASHTBLSZ	
R-0h	R-4h				R-0h	R-1h	
23	22	21	20	19	18	17	16
POUOST	RESERVED	RAVSEL	AVSEL	DBGMEMA	TSOEN	SPHEN	DCBEN
R-1h	R-0h	R-0h	R-0h	R-1h	R-1h	R-1h	R-0h
15	14	13	12	11	10	9	8
ADDR64		ADVTHWORD	PTOEN	OSTEN	TXFIFOSIZE		
R-0h		R-1h	R-1h	R-1h	R-5h		
7	6	5	4	3	2	1	0
TXFIFOSIZE		SPRAM	RXFIFOSIZE				
R-5h		R-1h	R-5h				

Table 43-123. MAC_HW_Feature1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved.
30-27	L3L4FNUM	R	4h	Total number of L3 or L4 Filters This field indicates the total number of L3 or L4 filters: 0h = No L3 or L4 Filter : 0x0 1h = 1 L3 or L4 Filter : 0x1 2h = 2 L3 or L4 Filters : 0x2 3h = 3 L3 or L4 Filters : 0x3 4h = 4 L3 or L4 Filters : 0x4 5h = 5 L3 or L4 Filters : 0x5 6h = 6 L3 or L4 Filters : 0x6 7h = 7 L3 or L4 Filters : 0x7 8h = 8 L3 or L4 Filters : 0x8
26	RESERVED	R	0h	Reserved.
25-24	HASHTBLSZ	R	1h	Hash Table Size This field indicates the size of the hash table: 0h = No hash table : 0x0 1h = 64 : 0x1 2h = 128 : 0x2 3h = 256 : 0x3
23	POUOST	R	1h	One Step for PTP over UDP/IP Feature Enable This bit is set to 1 when the Enable One step timestamp for PTP over UDP/IP feature is selected. 0h = One Step for PTP over UDP/IP Feature is not selected : 0x0 1h = One Step for PTP over UDP/IP Feature is selected : 0x1
22	RESERVED	R	0h	Reserved.
21	RAVSEL	R	0h	Rx Side Only AV Feature Enable This bit is set to 1 when the Enable Audio Video Bridging option on Rx Side Only is selected. 0h = Rx Side Only AV Feature is not selected : 0x0 1h = Rx Side Only AV Feature is selected : 0x1

Table 43-123. MAC_HW_Feature1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	AVSEL	R	0h	AV Feature Enable This bit is set to 1 when the Enable Audio Video Bridging option is selected. 0h = AV Feature is not selected : 0x0 1h = AV Feature is selected : 0x1
19	DBGMEMA	R	1h	DMA Debug Registers Enable This bit is set to 1 when the Debug Mode Enable option is selected 0h = DMA Debug Registers option is not selected : 0x0 1h = DMA Debug Registers option is selected : 0x1
18	TSOEN	R	1h	TCP Segmentation Offload Enable This bit is set to 1 when the Enable TCP Segmentation Offloading for TCP/IP Packets option is selected 0h = TCP Segmentation Offload Feature is not selected : 0x0 1h = TCP Segmentation Offload Feature is selected : 0x1
17	SPHEN	R	1h	Split Header Feature Enable This bit is set to 1 when the Enable Split Header Structure option is selected 0h = Split Header Feature is not selected : 0x0 1h = Split Header Feature is selected : 0x1
16	DCBEN	R	0h	DCB Feature Enable This bit is set to 1 when the Enable Data Center Bridging option is selected 0h = DCB Feature is not selected : 0x0 1h = DCB Feature is selected : 0x1
15-14	ADDR64	R	0h	Address Width. This field indicates the configured address width: 0h = 32 : 0x0 1h = 40 : 0x1 2h = 48 : 0x2 3h = Reserved : 0x3
13	ADVTHWORD	R	1h	IEEE 1588 High Word Register Enable This bit is set to 1 when the Add IEEE 1588 Higher Word Register option is selected 0h = IEEE 1588 High Word Register option is not selected : 0x0 1h = IEEE 1588 High Word Register option is selected : 0x1
12	PTOEN	R	1h	PTP Offload Enable This bit is set to 1 when the Enable PTP Timestamp Offload Feature is selected. 0h = PTP Offload feature is not selected : 0x0 1h = PTP Offload feature is selected : 0x1
11	OSTEN	R	1h	One-Step Timestamping Enable This bit is set to 1 when the Enable One-Step Timestamp Feature is selected. 0h = One-Step Timestamping feature is not selected : 0x0 1h = One-Step Timestamping feature is selected : 0x1
10-6	TXFIFOSIZE	R	5h	MTL Transmit FIFO Size This field contains the configured value of MTL Tx FIFO in bytes expressed as Log to base 2 minus 7, that is, Log2(TXFIFO_SIZE) -7: 0h = 128 bytes : 0x0 1h = 256 bytes : 0x1 2h = 512 bytes : 0x2 3h = 1024 bytes : 0x3 4h = 2048 bytes : 0x4 5h = 4096 bytes : 0x5 6h = 8192 bytes : 0x6 7h = 16384 bytes : 0x7 8h = 32 KB : 0x8 9h = 64 KB : 0x9 Ah = 128 KB : 0xa Bh = Reserved : 0xb

Table 43-123. MAC_HW_Feature1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	SPRAM	R	1h	Single Port RAM Enable This bit is set to 1 when the Use single port RAM Feature is selected. 0h = Single Port RAM feature is not selected : 0x0 1h = Single Port RAM feature is selected : 0x1
4-0	RXFIFOSIZE	R	5h	MTL Receive FIFO Size This field contains the configured value of MTL Rx FIFO in bytes expressed as Log to base 2 minus 7, that is, Log2(RXFIFO_SIZE) -7: 0h = 128 bytes : 0x0 1h = 256 bytes : 0x1 2h = 512 bytes : 0x2 3h = 1024 bytes : 0x3 4h = 2048 bytes : 0x4 5h = 4096 bytes : 0x5 6h = 8192 bytes : 0x6 7h = 16384 bytes : 0x7 8h = 32 KB : 0x8 9h = 64 KB : 0x9 Ah = 128 KB : 0xa Bh = 256 KB : 0xb Ch = Reserved : 0xc

43.7.3.31 MAC_HW_Feature2 Register (Offset = 124h) [Reset = 22041041h]

MAC_HW_Feature2 is shown in Figure 43-70 and described in Table 43-124.

Return to the Summary Table.

This register indicates the presence of third set of the optional features or functions of the DWC_ether_qos. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks.

Figure 43-70. MAC_HW_Feature2 Register

31	30	29	28	27	26	25	24
RESERVED	AUXSNAPNUM			RESERVED	PPSOUTNUM		
R-0h	R-2h			R-0h	R-2h		
23	22	21	20	19	18	17	16
RESERVED		TXCHCNT				RESERVED	
R-0h		R-1h				R-0h	
15	14	13	12	11	10	9	8
RXCHCNT				RESERVED		TXQCNT	
R-1h				R-0h		R-1h	
7	6	5	4	3	2	1	0
TXQCNT		RESERVED		RXQCNT			
R-1h		R-0h		R-1h			

Table 43-124. MAC_HW_Feature2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved.
30-28	AUXSNAPNUM	R	2h	Number of Auxiliary Snapshot Inputs This field indicates the number of auxiliary snapshot inputs: 0h = No auxiliary input : 0x0 1h = 1 auxiliary input : 0x1 2h = 2 auxiliary input : 0x2 3h = 3 auxiliary input : 0x3 4h = 4 auxiliary input : 0x4 5h = Reserved : 0x5
27	RESERVED	R	0h	Reserved.
26-24	PPSOUTNUM	R	2h	Number of PPS Outputs This field indicates the number of PPS outputs: 0h = No PPS output : 0x0 1h = 1 PPS output : 0x1 2h = 2 PPS output : 0x2 3h = 3 PPS output : 0x3 4h = 4 PPS output : 0x4 5h = Reserved : 0x5
23-22	RESERVED	R	0h	Reserved.
21-18	TXCHCNT	R	1h	Number of DMA Transmit Channels This field indicates the number of DMA Transmit channels: 0h = 1 MTL Tx Channel : 0x0 1h = 2 MTL Tx Channels : 0x1 2h = 3 MTL Tx Channels : 0x2 3h = 4 MTL Tx Channels : 0x3 4h = 5 MTL Tx Channels : 0x4 5h = 6 MTL Tx Channels : 0x5 6h = 7 MTL Tx Channels : 0x6 7h = 8 MTL Tx Channels : 0x7
17-16	RESERVED	R	0h	Reserved.

Table 43-124. MAC_HW_Feature2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-12	RXCHCNT	R	1h	Number of DMA Receive Channels This field indicates the number of DMA Receive channels: 0h = 1 MTL Rx Channel : 0x0 1h = 2 MTL Rx Channels : 0x1 2h = 3 MTL Rx Channels : 0x2 3h = 4 MTL Rx Channels : 0x3 4h = 5 MTL Rx Channels : 0x4 5h = 6 MTL Rx Channels : 0x5 6h = 7 MTL Rx Channels : 0x6 7h = 8 MTL Rx Channels : 0x7
11-10	RESERVED	R	0h	Reserved.
9-6	TXQCNT	R	1h	Number of MTL Transmit Queues This field indicates the number of MTL Transmit queues: 0h = 1 MTL Tx Queue : 0x0 1h = 2 MTL Tx Queues : 0x1 2h = 3 MTL Tx Queues : 0x2 3h = 4 MTL Tx Queues : 0x3 4h = 5 MTL Tx Queues : 0x4 5h = 6 MTL Tx Queues : 0x5 6h = 7 MTL Tx Queues : 0x6 7h = 8 MTL Tx Queues : 0x7
5-4	RESERVED	R	0h	Reserved.
3-0	RXQCNT	R	1h	Number of MTL Receive Queues This field indicates the number of MTL Receive queues: 0h = 1 MTL Rx Queue : 0x0 1h = 2 MTL Rx Queues : 0x1 2h = 3 MTL Rx Queues : 0x2 3h = 4 MTL Rx Queues : 0x3 4h = 5 MTL Rx Queues : 0x4 5h = 6 MTL Rx Queues : 0x5 6h = 7 MTL Rx Queues : 0x6 7h = 8 MTL Rx Queues : 0x7

43.7.3.32 MAC_HW_Feature3 Register (Offset = 128h) [Reset = 00320031h]

MAC_HW_Feature3 is shown in Figure 43-71 and described in Table 43-125.

Return to the Summary Table.

This register indicates the presence of fourth set the optional features or functions of the DWC_ether_qos. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks.

Figure 43-71. MAC_HW_Feature3 Register

31	30	29	28	27	26	25	24
RESERVED				TBSSEL	FPESEL	RESERVED	ESTTISW
R-0h				R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
ESTTISW	RESERVED	ESTWID		ESTDEP			ESTSEL
R-0h	R-0h	R-3h		R-1h			R-0h
15	14	13	12	11	10	9	8
RESERVED						PDUPSEL	DBGSSEL
R-0h						R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED		DVLAN	CBTISEL	RESERVED	NRVF		
R-0h		R-1h	R-1h	R-0h	R-1h		

Table 43-125. MAC_HW_Feature3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved.
27	TBSSEL	R	0h	Time Based Scheduling Enable This bit is set to 1 when the Time Based Scheduling feature is selected. 0h = Time Based Scheduling Enable feature is not selected : 0x0 1h = Time Based Scheduling Enable feature is selected : 0x1
26	FPESEL	R	0h	Frame Preemption Enable This bit is set to 1 when the Enable Frame preemption feature is selected. 0h = Frame Preemption Enable feature is not selected : 0x0 1h = Frame Preemption Enable feature is selected : 0x1
25	RESERVED	R	0h	Reserved.
24-23	ESTTISW	R	0h	Width of the Left Shift Amount for Time Interval This field indicates the width of programmable left shift field for Time Interval 0h = 0 : 0x0 1h = 1 : 0x1 2h = 2 : 0x2 3h = 3 : 0x3
22	RESERVED	R	0h	Reserved.
21-20	ESTWID	R	3h	Width of the Time Interval field in the Gate Control List This field indicates the width of the Configured Time Interval Field 0h = Width not configured : 0x0 1h = 16 : 0x1 2h = 20 : 0x2 3h = 24 : 0x3

Table 43-125. MAC_HW_Feature3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-17	ESTDEP	R	1h	Depth of the Gate Control List This field indicates the depth of Gate Control list expressed as Log2(DWC_EQOS_EST_DEP)-5 0h = No Depth configured : 0x0 1h = 64 : 0x1 2h = 128 : 0x2 3h = 256 : 0x3 4h = 512 : 0x4 5h = 1024 : 0x5 6h = Reserved : 0x6
16	ESTSEL	R	0h	Enhancements to Scheduling Traffic Enable This bit is set to 1 when the Enable Enhancements to Scheduling Traffic feature is selected. 0h = Enable Enhancements to Scheduling Traffic feature is not selected : 0x0 1h = Enable Enhancements to Scheduling Traffic feature is selected : 0x1
15-10	RESERVED	R	0h	Reserved.
9	PDUPSEL	R	0h	Broadcast/Multicast Packet Duplication This bit is set to 1 when the Broadcast/Multicast Packet Duplication feature is selected. 0h = Broadcast/Multicast Packet Duplication feature is not selected : 0x0 1h = Broadcast/Multicast Packet Duplication feature is selected : 0x1
8	DBGSSEL	R	0h	Debug Bus Support Enable This bit is set to 1 when the Enable Debug Bus Support feature is selected. 0h = Debug Bus Support Enable feature is not selected : 0x0 1h = Debug Bus Support Enable feature is selected : 0x1
7-6	RESERVED	R	0h	Reserved.
5	DVLAN	R	1h	
4	CBTISEL	R	1h	Queue/Channel based VLAN tag insertion on Tx Enable This bit is set to 1 when the Enable Queue/Channel based VLAN tag insertion on Tx Feature is selected. 0h = Enable Queue/Channel based VLAN tag insertion on Tx feature is not selected : 0x0 1h = Enable Queue/Channel based VLAN tag insertion on Tx feature is selected : 0x1
3	RESERVED	R	0h	Reserved.
2-0	NRVF	R	1h	Number of Extended VLAN Tag Filters Enabled This field indicates the Number of Extended VLAN Tag Filters selected: 0h = No Extended Rx VLAN Filters : 0x0 1h = 4 Extended Rx VLAN Filters : 0x1 2h = 8 Extended Rx VLAN Filters : 0x2 3h = 16 Extended Rx VLAN Filters : 0x3 4h = 24 Extended Rx VLAN Filters : 0x4 5h = 32 Extended Rx VLAN Filters : 0x5 6h = Reserved : 0x6

43.7.3.33 MAC_MDIO_Address Register (Offset = 200h) [Reset = 0h]

MAC_MDIO_Address is shown in Figure 43-72 and described in Table 43-126.

Return to the Summary Table.

The MDIO Address register controls the management cycles to external PHY through a management interface.

Figure 43-72. MAC_MDIO_Address Register

31	30	29	28	27	26	25	24
RESERVED				PSE	BTB	PA	
R-0h				R/W-0h	R/W-0h	R/W-0h	
23	22	21	20	19	18	17	16
PA			RDA				
R/W-0h			R/W-0h				
15	14	13	12	11	10	9	8
RESERVED	NTC			CR			
R-0h	R/W-0h			R/W-0h			
7	6	5	4	3	2	1	0
RESERVED			SKAP	GOC_1	GOC_0	C45E	GB
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 43-126. MAC_MDIO_Address Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved.
27	PSE	R/W	0h	Preamble Suppression Enable When this bit is set, the SMA suppresses the 32-bit preamble and transmits MDIO frames with only 1 preamble bit. When this bit is 0, the MDIO frame always has 32 bits of preamble as defined in the IEEE specifications. 0h = Preamble Suppression disabled : 0x0 1h = Preamble Suppression enabled : 0x1
26	BTB	R/W	0h	Back to Back transactions When this bit is set and the NTC has value greater than 0, then the MAC informs the completion of a read or write command at the end of frame transfer (before the trailing clocks are transmitted). The software can thus initiate the next command which is executed immediately irrespective of the number trailing clocks generated for the previous frame. When this bit is reset, then the read/write command completion (GB is cleared)only after the trailing clocks are generated. In this mode, it is ensured that the NTC is always generated after each frame. This bit must not be set when NTC=0. 0h = Back to Back transactions disabled : 0x0 1h = Back to Back transactions enabled : 0x1
25-21	PA	R/W	0h	Physical Layer Address This field indicates which Clause 22 PHY devices (out of 32 devices) the MAC is accessing. For RevMII, this field gives the PHY Address of the RevMII module. This field indicates which Clause 45 capable PHYs (out of 32 PHYs) the MAC is accessing.

Table 43-126. MAC_MDIO_Address Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20-16	RDA	R/W	0h	Register/Device Address These bits select the PHY register in selected Clause 22 PHY device. For RevMII, these bits select the CSR register in the RevMII Registers set. These bits select the Device (MMD) in selected Clause 45 capable PHY.
15	RESERVED	R	0h	Reserved.
14-12	NTC	R/W	0h	Number of Trailing Clocks This field controls the number of trailing clock cycles generated on gmii_mdc_o (MDC) after the end of transmission of MDIO frame. The valid values can be from 0 to 7. Programming the value to 3'h3 indicates that there are additional three clock cycles on the MDC line after the end of MDIO frame transfer.
11-8	CR	R/W	0h	CSR Clock Range The CSR Clock Range selection determines the frequency of the MDC clock according to the CSR clock frequency used in your design: - 0000: CSR clock = 60-100 MHz MDC clock = CSR clock/42 - 0001: CSR clock = 100-150 MHz MDC clock = CSR clock/62 - 0010: CSR clock = 20-35 MHz MDC clock = CSR clock/16 - 0011: CSR clock = 35-60 MHz MDC clock = CSR clock/26 - 0100: CSR clock = 150-250 MHz MDC clock = CSR clock/102 - 0101: CSR clock = 250-300 MHz MDC clock = CSR clock/124 - 0110: CSR clock = 300-500 MHz MDC clock = CSR clock/204 - 0111: CSR clock = 500-800 MHz MDC clock = CSR clock/324 The suggested range of CSR clock frequency applicable for each value (when Bit 11 = 0) ensures that the MDC clock is approximately between 1.0 MHz to 2.5 MHz freqency range. When Bit 11 is set, you can achieve a higher frequency of the MDC clock than the frequency limit of 2.5 MHz (specified in the IEEE 802.3) and program a clock divider of lower value. For example, when CSR clock is of 100 MHz frequency and you program these bits as 1010, the resultant MDC clock is of 12.5 MHz which is above the range specified in IEEE 802.3. Program the following values only if the interfacing chips support faster MDC clocks: - 1000: CSR clock/4 - 1001: CSR clock/6 - 1010: CSR clock/8 - 1011: CSR clock/10 - 1100: CSR clock/12 - 1101: CSR clock/14 - 1110: CSR clock/16 - 1111: CSR clock/18 These bits are not used for accessing RevMII. These bits are read-only if the RevMII interface is selected as single PHY interface.
7-5	RESERVED	R	0h	Reserved.
4	SKAP	R/W	0h	Skip Address Packet When this bit is set, the SMA does not send the address packets before read, write, or post-read increment address packets. This bit is valid only when C45E is set. 0h = Skip Address Packet is disabled : 0x0 1h = Skip Address Packet is enabled : 0x1

Table 43-126. MAC_MDIO_Address Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	GOC_1	R/W	0h	GMII Operation Command 1 This bit is higher bit of the operation command to the PHY or RevMII, GOC_1 and GOC_O is encoded as follows: - 00: Reserved - 01: Write - 10: Post Read Increment Address for Clause 45 PHY - 11: Read When Clause 22 PHY or RevMII is enabled, only Write and Read commands are valid. 0h = GMII Operation Command 1 is disabled : 0x0 1h = GMII Operation Command 1 is enabled : 0x1
2	GOC_0	R/W	0h	GMII Operation Command 0 This is the lower bit of the operation command to the PHY or RevMII. When in SMA mode (MDIO master) this bit along with GOC_1 determines the operation to be performed to the PHY. When only RevMII is selected in configuration this bit is read-only and tied to 1. 0h = GMII Operation Command 0 is disabled : 0x0 1h = GMII Operation Command 0 is enabled : 0x1
1	C45E	R/W	0h	Clause 45 PHY Enable When this bit is set, Clause 45 capable PHY is connected to MDIO. When this bit is reset, Clause 22 capable PHY is connected to MDIO. 0h = Clause 45 PHY is disabled : 0x0 1h = Clause 45 PHY is enabled : 0x1
0	GB	R/W	0h	GMII Busy The application sets this bit to instruct the SMA to initiate a Read or Write access to the MDIO slave. The MAC clears this bit after the MDIO frame transfer is completed. Hence the software must not write or change any of the fields in MAC_MDIO_Address and MAC_MDIO_Data registers as long as this bit is set. For write transfers, the application must first write 16-bit data in the GDI field (and also RA field when C45E is set) in MAC_MDIO_Data register before setting this bit. When C45E is set, it should also write into the RA field of MAC_MDIO_Data register before initiating a read transfer. When a read transfer is completed (GB=0), the data read from the PHY register is valid in the GD field of the MAC_MDIO_Data register. Note: Even if the addressed PHY is not present, there is no change in the functionality of this bit. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0h = GMII Busy is disabled : 0x0 1h = GMII Busy is enabled : 0x1

43.7.3.34 MAC_MDIO_Data Register (Offset = 204h) [Reset = 0h]

MAC_MDIO_Data is shown in Figure 43-73 and described in Table 43-127.

Return to the Summary Table.

The MDIO Data register stores the Write data to be written to the PHY register located at the address specified in MAC_MDIO_Address. This register also stores the Read data from the PHY register located at the address specified by MDIO Address register.

Figure 43-73. MAC_MDIO_Data Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RA																GD															
R/W-0h																R/W-0h															

Table 43-127. MAC_MDIO_Data Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RA	R/W	0h	Register Address This field is valid only when C45E is set. It contains the Register Address in the PHY to which the MDIO frame is intended for.
15-0	GD	R/W	0h	GMII Data This field contains the 16-bit data value read from the PHY or RevMII after a Management Read operation or the 16-bit data value to be written to the PHY or RevMII before a Management Write operation.

43.7.3.35 MAC_ARP_Address Register (Offset = 210h) [Reset = 0h]

MAC_ARP_Address is shown in Figure 43-74 and described in Table 43-128.

Return to the Summary Table.

The ARP Address register contains the IPv4 Destination Address of the MAC. Note: IP address should be written to this register in host byte order format.

Figure 43-74. MAC_ARP_Address Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARPPA																															
R/W-0h																															

Table 43-128. MAC_ARP_Address Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ARPPA	R/W	0h	ARP Protocol Address This field contains the IPv4 Destination Address of the MAC. This address is used for perfect match with the Protocol Address of Target field in the received ARP packet. This field is available only when the Enable IPv4 ARP Offload option is selected.

43.7.3.36 MAC_CSR_SW_Ctrl Register (Offset = 230h) [Reset = 0h]

MAC_CSR_SW_Ctrl is shown in Figure 43-75 and described in Table 43-129.

Return to the Summary Table.

This register contains SW programmable controls for changing the CSR access response and status bits clearing.

Figure 43-75. MAC_CSR_SW_Ctrl Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							RESERVED
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED							RCWE
R-0h							R/W-0h

Table 43-129. MAC_CSR_SW_Ctrl Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved.
8	RESERVED	R	0h	Reserved.
7-1	RESERVED	R	0h	Reserved.
0	RCWE	R/W	0h	Register Clear on Write 1 Enable When this bit is set, the access mode of some register fields changes to Clear on Write 1, the application needs to set that respective bit to 1 to clear it. When this bit is reset, the access mode of these register fields remain as Clear on Read. 0h = Register Clear on Write 1 is disabled : 0x0 1h = Register Clear on Write 1 is enabled : 0x1

43.7.3.37 MAC_Ext_Cfg1 Register (Offset = 238h) [Reset = 2h]

MAC_Ext_Cfg1 is shown in Figure 43-76 and described in Table 43-130.

Return to the Summary Table.

This register contains Split mode control field and offset field for Split Header feature.

Figure 43-76. MAC_Ext_Cfg1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						SPLM	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
RESERVED	SPLOFST						
R-0h	R/W-2h						

Table 43-130. MAC_Ext_Cfg1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved.
9-8	SPLM	R/W	0h	Split Mode These bits indicate the mode of splitting the incoming Rx packets. They are 0h = Split at L3/L4 header : 0x0 1h = Split at L2 header with an offset. Always Split at SPLOFST bytes from the beginning of Length/Type field of the Frame : 0x1 2h = Combination mode: Split similar to SPLM=00 for IP packets that are untagged or tagged and VLAN stripped : 0x2 3h = Reserved : 0x3
7	RESERVED	R	0h	Reserved.
6-0	SPLOFST	R/W	2h	Split Offset These bits indicate the value of offset from the beginning of Length/Type field at which header split should take place when the appropriate SPLM is selected. The reset value of this field is 2 bytes indicating a split at L2 header. Value is in terms of bytes.

43.7.3.38 MAC_Address0_High Register (Offset = 300h) [Reset = 8000FFFFh]

MAC_Address0_High is shown in Figure 43-77 and described in Table 43-131.

Return to the Summary Table.

The MAC Address0 High register holds the upper 16 bits of the first 6-byte MAC address of the station. The first DA byte that is received on the (G)MII interface corresponds to the LS byte (Bits [7:0]) of the MAC Address Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (G)MII as the destination address, then the MacAddress0 Register [47:0] is compared with 0x665544332211.
If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address0 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.

Figure 43-77. MAC_Address0_High Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	RESERVED														DCS
R-1h	R-0h														R/W-0h
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
R/W-FFFFh															

Table 43-131. MAC_Address0_High Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AE	R	1h	Address Enable This bit is always set to 1. 0h = INVALID : This bit must be always set to 1 : 0x0 1h = This bit is always set to 1 : 0x1
30-17	RESERVED	R	0h	Reserved.
16	DCS	R/W	0h	DMA Channel Select This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address(#) content is routed.
15-0	ADDRHI	R/W	FFFFh	MAC Address0[47:32] This field contains the upper 16 bits [47:32] of the first 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

43.7.3.39 MAC_Address0_Low Register (Offset = 304h) [Reset = FFFFFFFFh]

MAC_Address0_Low is shown in Figure 43-78 and described in Table 43-132.

Return to the Summary Table.

The MAC Address0 Low register holds the lower 32 bits of the 6-byte first MAC address of the station.

Figure 43-78. MAC_Address0_Low Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO																															
R/W-FFFFFFFFh																															

Table 43-132. MAC_Address0_Low Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRLO	R/W	FFFFFFFFh	MAC Address0[31:0] This field contains the lower 32 bits of the first 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets.

43.7.3.40 MAC_Address1_High Register (Offset = 308h) [Reset = FFFFh]

MAC_Address1_High is shown in Figure 43-79 and described in Table 43-133.

Return to the Summary Table.

The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.

Figure 43-79. MAC_Address1_High Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						RESERVED							DCS
R/W-0h	R/W-0h	R/W-0h						R-0h							R/W-0h
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
R/W-FFFFh															

Table 43-133. MAC_Address1_High Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AE	R/W	0h	Address Enable When this bit is set, the address filter module uses the second MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. 0h = Address is ignored : 0x0 1h = Address is enabled : 0x1
30	SA	R/W	0h	Source Address When this bit is set, the MAC Address1[47:0] is used to compare with the SA fields of the received packet. When this bit is reset, the MAC Address1[47:0] is used to compare with the DA fields of the received packet. 0h = Compare with Destination Address : 0x0 1h = Compare with Source Address : 0x1
29-24	MBC	R/W	0h	Mask Byte Control These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 registers. Each bit controls the masking of the bytes as follows: - Bit 29: Register 194[15:8] - Bit 28: Register 194[7:0] - Bit 27: Register 195[31:24] - .. - Bit 24: Register 195[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
23-17	RESERVED	R	0h	Reserved.
16	DCS	R/W	0h	DMA Channel Select This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address(#) content is routed.
15-0	ADDRHI	R/W	FFFFh	MAC Address1 [47:32] This field contains the upper 16 bits[47:32] of the second 6-byte MAC address.

43.7.3.41 MAC_Address1_Low Register (Offset = 30Ch) [Reset = FFFFFFFFh]

MAC_Address1_Low is shown in Figure 43-80 and described in Table 43-134.

Return to the Summary Table.

The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.

Figure 43-80. MAC_Address1_Low Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO																															
R/W-FFFFFFFFh																															

Table 43-134. MAC_Address1_Low Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRLO	R/W	FFFFFFFFh	MAC Address1 [31:0] This field contains the lower 32 bits of second 6-byte MAC address. The content of this field is undefined until loaded by the application after the initialization process.

43.7.3.42 MAC_Address2_High Register (Offset = 310h) [Reset = FFFFh]

MAC_Address2_High is shown in Figure 43-81 and described in Table 43-135.

Return to the Summary Table.

The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.

Figure 43-81. MAC_Address2_High Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						RESERVED							DCS
R/W-0h	R/W-0h	R/W-0h						R-0h							R/W-0h
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
R/W-FFFFh															

Table 43-135. MAC_Address2_High Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AE	R/W	0h	Address Enable When this bit is set, the address filter module uses the second MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. 0h = Address is ignored : 0x0 1h = Address is enabled : 0x1
30	SA	R/W	0h	Source Address When this bit is set, the MAC Address1[47:0] is used to compare with the SA fields of the received packet. When this bit is reset, the MAC Address1[47:0] is used to compare with the DA fields of the received packet. 0h = Compare with Destination Address : 0x0 1h = Compare with Source Address : 0x1
29-24	MBC	R/W	0h	Mask Byte Control These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 registers. Each bit controls the masking of the bytes as follows: - Bit 29: Register 194[15:8] - Bit 28: Register 194[7:0] - Bit 27: Register 195[31:24] - .. - Bit 24: Register 195[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
23-17	RESERVED	R	0h	Reserved.
16	DCS	R/W	0h	DMA Channel Select This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address(#) content is routed.
15-0	ADDRHI	R/W	FFFFh	MAC Address1 [47:32] This field contains the upper 16 bits[47:32] of the second 6-byte MAC address.

43.7.3.43 MAC_Address2_Low Register (Offset = 314h) [Reset = FFFFFFFFh]

MAC_Address2_Low is shown in Figure 43-82 and described in Table 43-136.

Return to the Summary Table.

The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.

Figure 43-82. MAC_Address2_Low Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO																															
R/W-FFFFFFFFh																															

Table 43-136. MAC_Address2_Low Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRLO	R/W	FFFFFFFFh	MAC Address1 [31:0] This field contains the lower 32 bits of second 6-byte MAC address. The content of this field is undefined until loaded by the application after the initialization process.

43.7.3.44 MAC_Address3_High Register (Offset = 318h) [Reset = FFFFh]

MAC_Address3_High is shown in Figure 43-83 and described in Table 43-137.

Return to the Summary Table.

The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.

Figure 43-83. MAC_Address3_High Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						RESERVED							DCS
R/W-0h	R/W-0h	R/W-0h						R-0h							R/W-0h
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
R/W-FFFFh															

Table 43-137. MAC_Address3_High Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AE	R/W	0h	Address Enable When this bit is set, the address filter module uses the second MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. 0h = Address is ignored : 0x0 1h = Address is enabled : 0x1
30	SA	R/W	0h	Source Address When this bit is set, the MAC Address1[47:0] is used to compare with the SA fields of the received packet. When this bit is reset, the MAC Address1[47:0] is used to compare with the DA fields of the received packet. 0h = Compare with Destination Address : 0x0 1h = Compare with Source Address : 0x1
29-24	MBC	R/W	0h	Mask Byte Control These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 registers. Each bit controls the masking of the bytes as follows: - Bit 29: Register 194[15:8] - Bit 28: Register 194[7:0] - Bit 27: Register 195[31:24] - .. - Bit 24: Register 195[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
23-17	RESERVED	R	0h	Reserved.
16	DCS	R/W	0h	DMA Channel Select This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address(#) content is routed.
15-0	ADDRHI	R/W	FFFFh	MAC Address1 [47:32] This field contains the upper 16 bits[47:32] of the second 6-byte MAC address.

43.7.3.45 MAC_Address3_Low Register (Offset = 31Ch) [Reset = FFFFFFFFh]

MAC_Address3_Low is shown in Figure 43-84 and described in Table 43-138.

Return to the Summary Table.

The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.

Figure 43-84. MAC_Address3_Low Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO																															
R/W-FFFFFFFFh																															

Table 43-138. MAC_Address3_Low Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRLO	R/W	FFFFFFFFh	MAC Address1 [31:0] This field contains the lower 32 bits of second 6-byte MAC address. The content of this field is undefined until loaded by the application after the initialization process.

43.7.3.46 MAC_Address4_High Register (Offset = 320h) [Reset = FFFFh]

MAC_Address4_High is shown in Figure 43-85 and described in Table 43-139.

Return to the Summary Table.

The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.

Figure 43-85. MAC_Address4_High Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						RESERVED							DCS
R/W-0h	R/W-0h	R/W-0h						R-0h							R/W-0h
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
R/W-FFFFh															

Table 43-139. MAC_Address4_High Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AE	R/W	0h	Address Enable When this bit is set, the address filter module uses the second MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. 0h = Address is ignored : 0x0 1h = Address is enabled : 0x1
30	SA	R/W	0h	Source Address When this bit is set, the MAC Address1[47:0] is used to compare with the SA fields of the received packet. When this bit is reset, the MAC Address1[47:0] is used to compare with the DA fields of the received packet. 0h = Compare with Destination Address : 0x0 1h = Compare with Source Address : 0x1
29-24	MBC	R/W	0h	Mask Byte Control These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 registers. Each bit controls the masking of the bytes as follows: - Bit 29: Register 194[15:8] - Bit 28: Register 194[7:0] - Bit 27: Register 195[31:24] - .. - Bit 24: Register 195[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
23-17	RESERVED	R	0h	Reserved.
16	DCS	R/W	0h	DMA Channel Select This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address(#) content is routed.
15-0	ADDRHI	R/W	FFFFh	MAC Address1 [47:32] This field contains the upper 16 bits[47:32] of the second 6-byte MAC address.

43.7.3.47 MAC_Address4_Low Register (Offset = 324h) [Reset = FFFFFFFFh]

MAC_Address4_Low is shown in Figure 43-86 and described in Table 43-140.

Return to the Summary Table.

The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.

Figure 43-86. MAC_Address4_Low Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO																															
R/W-FFFFFFFFh																															

Table 43-140. MAC_Address4_Low Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRLO	R/W	FFFFFFFFh	MAC Address1 [31:0] This field contains the lower 32 bits of second 6-byte MAC address. The content of this field is undefined until loaded by the application after the initialization process.

43.7.3.48 MAC_Address5_High Register (Offset = 328h) [Reset = FFFFh]

MAC_Address5_High is shown in Figure 43-87 and described in Table 43-141.

Return to the Summary Table.

The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.

Figure 43-87. MAC_Address5_High Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						RESERVED							DCS
R/W-0h	R/W-0h	R/W-0h						R-0h							R/W-0h
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
R/W-FFFFh															

Table 43-141. MAC_Address5_High Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AE	R/W	0h	Address Enable When this bit is set, the address filter module uses the second MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. 0h = Address is ignored : 0x0 1h = Address is enabled : 0x1
30	SA	R/W	0h	Source Address When this bit is set, the MAC Address1[47:0] is used to compare with the SA fields of the received packet. When this bit is reset, the MAC Address1[47:0] is used to compare with the DA fields of the received packet. 0h = Compare with Destination Address : 0x0 1h = Compare with Source Address : 0x1
29-24	MBC	R/W	0h	Mask Byte Control These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 registers. Each bit controls the masking of the bytes as follows: - Bit 29: Register 194[15:8] - Bit 28: Register 194[7:0] - Bit 27: Register 195[31:24] - .. - Bit 24: Register 195[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
23-17	RESERVED	R	0h	Reserved.
16	DCS	R/W	0h	DMA Channel Select This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address(#) content is routed.
15-0	ADDRHI	R/W	FFFFh	MAC Address1 [47:32] This field contains the upper 16 bits[47:32] of the second 6-byte MAC address.

43.7.3.49 MAC_Address5_Low Register (Offset = 32Ch) [Reset = FFFFFFFFh]

MAC_Address5_Low is shown in Figure 43-88 and described in Table 43-142.

Return to the Summary Table.

The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.

Figure 43-88. MAC_Address5_Low Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO																															
R/W-FFFFFFFFh																															

Table 43-142. MAC_Address5_Low Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRLO	R/W	FFFFFFFFh	MAC Address1 [31:0] This field contains the lower 32 bits of second 6-byte MAC address. The content of this field is undefined until loaded by the application after the initialization process.

43.7.3.50 MAC_Address6_High Register (Offset = 330h) [Reset = FFFFh]

MAC_Address6_High is shown in Figure 43-89 and described in Table 43-143.

Return to the Summary Table.

The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.

Figure 43-89. MAC_Address6_High Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						RESERVED							DCS
R/W-0h	R/W-0h	R/W-0h						R-0h							R/W-0h
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
R/W-FFFFh															

Table 43-143. MAC_Address6_High Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AE	R/W	0h	Address Enable When this bit is set, the address filter module uses the second MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. 0h = Address is ignored : 0x0 1h = Address is enabled : 0x1
30	SA	R/W	0h	Source Address When this bit is set, the MAC Address1[47:0] is used to compare with the SA fields of the received packet. When this bit is reset, the MAC Address1[47:0] is used to compare with the DA fields of the received packet. 0h = Compare with Destination Address : 0x0 1h = Compare with Source Address : 0x1
29-24	MBC	R/W	0h	Mask Byte Control These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 registers. Each bit controls the masking of the bytes as follows: - Bit 29: Register 194[15:8] - Bit 28: Register 194[7:0] - Bit 27: Register 195[31:24] - .. - Bit 24: Register 195[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
23-17	RESERVED	R	0h	Reserved.
16	DCS	R/W	0h	DMA Channel Select This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address(#) content is routed.
15-0	ADDRHI	R/W	FFFFh	MAC Address1 [47:32] This field contains the upper 16 bits[47:32] of the second 6-byte MAC address.

43.7.3.51 MAC_Address6_Low Register (Offset = 334h) [Reset = FFFFFFFFh]

MAC_Address6_Low is shown in Figure 43-90 and described in Table 43-144.

Return to the Summary Table.

The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.

Figure 43-90. MAC_Address6_Low Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO																															
R/W-FFFFFFFFh																															

Table 43-144. MAC_Address6_Low Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRLO	R/W	FFFFFFFFh	MAC Address1 [31:0] This field contains the lower 32 bits of second 6-byte MAC address. The content of this field is undefined until loaded by the application after the initialization process.

43.7.3.52 MAC_Address7_High Register (Offset = 338h) [Reset = FFFFh]

MAC_Address7_High is shown in Figure 43-91 and described in Table 43-145.

Return to the Summary Table.

The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.

Figure 43-91. MAC_Address7_High Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						RESERVED							DCS
R/W-0h	R/W-0h	R/W-0h						R-0h							R/W-0h
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRHI															
R/W-FFFFh															

Table 43-145. MAC_Address7_High Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AE	R/W	0h	Address Enable When this bit is set, the address filter module uses the second MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. 0h = Address is ignored : 0x0 1h = Address is enabled : 0x1
30	SA	R/W	0h	Source Address When this bit is set, the MAC Address1[47:0] is used to compare with the SA fields of the received packet. When this bit is reset, the MAC Address1[47:0] is used to compare with the DA fields of the received packet. 0h = Compare with Destination Address : 0x0 1h = Compare with Source Address : 0x1
29-24	MBC	R/W	0h	Mask Byte Control These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 registers. Each bit controls the masking of the bytes as follows: - Bit 29: Register 194[15:8] - Bit 28: Register 194[7:0] - Bit 27: Register 195[31:24] - .. - Bit 24: Register 195[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.
23-17	RESERVED	R	0h	Reserved.
16	DCS	R/W	0h	DMA Channel Select This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address(#) content is routed.
15-0	ADDRHI	R/W	FFFFh	MAC Address1 [47:32] This field contains the upper 16 bits[47:32] of the second 6-byte MAC address.

43.7.3.53 MAC_Address7_Low Register (Offset = 33Ch) [Reset = FFFFFFFFh]

MAC_Address7_Low is shown in Figure 43-92 and described in Table 43-146.

Return to the Summary Table.

The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.

Figure 43-92. MAC_Address7_Low Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRLO																															
R/W-FFFFFFFFh																															

Table 43-146. MAC_Address7_Low Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDRLO	R/W	FFFFFFFFh	MAC Address1 [31:0] This field contains the lower 32 bits of second 6-byte MAC address. The content of this field is undefined until loaded by the application after the initialization process.

43.7.3.54 MMC_Control Register (Offset = 700h) [Reset = 0h]

MMC_Control is shown in Figure 43-93 and described in Table 43-147.

Return to the Summary Table.

This register establishes the operating mode of MMC.

Figure 43-93. MMC_Control Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							UCDBC
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED		CNTPRSTLVL	CNTPRST	CNTFREEZ	RSTONRD	CNTSTOPRO	CNTRST
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 43-147. MMC_Control Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved.
8	UCDBC	R/W	0h	Update MMC Counters for Dropped Broadcast Packets Note: The CNTRST bit has a higher priority than the CNTPRST bit. Therefore, when the software tries to set both bits in the same write cycle, all counters are cleared and the CNTPRST bit is not set. When set, the MAC updates all related MMC Counters for Broadcast packets that are dropped because of the setting of the DBF bit of MAC_Packet_Filter register. When reset, the MMC Counters are not updated for dropped Broadcast packets. 0h = Update MMC Counters for Dropped Broadcast Packets is disabled : 0x0 1h = Update MMC Counters for Dropped Broadcast Packets is enabled : 0x1
7-6	RESERVED	R	0h	Reserved.
5	CNTPRSTLVL	R/W	0h	Full-Half Preset When this bit is low and the CNTPRST bit is set, all MMC counters get preset to almost-half value. All octet counters get preset to 0x7FFF_F800 (Half 2KBytes) and all packet-counters gets preset to 0x7FFF_FFF0 (Half 16). When this bit is high and the CNTPRST bit is set, all MMC counters get preset to almost-full value. All octet counters get preset to 0xFFFF_F800 (Full 2KBytes) and all packet-counters gets preset to 0xFFFF_FFF0 (Full 16). For 16-bit counters, the almost-half preset values are 0x7800 and 0x7FF0 for the respective octet and packet counters. Similarly, the almost-full preset values for the 16-bit counters are 0xF800 and 0xFFF0. 0h = Full-Half Preset is disabled : 0x0 1h = Full-Half Preset is enabled : 0x1

Table 43-147. MMC_Control Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	CNTPRST	R/W	0h	Counters Preset When this bit is set, all counters are initialized or preset to almost full or almost half according to the CNTPRSTLVL bit. This bit is cleared automatically after 1 clock cycle. This bit, along with the CNTPRSTLVL bit, is useful for debugging and testing the assertion of interrupts because of MMC counter becoming half-full or full. Access restriction applies. Self-cleared. Setting 0 clears. Setting 1 sets. 0h = Counters Preset is disabled : 0x0 1h = Counters Preset is enabled : 0x1
3	CNTFREEZ	R/W	0h	MMC Counter Freeze When this bit is set, it freezes all MMC counters to their current value. Until this bit is reset to 0, no MMC counter is updated because of any transmitted or received packet. If any MMC counter is read with the Reset on Read bit set, then that counter is also cleared in this mode. 0h = MMC Counter Freeze is disabled : 0x0 1h = MMC Counter Freeze is enabled : 0x1
2	RSTONRD	R/W	0h	Reset on Read When this bit is set, the MMC counters are reset to zero after Read (self-clearing after reset). The counters are cleared when the least significant byte lane (Bits[7:0]) is read. 0h = Reset on Read is disabled : 0x0 1h = Reset on Read is enabled : 0x1
1	CNTSTOPRO	R/W	0h	Counter Stop Rollover When this bit is set, the counter does not roll over to zero after reaching the maximum value. 0h = Counter Stop Rollover is disabled : 0x0 1h = Counter Stop Rollover is enabled : 0x1
0	CNTRST	R/W	0h	Counters Reset When this bit is set, all counters are reset. This bit is cleared automatically after 1 clock cycle. Access restriction applies. Self-cleared. Setting 0 clears. Setting 1 sets. 0h = Counters are not reset : 0x0 1h = All counters are reset : 0x1

43.7.3.55 MMC_Rx_Interrupt Register (Offset = 704h) [Reset = 0h]

MMC_Rx_Interrupt is shown in Figure 43-94 and described in Table 43-148.

Return to the Summary Table.

This register maintains the interrupts generated from all Receive statistics counters.
The MMC Receive Interrupt register maintains the interrupts that are generated when the following occur:
- Receive statistic counters reach half of their maximum values (0x8000_0000 for 32 bit counter and 0x8000 for 16 bit counter).
- Receive statistic counters cross their maximum values (0xFFFF_FFFF for 32 bit counter and 0xFFFF for 16 bit counter).

When the Counter Stop Rollover is set, interrupts are set but the counter remains at all-ones. The MMC Receive Interrupt register
is a 32 bit register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (Bits[7:0]) of the respective counter must be read to clear the interrupt bit.
Note: R_SS_RC means that this register bit is set internally, and it is cleared when the Counter register is read.

Figure 43-94. MMC_Rx_Interrupt Register

31	30	29	28	27	26	25	24
RESERVED				RXLPITRCIS	RXLPIUSCIS	RXCTRLPIS	RXRCVERRPIS
R-0h				R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
RXWDOGPIS	RXVLANGBPIS	RXFOVPIS	RXPAUSPIS	RXORANGEPIS	RXLENERPIS	RXUCGPIS	RX1024TMAXOCTGBPIS
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RX512T1023OCTGBPIS	RX256T511OCTGBPIS	RX128T255OCTGBPIS	RX65T127OCTGBPIS	RX64OCTGBPIS	RXOSIZEGPIS	RXUSIZEGPIS	RXJABERPIS
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RXRUNTPIS	RXALGNERPIS	RXCRCERPIS	RXMCGPIS	RXBCGPIS	RXGOCTIS	RXGBOCTIS	RXGBPKTIS
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 43-148. MMC_Rx_Interrupt Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved.
27	RXLPITRCIS	R	0h	MMC Receive LPI transition counter interrupt status This bit is set when the Rx_LPI_Tran_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive LPI transition Counter Interrupt Status not detected : 0x0 1h = MMC Receive LPI transition Counter Interrupt Status detected : 0x1
26	RXLPIUSCIS	R	0h	MMC Receive LPI microsecond counter interrupt status This bit is set when the Rx_LPI_USEC_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive LPI microsecond Counter Interrupt Status not detected : 0x0 1h = MMC Receive LPI microsecond Counter Interrupt Status detected : 0x1

Table 43-148. MMC_Rx_Interrupt Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	RXCTRLPIS	R	0h	MMC Receive Control Packet Counter Interrupt Status This bit is set when the rxctrlpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive Control Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive Control Packet Counter Interrupt Status detected : 0x1
24	RXRCVERRPIS	R	0h	MMC Receive Error Packet Counter Interrupt Status This bit is set when the rxrcverror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive Error Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive Error Packet Counter Interrupt Status detected : 0x1
23	RXWDOGPIS	R	0h	MMC Receive Watchdog Error Packet Counter Interrupt Status This bit is set when the rxwatchdog error counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive Watchdog Error Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive Watchdog Error Packet Counter Interrupt Status detected : 0x1
22	RXVLANGBPIS	R	0h	MMC Receive VLAN Good Bad Packet Counter Interrupt Status This bit is set when the rxvlanpackets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive VLAN Good Bad Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive VLAN Good Bad Packet Counter Interrupt Status detected : 0x1
21	RXFOVPIS	R	0h	MMC Receive FIFO Overflow Packet Counter Interrupt Status This bit is set when the rxfifooverflow counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive FIFO Overflow Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive FIFO Overflow Packet Counter Interrupt Status detected : 0x1
20	RXPAUSPIS	R	0h	MMC Receive Pause Packet Counter Interrupt Status This bit is set when the rxpausepackets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive Pause Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive Pause Packet Counter Interrupt Status detected : 0x1
19	RXORANGEPIS	R	0h	MMC Receive Out Of Range Error Packet Counter Interrupt Status. This bit is set when the rxoutofrangetype counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive Out Of Range Error Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive Out Of Range Error Packet Counter Interrupt Status detected : 0x1

Table 43-148. MMC_Rx_Interrupt Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	RXLENERPIS	R	0h	MMC Receive Length Error Packet Counter Interrupt Status This bit is set when the rxlengtherror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive Length Error Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive Length Error Packet Counter Interrupt Status detected : 0x1
17	RXUCGPIS	R	0h	MMC Receive Unicast Good Packet Counter Interrupt Status This bit is set when the rxunicastpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive Unicast Good Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive Unicast Good Packet Counter Interrupt Status detected : 0x1
16	RX1024TMAXOCTGBPIS	R	0h	MMC Receive 1024 to Maximum Octet Good Bad Packet Counter Interrupt Status This bit is set when the rx1024tomaxoctets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive 1024 to Maximum Octet Good Bad Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive 1024 to Maximum Octet Good Bad Packet Counter Interrupt Status detected : 0x1
15	RX512T1023OCTGBPIS	R	0h	MMC Receive 512 to 1023 Octet Good Bad Packet Counter Interrupt Status This bit is set when the rx512to1023octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive 512 to 1023 Octet Good Bad Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive 512 to 1023 Octet Good Bad Packet Counter Interrupt Status detected : 0x1
14	RX256T511OCTGBPIS	R	0h	MMC Receive 256 to 511 Octet Good Bad Packet Counter Interrupt Status This bit is set when the rx256to511octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive 256 to 511 Octet Good Bad Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive 256 to 511 Octet Good Bad Packet Counter Interrupt Status detected : 0x1
13	RX128T255OCTGBPIS	R	0h	MMC Receive 128 to 255 Octet Good Bad Packet Counter Interrupt Status This bit is set when the rx128to255octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive 128 to 255 Octet Good Bad Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive 128 to 255 Octet Good Bad Packet Counter Interrupt Status detected : 0x1

Table 43-148. MMC_Rx_Interrupt Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	RX65T127OCTGBPIS	R	0h	MMC Receive 65 to 127 Octet Good Bad Packet Counter Interrupt Status This bit is set when the rx65to127octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive 65 to 127 Octet Good Bad Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive 65 to 127 Octet Good Bad Packet Counter Interrupt Status detected : 0x1
11	RX64OCTGBPIS	R	0h	MMC Receive 64 Octet Good Bad Packet Counter Interrupt Status This bit is set when the rx64octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive 64 Octet Good Bad Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive 64 Octet Good Bad Packet Counter Interrupt Status detected : 0x1
10	RXOSIZEGPIS	R	0h	MMC Receive Oversize Good Packet Counter Interrupt Status This bit is set when the rxoversize_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive Oversize Good Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive Oversize Good Packet Counter Interrupt Status detected : 0x1
9	RXUSIZEGPIS	R	0h	MMC Receive Undersize Good Packet Counter Interrupt Status This bit is set when the rxundersize_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive Undersize Good Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive Undersize Good Packet Counter Interrupt Status detected : 0x1
8	RXJABERPIS	R	0h	MMC Receive Jabber Error Packet Counter Interrupt Status This bit is set when the rxjabbererror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive Jabber Error Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive Jabber Error Packet Counter Interrupt Status detected : 0x1
7	RXRUNTPIS	R	0h	MMC Receive Runt Packet Counter Interrupt Status This bit is set when the rxrunterror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive Runt Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive Runt Packet Counter Interrupt Status detected : 0x1

Table 43-148. MMC_Rx_Interrupt Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	RXALGNERPIS	R	0h	MMC Receive Alignment Error Packet Counter Interrupt Status This bit is set when the rxalignmenterror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive Alignment Error Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive Alignment Error Packet Counter Interrupt Status detected : 0x1
5	RXCRCERPIS	R	0h	MMC Receive CRC Error Packet Counter Interrupt Status This bit is set when the rxcrcerror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive CRC Error Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive CRC Error Packet Counter Interrupt Status detected : 0x1
4	RXMCGPIS	R	0h	MMC Receive Multicast Good Packet Counter Interrupt Status This bit is set when the rxmulticastpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive Multicast Good Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive Multicast Good Packet Counter Interrupt Status detected : 0x1
3	RXBCGPIS	R	0h	MMC Receive Broadcast Good Packet Counter Interrupt Status This bit is set when the rxbroadcastpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive Broadcast Good Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive Broadcast Good Packet Counter Interrupt Status detected : 0x1
2	RXGOCTIS	R	0h	MMC Receive Good Octet Counter Interrupt Status This bit is set when the rxoctetcount_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive Good Octet Counter Interrupt Status not detected : 0x0 1h = MMC Receive Good Octet Counter Interrupt Status detected : 0x1
1	RXGBOCTIS	R	0h	MMC Receive Good Bad Octet Counter Interrupt Status This bit is set when the rxoctetcount_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive Good Bad Octet Counter Interrupt Status not detected : 0x0 1h = MMC Receive Good Bad Octet Counter Interrupt Status detected : 0x1

Table 43-148. MMC_Rx_Interrupt Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	RXGBPKTIS	R	0h	MMC Receive Good Bad Packet Counter Interrupt Status This bit is set when the rxpacketcount_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive Good Bad Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive Good Bad Packet Counter Interrupt Status detected : 0x1

43.7.3.56 MMC_Tx_Interrupt Register (Offset = 708h) [Reset = 0h]

MMC_Tx_Interrupt is shown in Figure 43-95 and described in Table 43-149.

Return to the Summary Table.

This register maintains the interrupts generated from all Transmit statistics counters.
The MMC Transmit Interrupt register maintains the interrupts generated when transmit statistic counters reach half their maximum values
(0x8000_0000 for 32 bit counter and 0x8000 for 16 bit counter), and when they cross their maximum values (0xFFFF_FFFF for 32-bit counter and 0xFFFF for 16-bit counter).
When Counter Stop Rollover is set, the interrupts are set but the counter remains at all-ones.
The MMC Transmit Interrupt register is a 32 bit register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read.
The least significant byte lane (Bits[7:0]) of the respective counter must be read to clear the interrupt bit.

Figure 43-95. MMC_Tx_Interrupt Register

31	30	29	28	27	26	25	24
RESERVED				TXLPITRCIS	TXLPIUSCIS	TXOSIZEGPIS	TXVLANGPIS
R-0h				R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
TXPAUSPIS	TXEXDEFPIS	TXGPKTIS	TXGOCTIS	TXCARERPIS	TXEXCOLPIS	TXLATCOLPIS	TXDEFPIS
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
TXMCOLGPIS	TXSCOLGPIS	TXUFLOWERPIS	TXBCGBPIS	TXMCGBPIS	TXUCGBPIS	TX1024TMAXOCTGBPIS	TX512T1023OCTGBPIS
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
TX256T511OCTGBPIS	TX128T255OCTGBPIS	TX65T127OCTGBPIS	TX64OCTGBPIS	TXMCGPIS	TXBCGPIS	TXGBPKTIS	TXGBOCTIS
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 43-149. MMC_Tx_Interrupt Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved.
27	TXLPITRCIS	R	0h	MMC Transmit LPI transition counter interrupt status This bit is set when the Tx_LPI_Tran_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit LPI transition Counter Interrupt Status not detected : 0x0 1h = MMC Transmit LPI transition Counter Interrupt Status detected : 0x1
26	TXLPIUSCIS	R	0h	MMC Transmit LPI microsecond counter interrupt status This bit is set when the Tx_LPI_USEC_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit LPI microsecond Counter Interrupt Status not detected : 0x0 1h = MMC Transmit LPI microsecond Counter Interrupt Status detected : 0x1

Table 43-149. MMC_Tx_Interrupt Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	TXOSIZEGPIS	R	0h	MMC Transmit Oversize Good Packet Counter Interrupt Status This bit is set when the txoversize_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Oversize Good Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Oversize Good Packet Counter Interrupt Status detected : 0x1
24	TXVLANGPIS	R	0h	MMC Transmit VLAN Good Packet Counter Interrupt Status This bit is set when the txvlanpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit VLAN Good Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit VLAN Good Packet Counter Interrupt Status detected : 0x1
23	TXPAUSPIS	R	0h	MMC Transmit Pause Packet Counter Interrupt Status This bit is set when the txpausepacketserror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Pause Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Pause Packet Counter Interrupt Status detected : 0x1
22	TXEXDEFPIS	R	0h	MMC Transmit Excessive Deferral Packet Counter Interrupt Status This bit is set when the txexcessdef counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Excessive Deferral Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Excessive Deferral Packet Counter Interrupt Status detected : 0x1
21	TXGPKTIS	R	0h	MMC Transmit Good Packet Counter Interrupt Status This bit is set when the txpacketcount_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Good Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Good Packet Counter Interrupt Status detected : 0x1
20	TXGOCTIS	R	0h	MMC Transmit Good Octet Counter Interrupt Status This bit is set when the txoctetcount_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Good Octet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Good Octet Counter Interrupt Status detected : 0x1
19	TXCARERPIS	R	0h	MMC Transmit Carrier Error Packet Counter Interrupt Status This bit is set when the txcarriererror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Carrier Error Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Carrier Error Packet Counter Interrupt Status detected : 0x1

Table 43-149. MMC_Tx_Interrupt Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	TXEXCOLPIS	R	0h	MMC Transmit Excessive Collision Packet Counter Interrupt Status This bit is set when the txexesscol counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Excessive Collision Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Excessive Collision Packet Counter Interrupt Status detected : 0x1
17	TXLATCOLPIS	R	0h	MMC Transmit Late Collision Packet Counter Interrupt Status This bit is set when the txlatecol counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Late Collision Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Late Collision Packet Counter Interrupt Status detected : 0x1
16	TXDEFPIS	R	0h	MMC Transmit Deferred Packet Counter Interrupt Status This bit is set when the txdeferred counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Deferred Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Deferred Packet Counter Interrupt Status detected : 0x1
15	TXMCOLGPIS	R	0h	MMC Transmit Multiple Collision Good Packet Counter Interrupt Status This bit is set when the txmulticol_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Multiple Collision Good Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Multiple Collision Good Packet Counter Interrupt Status detected : 0x1
14	TXSCOLGPIS	R	0h	MMC Transmit Single Collision Good Packet Counter Interrupt Status This bit is set when the txsinglecol_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Single Collision Good Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Single Collision Good Packet Counter Interrupt Status detected : 0x1
13	TXUFLOWERPIS	R	0h	MMC Transmit Underflow Error Packet Counter Interrupt Status This bit is set when the txunderflowerror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Underflow Error Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Underflow Error Packet Counter Interrupt Status detected : 0x1

Table 43-149. MMC_Tx_Interrupt Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	TXBCGBPIS	R	0h	MMC Transmit Broadcast Good Bad Packet Counter Interrupt Status This bit is set when the txbroadcastpackets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Broadcast Good Bad Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Broadcast Good Bad Packet Counter Interrupt Status detected : 0x1
11	TXMCGBPIS	R	0h	MMC Transmit Multicast Good Bad Packet Counter Interrupt Status The bit is set when the txmulticastpackets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Multicast Good Bad Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Multicast Good Bad Packet Counter Interrupt Status detected : 0x1
10	TXUCGBPIS	R	0h	MMC Transmit Unicast Good Bad Packet Counter Interrupt Status This bit is set when the txunicastpackets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Unicast Good Bad Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Unicast Good Bad Packet Counter Interrupt Status detected : 0x1
9	TX1024TMAXOCTGBPIS	R	0h	MMC Transmit 1024 to Maximum Octet Good Bad Packet Counter Interrupt Status This bit is set when the tx1024tomaxoctets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit 1024 to Maximum Octet Good Bad Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit 1024 to Maximum Octet Good Bad Packet Counter Interrupt Status detected : 0x1
8	TX512T1023OCTGBPIS	R	0h	MMC Transmit 512 to 1023 Octet Good Bad Packet Counter Interrupt Status This bit is set when the tx512to1023octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit 512 to 1023 Octet Good Bad Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit 512 to 1023 Octet Good Bad Packet Counter Interrupt Status detected : 0x1
7	TX256T511OCTGBPIS	R	0h	MMC Transmit 256 to 511 Octet Good Bad Packet Counter Interrupt Status This bit is set when the tx256to511octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit 256 to 511 Octet Good Bad Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit 256 to 511 Octet Good Bad Packet Counter Interrupt Status detected : 0x1

Table 43-149. MMC_Tx_Interrupt Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	TX128T255OCTGBPIS	R	0h	MMC Transmit 128 to 255 Octet Good Bad Packet Counter Interrupt Status This bit is set when the tx128to255octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit 128 to 255 Octet Good Bad Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit 128 to 255 Octet Good Bad Packet Counter Interrupt Status detected : 0x1
5	TX65T127OCTGBPIS	R	0h	MMC Transmit 65 to 127 Octet Good Bad Packet Counter Interrupt Status This bit is set when the tx65to127octets_gb counter reaches half the maximum value, and also when it reaches the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit 65 to 127 Octet Good Bad Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit 65 to 127 Octet Good Bad Packet Counter Interrupt Status detected : 0x1
4	TX64OCTGBPIS	R	0h	MMC Transmit 64 Octet Good Bad Packet Counter Interrupt Status This bit is set when the tx64octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit 64 Octet Good Bad Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit 64 Octet Good Bad Packet Counter Interrupt Status detected : 0x1
3	TXMCGPIS	R	0h	MMC Transmit Multicast Good Packet Counter Interrupt Status This bit is set when the txmulticastpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Multicast Good Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Multicast Good Packet Counter Interrupt Status detected : 0x1
2	TXBCGPIS	R	0h	MMC Transmit Broadcast Good Packet Counter Interrupt Status This bit is set when the txbroadcastpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Broadcast Good Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Broadcast Good Packet Counter Interrupt Status detected : 0x1
1	TXGBPKTIS	R	0h	MMC Transmit Good Bad Packet Counter Interrupt Status This bit is set when the txpacketcount_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Good Bad Packet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Good Bad Packet Counter Interrupt Status detected : 0x1

Table 43-149. MMC_Tx_Interrupt Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	TXGBOCTIS	R	0h	MMC Transmit Good Bad Octet Counter Interrupt Status This bit is set when the txoctetcount_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Transmit Good Bad Octet Counter Interrupt Status not detected : 0x0 1h = MMC Transmit Good Bad Octet Counter Interrupt Status detected : 0x1

43.7.3.57 MMC_Rx_Interrupt_Mask Register (Offset = 70Ch) [Reset = 0h]

MMC_Rx_Interrupt_Mask is shown in Figure 43-96 and described in Table 43-150.

Return to the Summary Table.

This register maintains the masks for interrupts generated from all Receive statistics counters.
The MMC Receive Interrupt Mask register maintains the masks for the interrupts generated when receive statistic counters reach half of their maximum value or the maximum values.
This register is 32 bit wide.

Figure 43-96. MMC_Rx_Interrupt_Mask Register

31	30	29	28	27	26	25	24
RESERVED				RXLPITRCIM	RXLPIUSCIM	RXCTRLPIM	RXRCVERRPIM
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RXWDOGPIM	RXVLANGBPIM	RXFOVPIM	RXPAUSPIM	RXORANGEPIM	RXLENERPIM	RXUCGPIM	RX1024TMAXOCTGBPIM
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RX512T1023OCTGBPIM	RX256T511OCTGBPIM	RX128T255OCTGBPIM	RX65T127OCTGBPIM	RX64OCTGBPIM	RXOSIZEGPIM	RXUSIZEGPIM	RXJABERPIM
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RXRUNTPIM	RXALGNERPIM	RXCRCERPIM	RXMCGPIM	RXBCGPIM	RXGOCTIM	RXGBOCTIM	RXGBPKTIM
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 43-150. MMC_Rx_Interrupt_Mask Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved.
27	RXLPITRCIM	R/W	0h	MMC Receive LPI transition counter interrupt Mask Setting this bit masks the interrupt when the Rx_LPI_Tran_Cntr counter reaches half of the maximum value or the maximum value. 0h = MMC Receive LPI transition counter interrupt Mask is disabled : 0x0 1h = MMC Receive LPI transition counter interrupt Mask is enabled : 0x1
26	RXLPIUSCIM	R/W	0h	MMC Receive LPI microsecond counter interrupt Mask Setting this bit masks the interrupt when the Rx_LPI_USEC_Cntr counter reaches half of the maximum value or the maximum value. 0h = MMC Receive LPI microsecond counter interrupt Mask is disabled : 0x0 1h = MMC Receive LPI microsecond counter interrupt Mask is enabled : 0x1
25	RXCTRLPIM	R/W	0h	MMC Receive Control Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxctrlpackets_g counter reaches half of the maximum value or the maximum value. 0h = MMC Receive Control Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive Control Packet Counter Interrupt Mask is enabled : 0x1

Table 43-150. MMC_Rx_Interrupt_Mask Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24	RXRCVERRPIM	R/W	0h	MMC Receive Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxrcverror counter reaches half of the maximum value or the maximum value. 0h = MMC Receive Error Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive Error Packet Counter Interrupt Mask is enabled : 0x1
23	RXWDOGPIM	R/W	0h	MMC Receive Watchdog Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxwatchdog counter reaches half of the maximum value or the maximum value. 0h = MMC Receive Watchdog Error Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive Watchdog Error Packet Counter Interrupt Mask is enabled : 0x1
22	RXVLANGBPIM	R/W	0h	MMC Receive VLAN Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxvlanpackets_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Receive VLAN Good Bad Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive VLAN Good Bad Packet Counter Interrupt Mask is enabled : 0x1
21	RXFOVPIM	R/W	0h	MMC Receive FIFO Overflow Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxfifooverflow counter reaches half of the maximum value or the maximum value. 0h = MMC Receive FIFO Overflow Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive FIFO Overflow Packet Counter Interrupt Mask is enabled : 0x1
20	RXPAUSPIM	R/W	0h	MMC Receive Pause Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxpausepackets counter reaches half of the maximum value or the maximum value. 0h = MMC Receive Pause Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive Pause Packet Counter Interrupt Mask is enabled : 0x1
19	RXORANGEPIM	R/W	0h	MMC Receive Out Of Range Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxoutofrangetype counter reaches half of the maximum value or the maximum value. 0h = MMC Receive Out Of Range Error Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive Out Of Range Error Packet Counter Interrupt Mask is enabled : 0x1
18	RXLENERPIM	R/W	0h	MMC Receive Length Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxlengtherror counter reaches half of the maximum value or the maximum value. 0h = MMC Receive Length Error Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive Length Error Packet Counter Interrupt Mask is enabled : 0x1
17	RXUCGPIM	R/W	0h	MMC Receive Unicast Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxunicastpackets_g counter reaches half of the maximum value or the maximum value. 0h = MMC Receive Unicast Good Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive Unicast Good Packet Counter Interrupt Mask is enabled : 0x1

Table 43-150. MMC_Rx_Interrupt_Mask Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	RX1024TMAXOCTGBPIM	R/W	0h	MMC Receive 1024 to Maximum Octet Good Bad Packet Counter Interrupt Mask. Setting this bit masks the interrupt when the rx1024tomaxoctets_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Receive 1024 to Maximum Octet Good Bad Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive 1024 to Maximum Octet Good Bad Packet Counter Interrupt Mask is enabled : 0x1
15	RX512T1023OCTGBPIM	R/W	0h	MMC Receive 512 to 1023 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rx512to1023octets_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Receive 512 to 1023 Octet Good Bad Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive 512 to 1023 Octet Good Bad Packet Counter Interrupt Mask is enabled : 0x1
14	RX256T511OCTGBPIM	R/W	0h	MMC Receive 256 to 511 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rx256to511octets_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Receive 256 to 511 Octet Good Bad Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive 256 to 511 Octet Good Bad Packet Counter Interrupt Mask is enabled : 0x1
13	RX128T255OCTGBPIM	R/W	0h	MMC Receive 128 to 255 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rx128to255octets_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Receive 128 to 255 Octet Good Bad Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive 128 to 255 Octet Good Bad Packet Counter Interrupt Mask is enabled : 0x1
12	RX65T127OCTGBPIM	R/W	0h	MMC Receive 65 to 127 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rx65to127octets_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Receive 65 to 127 Octet Good Bad Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive 65 to 127 Octet Good Bad Packet Counter Interrupt Mask is enabled : 0x1
11	RX64OCTGBPIM	R/W	0h	MMC Receive 64 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rx64octets_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Receive 64 Octet Good Bad Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive 64 Octet Good Bad Packet Counter Interrupt Mask is enabled : 0x1
10	RXOSIZEGPIM	R/W	0h	MMC Receive Oversize Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxoversize_g counter reaches half of the maximum value or the maximum value. 0h = MMC Receive Oversize Good Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive Oversize Good Packet Counter Interrupt Mask is enabled : 0x1
9	RXUSIZEGPIM	R/W	0h	MMC Receive Undersize Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxundersize_g counter reaches half of the maximum value or the maximum value. 0h = MMC Receive Undersize Good Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive Undersize Good Packet Counter Interrupt Mask is enabled : 0x1

Table 43-150. MMC_Rx_Interrupt_Mask Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	RXJABERPIM	R/W	0h	MMC Receive Jabber Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxjabbererror counter reaches half of the maximum value or the maximum value. 0h = MMC Receive Jabber Error Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive Jabber Error Packet Counter Interrupt Mask is enabled : 0x1
7	RXRUNTPIM	R/W	0h	MMC Receive Runt Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxrunterror counter reaches half of the maximum value or the maximum value. 0h = MMC Receive Runt Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive Runt Packet Counter Interrupt Mask is enabled : 0x1
6	RXALGNERPIM	R/W	0h	MMC Receive Alignment Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxalignmenterror counter reaches half of the maximum value or the maximum value. 0h = MMC Receive Alignment Error Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive Alignment Error Packet Counter Interrupt Mask is enabled : 0x1
5	RXCRCERPIM	R/W	0h	MMC Receive CRC Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxcrcerror counter reaches half of the maximum value or the maximum value. 0h = MMC Receive CRC Error Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive CRC Error Packet Counter Interrupt Mask is enabled : 0x1
4	RXMCGPIM	R/W	0h	MMC Receive Multicast Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxmulticastpackets_g counter reaches half of the maximum value or the maximum value. 0h = MMC Receive Multicast Good Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive Multicast Good Packet Counter Interrupt Mask is enabled : 0x1
3	RXBCGPIM	R/W	0h	MMC Receive Broadcast Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxbroadcastpackets_g counter reaches half of the maximum value or the maximum value. 0h = MMC Receive Broadcast Good Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive Broadcast Good Packet Counter Interrupt Mask is enabled : 0x1
2	RXGOCTIM	R/W	0h	MMC Receive Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxoctetcount_g counter reaches half of the maximum value or the maximum value. 0h = MMC Receive Good Octet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive Good Octet Counter Interrupt Mask is enabled : 0x1
1	RXGBOCTIM	R/W	0h	MMC Receive Good Bad Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxoctetcount_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Receive Good Bad Octet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive Good Bad Octet Counter Interrupt Mask is enabled : 0x1

Table 43-150. MMC_Rx_Interrupt_Mask Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	RXGBPKTIM	R/W	0h	MMC Receive Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxpacketcount_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Receive Good Bad Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive Good Bad Packet Counter Interrupt Mask is enabled : 0x1

43.7.3.58 MMC_Tx_Interrupt_Mask Register (Offset = 710h) [Reset = 0h]

MMC_Tx_Interrupt_Mask is shown in Figure 43-97 and described in Table 43-151.

Return to the Summary Table.

This register maintains the masks for interrupts generated from all Transmit statistics counters.
The MMC Transmit Interrupt Mask register maintains the masks for the interrupts generated when the transmit statistic counters reach half of their maximum value or the maximum values. This register is 32 bit wide.

Figure 43-97. MMC_Tx_Interrupt_Mask Register

31	30	29	28	27	26	25	24
RESERVED				TXLPITRCIM	TXLPIUSCIM	TXOSIZEGPIM	TXVLANGPIM
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
TXPAUSPIM	TXEXDEFPIM	TXGPKTIM	TXGOCTIM	TXCARERPIM	TXEXCOLPIM	TXLATCOLPIM	TXDEFPIM
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
TXMCOLGPIM	TXSCOLGPIM	TXUFLOWERPIM	TXBCGBPIM	TXMCGBPIM	TXUCGBPIM	TX1024TMAXOCTGBPIM	TX512T1023OCTGBPIM
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TX256T511OCTGBPIM	TX128T255OCTGBPIM	TX65T127OCTGBPIM	TX64OCTGBPIM	TXMCGPIM	TXBCGPIM	TXGBPKTIM	TXGBOCTIM
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 43-151. MMC_Tx_Interrupt_Mask Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved.
27	TXLPITRCIM	R/W	0h	MMC Transmit LPI transition counter interrupt Mask Setting this bit masks the interrupt when the Tx_LPI_Tran_Cntr counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit LPI transition counter interrupt Mask is disabled : 0x0 1h = MMC Transmit LPI transition counter interrupt Mask is enabled : 0x1
26	TXLPIUSCIM	R/W	0h	MMC Transmit LPI microsecond counter interrupt Mask Setting this bit masks the interrupt when the Tx_LPI_USEC_Cntr counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit LPI microsecond counter interrupt Mask is disabled : 0x0 1h = MMC Transmit LPI microsecond counter interrupt Mask is enabled : 0x1
25	TXOSIZEGPIM	R/W	0h	MMC Transmit Oversize Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txoversize_g counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Oversize Good Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Oversize Good Packet Counter Interrupt Mask is enabled : 0x1
24	TXVLANGPIM	R/W	0h	MMC Transmit VLAN Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txvlanpackets_g counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit VLAN Good Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit VLAN Good Packet Counter Interrupt Mask is enabled : 0x1

Table 43-151. MMC_Tx_Interrupt_Mask Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	TXPAUSPIM	R/W	0h	MMC Transmit Pause Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txpausepackets counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Pause Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Pause Packet Counter Interrupt Mask is enabled : 0x1
22	TXEXDEFPIM	R/W	0h	MMC Transmit Excessive Deferral Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txexcessdef counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Excessive Deferral Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Excessive Deferral Packet Counter Interrupt Mask is enabled : 0x1
21	TXGPKTIM	R/W	0h	MMC Transmit Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txpacketcount_g counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Good Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Good Packet Counter Interrupt Mask is enabled : 0x1
20	TXGOCTIM	R/W	0h	MMC Transmit Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the txoctetcount_g counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Good Octet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Good Octet Counter Interrupt Mask is enabled : 0x1
19	TXCARERPIM	R/W	0h	MMC Transmit Carrier Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txcarriererror counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Carrier Error Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Carrier Error Packet Counter Interrupt Mask is enabled : 0x1
18	TXEXCOLPIM	R/W	0h	MMC Transmit Excessive Collision Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txexcesscol counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Excessive Collision Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Excessive Collision Packet Counter Interrupt Mask is enabled : 0x1
17	TXLATCOLPIM	R/W	0h	MMC Transmit Late Collision Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txlatecol counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Late Collision Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Late Collision Packet Counter Interrupt Mask is enabled : 0x1
16	TXDEFPIM	R/W	0h	MMC Transmit Deferred Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txdeferred counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Deferred Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Deferred Packet Counter Interrupt Mask is enabled : 0x1

Table 43-151. MMC_Tx_Interrupt_Mask Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15	TXMCOLGPIM	R/W	0h	MMC Transmit Multiple Collision Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txmulticol_g counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Multiple Collision Good Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Multiple Collision Good Packet Counter Interrupt Mask is enabled : 0x1
14	TXSCOLGPIM	R/W	0h	MMC Transmit Single Collision Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txsinglecol_g counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Single Collision Good Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Single Collision Good Packet Counter Interrupt Mask is enabled : 0x1
13	TXUFLOWERPIM	R/W	0h	MMC Transmit Underflow Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txunderflowerror counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Underflow Error Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Underflow Error Packet Counter Interrupt Mask is enabled : 0x1
12	TXBCGBPIM	R/W	0h	MMC Transmit Broadcast Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txbroadcastpackets_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Broadcast Good Bad Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Broadcast Good Bad Packet Counter Interrupt Mask is enabled : 0x1
11	TXMCGBPIM	R/W	0h	MMC Transmit Multicast Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txmulticastpackets_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Multicast Good Bad Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Multicast Good Bad Packet Counter Interrupt Mask is enabled : 0x1
10	TXUCGBPIM	R/W	0h	MMC Transmit Unicast Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txunicastpackets_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Unicast Good Bad Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Unicast Good Bad Packet Counter Interrupt Mask is enabled : 0x1
9	TX1024TMAXOCTGBPIM	R/W	0h	MMC Transmit 1024 to Maximum Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the tx1024tomaxoctets_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit 1024 to Maximum Octet Good Bad Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit 1024 to Maximum Octet Good Bad Packet Counter Interrupt Mask is enabled : 0x1
8	TX512T1023OCTGBPIM	R/W	0h	MMC Transmit 512 to 1023 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the tx512to1023octets_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit 512 to 1023 Octet Good Bad Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit 512 to 1023 Octet Good Bad Packet Counter Interrupt Mask is enabled : 0x1

Table 43-151. MMC_Tx_Interrupt_Mask Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	TX256T511OCTGBPIM	R/W	0h	MMC Transmit 256 to 511 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the tx256to511octets_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit 256 to 511 Octet Good Bad Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit 256 to 511 Octet Good Bad Packet Counter Interrupt Mask is enabled : 0x1
6	TX128T255OCTGBPIM	R/W	0h	MMC Transmit 128 to 255 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the tx128to255octets_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit 128 to 255 Octet Good Bad Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit 128 to 255 Octet Good Bad Packet Counter Interrupt Mask is enabled : 0x1
5	TX65T127OCTGBPIM	R/W	0h	MMC Transmit 65 to 127 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the tx65to127octets_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit 65 to 127 Octet Good Bad Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit 65 to 127 Octet Good Bad Packet Counter Interrupt Mask is enabled : 0x1
4	TX64OCTGBPIM	R/W	0h	MMC Transmit 64 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the tx64octets_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit 64 Octet Good Bad Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit 64 Octet Good Bad Packet Counter Interrupt Mask is enabled : 0x1
3	TXMCGPIM	R/W	0h	MMC Transmit Multicast Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txmulticastpackets_g counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Multicast Good Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Multicast Good Packet Counter Interrupt Mask is enabled : 0x1
2	TXBCGPIM	R/W	0h	MMC Transmit Broadcast Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txbroadcastpackets_g counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Broadcast Good Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Broadcast Good Packet Counter Interrupt Mask is enabled : 0x1
1	TXGBPKTIM	R/W	0h	MMC Transmit Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txpacketcount_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Good Bad Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Good Bad Packet Counter Interrupt Mask is enabled : 0x1
0	TXGBOCTIM	R/W	0h	MMC Transmit Good Bad Octet Counter Interrupt Mask Setting this bit masks the interrupt when the txoctetcount_gb counter reaches half of the maximum value or the maximum value. 0h = MMC Transmit Good Bad Octet Counter Interrupt Mask is disabled : 0x0 1h = MMC Transmit Good Bad Octet Counter Interrupt Mask is enabled : 0x1

43.7.3.59 Tx_Octet_Count_Good_Bad Register (Offset = 714h) [Reset = 0h]

Tx_Octet_Count_Good_Bad is shown in Figure 43-98 and described in Table 43-152.

Return to the Summary Table.

This register provides the number of bytes transmitted by the DWC_ether_qos, exclusive of preamble and retried bytes, in good and bad packets.

Figure 43-98. Tx_Octet_Count_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXOCTGB																															
R-0h																															

Table 43-152. Tx_Octet_Count_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXOCTGB	R	0h	Tx Octet Count Good Bad This field indicates the number of bytes transmitted, exclusive of preamble and retried bytes, in good and bad packets.

43.7.3.60 Tx_Packet_Count_Good_Bad Register (Offset = 718h) [Reset = 0h]

Tx_Packet_Count_Good_Bad is shown in Figure 43-99 and described in Table 43-153.

Return to the Summary Table.

This register provides the number of good and bad packets transmitted by DWC_ether_qos, exclusive of retried packets.

Figure 43-99. Tx_Packet_Count_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXPKTGB																															
R-0h																															

Table 43-153. Tx_Packet_Count_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXPKTGB	R	0h	Tx Packet Count Good Bad This field indicates the number of good and bad packets transmitted, exclusive of retried packets.

43.7.3.61 Tx_Broadcast_Packets_Good Register (Offset = 71Ch) [Reset = 0h]

Tx_Broadcast_Packets_Good is shown in Figure 43-100 and described in Table 43-154.

Return to the Summary Table.

This register provides the number of good broadcast packets transmitted by DWC_ether_qos.

Figure 43-100. Tx_Broadcast_Packets_Good Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXBCASTG																															
R-0h																															

Table 43-154. Tx_Broadcast_Packets_Good Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXBCASTG	R	0h	Tx Broadcast Packets Good This field indicates the number of good broadcast packets transmitted.

43.7.3.62 Tx_Multicast_Packets_Good Register (Offset = 720h) [Reset = 0h]

Tx_Multicast_Packets_Good is shown in Figure 43-101 and described in Table 43-155.

Return to the Summary Table.

This register provides the number of good multicast packets transmitted by DWC_ether_qos.

Figure 43-101. Tx_Multicast_Packets_Good Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXMCASTG																															
R-0h																															

Table 43-155. Tx_Multicast_Packets_Good Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXMCASTG	R	0h	Tx Multicast Packets Good This field indicates the number of good multicast packets transmitted.

43.7.3.63 Tx_64Octets_Packets_Good_Bad Register (Offset = 724h) [Reset = 0h]

Tx_64Octets_Packets_Good_Bad is shown in Figure 43-102 and described in Table 43-156.

Return to the Summary Table.

This register provides the number of good and bad packets transmitted by DWC_ether_qos with length 64 bytes, exclusive of preamble and retried packets.

Figure 43-102. Tx_64Octets_Packets_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX64OCTGB																															
R-0h																															

Table 43-156. Tx_64Octets_Packets_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TX64OCTGB	R	0h	Tx 64Octets Packets Good_Bad This field indicates the number of good and bad packets transmitted with length 64 bytes, exclusive of preamble and retried packets.

43.7.3.64 Tx_65To127Octets_Packets_Good_Bad Register (Offset = 728h) [Reset = 0h]

Tx_65To127Octets_Packets_Good_Bad is shown in Figure 43-103 and described in Table 43-157.

Return to the Summary Table.

This register provides the number of good and bad packets transmitted by DWC_ether_qos with length between 65 and 127 (inclusive) bytes, exclusive of preamble and retried packets.

Figure 43-103. Tx_65To127Octets_Packets_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX65_127OCTGB																															
R-0h																															

Table 43-157. Tx_65To127Octets_Packets_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TX65_127OCTGB	R	0h	Tx 65To127Octets Packets Good Bad This field indicates the number of good and bad packets transmitted with length between 65 and 127 (inclusive) bytes, exclusive of preamble and retried packets.

43.7.3.65 Tx_128To255Octets_Packets_Good_Bad Register (Offset = 72Ch) [Reset = 0h]

Tx_128To255Octets_Packets_Good_Bad is shown in Figure 43-104 and described in Table 43-158.

Return to the Summary Table.

This register provides the number of good and bad packets transmitted by DWC_ether_qos with length between 128 to 255 (inclusive) bytes, exclusive of preamble and retried packets.

Figure 43-104. Tx_128To255Octets_Packets_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX128_255OCTGB																															
R-0h																															

Table 43-158. Tx_128To255Octets_Packets_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TX128_255OCTGB	R	0h	Tx 128To255Octets Packets Good Bad This field indicates the number of good and bad packets transmitted with length between 128 and 255 (inclusive) bytes, exclusive of preamble and retried packets.

43.7.3.66 Tx_256To511Octets_Packets_Good_Bad Register (Offset = 730h) [Reset = 0h]

Tx_256To511Octets_Packets_Good_Bad is shown in Figure 43-105 and described in Table 43-159.

Return to the Summary Table.

This register provides the number of good and bad packets transmitted by DWC_ether_qos with length between 256 to 511 (inclusive) bytes, exclusive of preamble and retried packets.

Figure 43-105. Tx_256To511Octets_Packets_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX256_511OCTGB																															
R-0h																															

Table 43-159. Tx_256To511Octets_Packets_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TX256_511OCTGB	R	0h	Tx 256To511Octets Packets Good Bad This field indicates the number of good and bad packets transmitted with length between 256 and 511 (inclusive) bytes, exclusive of preamble and retried packets.

43.7.3.67 Tx_512To1023Octets_Packets_Good_Bad Register (Offset = 734h) [Reset = 0h]

Tx_512To1023Octets_Packets_Good_Bad is shown in Figure 43-106 and described in Table 43-160.

Return to the Summary Table.

This register provides the number of good and bad packets transmitted by DWC_ether_qos with length 512 to 1023 (inclusive) bytes, exclusive of preamble and retried packets.

Figure 43-106. Tx_512To1023Octets_Packets_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX512_1023OCTGB																															
R-0h																															

Table 43-160. Tx_512To1023Octets_Packets_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TX512_1023OCTGB	R	0h	Tx 512To1023Octets Packets Good Bad This field indicates the number of good and bad packets transmitted with length between 512 and 1023 (inclusive) bytes, exclusive of preamble and retried packets.

43.7.3.68 Tx_1024ToMaxOctets_Packets_Good_Bad Register (Offset = 738h) [Reset = 0h]

Tx_1024ToMaxOctets_Packets_Good_Bad is shown in Figure 43-107 and described in Table 43-161.

Return to the Summary Table.

This register provides the number of good and bad packets transmitted by DWC_ether_qos with length 1024 to maxsize (inclusive) bytes, exclusive of preamble and retried packets.

Figure 43-107. Tx_1024ToMaxOctets_Packets_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX1024_MAXOCTGB																															
R-0h																															

Table 43-161. Tx_1024ToMaxOctets_Packets_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TX1024_MAXOCTGB	R	0h	Tx 1024ToMaxOctets Packets Good Bad This field indicates the number of good and bad packets transmitted with length between 1024 and maxsize (inclusive) bytes, exclusive of preamble and retried packets.

43.7.3.69 Tx_Unicast_Packets_Good_Bad Register (Offset = 73Ch) [Reset = 0h]

Tx_Unicast_Packets_Good_Bad is shown in Figure 43-108 and described in Table 43-162.

Return to the Summary Table.

This register provides the number of good and bad unicast packets transmitted by DWC_ether_qos.

Figure 43-108. Tx_Unicast_Packets_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXUCASTGB																															
R-0h																															

Table 43-162. Tx_Unicast_Packets_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXUCASTGB	R	0h	Tx Unicast Packets Good Bad This field indicates the number of good and bad unicast packets transmitted.

43.7.3.70 Tx_Multicast_Packets_Good_Bad Register (Offset = 740h) [Reset = 0h]

Tx_Multicast_Packets_Good_Bad is shown in Figure 43-109 and described in Table 43-163.

Return to the Summary Table.

This register provides the number of good and bad multicast packets transmitted by DWC_ether_qos.

Figure 43-109. Tx_Multicast_Packets_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXMCASTGB																															
R-0h																															

Table 43-163. Tx_Multicast_Packets_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXMCASTGB	R	0h	Tx Multicast Packets Good Bad This field indicates the number of good and bad multicast packets transmitted.

43.7.3.71 Tx_Broadcast_Packets_Good_Bad Register (Offset = 744h) [Reset = 0h]

Tx_Broadcast_Packets_Good_Bad is shown in Figure 43-110 and described in Table 43-164.

Return to the Summary Table.

This register provides the number of good and bad broadcast packets transmitted by DWC_ether_qos.

Figure 43-110. Tx_Broadcast_Packets_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXBCASTGB																															
R-0h																															

Table 43-164. Tx_Broadcast_Packets_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXBCASTGB	R	0h	Tx Broadcast Packets Good Bad This field indicates the number of good and bad broadcast packets transmitted.

43.7.3.72 Tx_Underflow_Error_Packets Register (Offset = 748h) [Reset = 0h]

Tx_Underflow_Error_Packets is shown in Figure 43-111 and described in Table 43-165.

Return to the Summary Table.

This register provides the number of packets aborted by DWC_ether_qos because of packets underflow error.

Figure 43-111. Tx_Underflow_Error_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXUNDRFLW																															
R-0h																															

Table 43-165. Tx_Underflow_Error_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXUNDRFLW	R	0h	Tx Underflow Error Packets This field indicates the number of packets aborted because of packets underflow error.

43.7.3.73 Tx_Single_Collision_Good_Packets Register (Offset = 74Ch) [Reset = 0h]

Tx_Single_Collision_Good_Packets is shown in Figure 43-112 and described in Table 43-166.

Return to the Summary Table.

This register provides the number of successfully transmitted packets by DWC_ether_qos after a single collision in the half-duplex mode.

Figure 43-112. Tx_Single_Collision_Good_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXSNGLCOLG																															
R-0h																															

Table 43-166. Tx_Single_Collision_Good_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXSNGLCOLG	R	0h	Tx Single Collision Good Packets This field indicates the number of successfully transmitted packets after a single collision in the half-duplex mode.

43.7.3.74 Tx_Multiple_Collision_Good_Packets Register (Offset = 750h) [Reset = 0h]

Tx_Multiple_Collision_Good_Packets is shown in Figure 43-113 and described in Table 43-167.

Return to the Summary Table.

This register provides the number of successfully transmitted packets by DWC_ether_qos after multiple collisions in the half-duplex mode.

Figure 43-113. Tx_Multiple_Collision_Good_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXMULTCOLG																															
R-0h																															

Table 43-167. Tx_Multiple_Collision_Good_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXMULTCOLG	R	0h	Tx Multiple Collision Good Packets This field indicates the number of successfully transmitted packets after multiple collisions in the half-duplex mode.

43.7.3.75 Tx_Deferred_Packets Register (Offset = 754h) [Reset = 0h]

Tx_Deferred_Packets is shown in Figure 43-114 and described in Table 43-168.

Return to the Summary Table.

This register provides the number of successfully transmitted by DWC_ether_qos after a deferral in the half-duplex mode.

Figure 43-114. Tx_Deferred_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXDEFRD																															
R-0h																															

Table 43-168. Tx_Deferred_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXDEFRD	R	0h	Tx Deferred Packets This field indicates the number of successfully transmitted after a deferral in the half-duplex mode.

43.7.3.76 Tx_Late_Collision_Packets Register (Offset = 758h) [Reset = 0h]

Tx_Late_Collision_Packets is shown in Figure 43-115 and described in Table 43-169.

Return to the Summary Table.

This register provides the number of packets aborted by DWC_ether_qos because of late collision error.

Figure 43-115. Tx_Late_Collision_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXLATECOL																															
R-0h																															

Table 43-169. Tx_Late_Collision_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXLATECOL	R	0h	Tx Late Collision Packets This field indicates the number of packets aborted because of late collision error.

43.7.3.77 Tx_Excessive_Collision_Packets Register (Offset = 75Ch) [Reset = 0h]

Tx_Excessive_Collision_Packets is shown in Figure 43-116 and described in Table 43-170.

Return to the Summary Table.

This register provides the number of packets aborted by DWC_ether_qos because of excessive (16) collision errors.

Figure 43-116. Tx_Excessive_Collision_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXEXSCOL																															
R-0h																															

Table 43-170. Tx_Excessive_Collision_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXEXSCOL	R	0h	Tx Excessive Collision Packets This field indicates the number of packets aborted because of excessive (16) collision errors.

43.7.3.78 Tx_Carrier_Error_Packets Register (Offset = 760h) [Reset = 0h]

Tx_Carrier_Error_Packets is shown in Figure 43-117 and described in Table 43-171.

Return to the Summary Table.

This register provides the number of packets aborted by DWC_ether_qos because of carrier sense error (no carrier or loss of carrier).

Figure 43-117. Tx_Carrier_Error_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXCARR																															
R-0h																															

Table 43-171. Tx_Carrier_Error_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXCARR	R	0h	Tx Carrier Error Packets This field indicates the number of packets aborted because of carrier sense error (no carrier or loss of carrier).

43.7.3.79 Tx_Octet_Count_Good Register (Offset = 764h) [Reset = 0h]

Tx_Octet_Count_Good is shown in Figure 43-118 and described in Table 43-172.

Return to the Summary Table.

This register provides the number of bytes transmitted by DWC_ether_qos, exclusive of preamble, only in good packets.

Figure 43-118. Tx_Octet_Count_Good Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXOCTG																															
R-0h																															

Table 43-172. Tx_Octet_Count_Good Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXOCTG	R	0h	Tx Octet Count Good This field indicates the number of bytes transmitted, exclusive of preamble, only in good packets.

43.7.3.80 Tx_Packet_Count_Good Register (Offset = 768h) [Reset = 0h]

Tx_Packet_Count_Good is shown in Figure 43-119 and described in Table 43-173.

Return to the Summary Table.

This register provides the number of good packets transmitted by DWC_ether_qos.

Figure 43-119. Tx_Packet_Count_Good Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXPKTG																															
R-0h																															

Table 43-173. Tx_Packet_Count_Good Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXPKTG	R	0h	Tx Packet Count Good This field indicates the number of good packets transmitted.

43.7.3.81 Tx_Excessive_Deferral_Error Register (Offset = 76Ch) [Reset = 0h]

Tx_Excessive_Deferral_Error is shown in Figure 43-120 and described in Table 43-174.

Return to the Summary Table.

This register provides the number of packets aborted by DWC_ether_qos because of excessive deferral error (deferred for more than two max-sized packet times).

Figure 43-120. Tx_Excessive_Deferral_Error Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXEXSDEF																															
R-0h																															

Table 43-174. Tx_Excessive_Deferral_Error Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXEXSDEF	R	0h	Tx Excessive Deferral Error This field indicates the number of packets aborted because of excessive deferral error (deferred for more than two max-sized packet times).

43.7.3.82 Tx_Pause_Packets Register (Offset = 770h) [Reset = 0h]

Tx_Pause_Packets is shown in Figure 43-121 and described in Table 43-175.

Return to the Summary Table.

This register provides the number of good Pause packets transmitted by DWC_ether_qos.

Figure 43-121. Tx_Pause_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXPAUSE																															
R-0h																															

Table 43-175. Tx_Pause_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXPAUSE	R	0h	Tx Pause Packets This field indicates the number of good Pause packets transmitted.

43.7.3.83 Tx_VLAN_Packets_Good Register (Offset = 774h) [Reset = 0h]

Tx_VLAN_Packets_Good is shown in Figure 43-122 and described in Table 43-176.

Return to the Summary Table.

This register provides the number of good VLAN packets transmitted by DWC_ether_qos.

Figure 43-122. Tx_VLAN_Packets_Good Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXVLANG																															
R-0h																															

Table 43-176. Tx_VLAN_Packets_Good Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXVLANG	R	0h	Tx VLAN Packets Good This field provides the number of good VLAN packets transmitted.

43.7.3.84 Tx_OSize_Packets_Good Register (Offset = 778h) [Reset = 0h]

Tx_OSize_Packets_Good is shown in Figure 43-123 and described in Table 43-177.

Return to the Summary Table.

This register provides the number of packets transmitted by DWC_ether_qos without errors and with length greater than the maxsize (1,518 or 1,522 bytes for VLAN tagged packets
2000 bytes if enabled in S2KP bit of the MAC_Configuration register).

Figure 43-123. Tx_OSize_Packets_Good Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXOSIZG																															
R-0h																															

Table 43-177. Tx_OSize_Packets_Good Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXOSIZG	R	0h	Tx OSize Packets Good This field indicates the number of packets transmitted without errors and with length greater than the maxsize (1,518 or 1,522 bytes for VLAN tagged packets 2000 bytes if enabled in S2KP bit of the MAC_Configuration register).

43.7.3.85 Rx_Packets_Count_Good_Bad Register (Offset = 780h) [Reset = 0h]

Rx_Packets_Count_Good_Bad is shown in Figure 43-124 and described in Table 43-178.

Return to the Summary Table.

This register provides the number of good and bad packets received by DWC_ether_qos.

Figure 43-124. Rx_Packets_Count_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXPKTGB																															
R-0h																															

Table 43-178. Rx_Packets_Count_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXPKTGB	R	0h	Rx Packets Count Good Bad This field indicates the number of good and bad packets received.

43.7.3.86 Rx_Octet_Count_Good_Bad Register (Offset = 784h) [Reset = 0h]

Rx_Octet_Count_Good_Bad is shown in Figure 43-125 and described in Table 43-179.

Return to the Summary Table.

This register provides the number of bytes received by DWC_ther_qos, exclusive of preamble, in good and bad packets.

Figure 43-125. Rx_Octet_Count_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXOCTGB																															
R-0h																															

Table 43-179. Rx_Octet_Count_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXOCTGB	R	0h	Rx Octet Count Good Bad This field indicates the number of bytes received, exclusive of preamble, in good and bad packets.

43.7.3.87 Rx_Octet_Count_Good Register (Offset = 788h) [Reset = 0h]

Rx_Octet_Count_Good is shown in Figure 43-126 and described in Table 43-180.

Return to the Summary Table.

This register provides the number of bytes received by DWC_ether_qos, exclusive of preamble, only in good packets.

Figure 43-126. Rx_Octet_Count_Good Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXOCTG																															
R-0h																															

Table 43-180. Rx_Octet_Count_Good Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXOCTG	R	0h	Rx Octet Count Good This field indicates the number of bytes received, exclusive of preamble, only in good packets.

43.7.3.88 Rx_Broadcast_Packets_Good Register (Offset = 78Ch) [Reset = 0h]

Rx_Broadcast_Packets_Good is shown in Figure 43-127 and described in Table 43-181.

Return to the Summary Table.

This register provides the number of good broadcast packets received by DWC_ether_qos.

Figure 43-127. Rx_Broadcast_Packets_Good Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXBCASTG																															
R-0h																															

Table 43-181. Rx_Broadcast_Packets_Good Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXBCASTG	R	0h	Rx Broadcast Packets Good This field indicates the number of good broadcast packets received.

43.7.3.89 Rx_Multicast_Packets_Good Register (Offset = 790h) [Reset = 0h]

Rx_Multicast_Packets_Good is shown in Figure 43-128 and described in Table 43-182.

Return to the Summary Table.

This register provides the number of good multicast packets received by DWC_ether_qos.

Figure 43-128. Rx_Multicast_Packets_Good Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXMCASTG																															
R-0h																															

Table 43-182. Rx_Multicast_Packets_Good Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXMCASTG	R	0h	Rx Multicast Packets Good This field indicates the number of good multicast packets received.

43.7.3.90 Rx_CRC_Error_Packets Register (Offset = 794h) [Reset = 0h]

Rx_CRC_Error_Packets is shown in Figure 43-129 and described in Table 43-183.

Return to the Summary Table.

This register provides the number of packets received by DWC_ether_qos with CRC error.

Figure 43-129. Rx_CRC_Error_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXCRCERR																															
R-0h																															

Table 43-183. Rx_CRC_Error_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXCRCERR	R	0h	Rx CRC Error Packets This field indicates the number of packets received with CRC error.

43.7.3.91 Rx_Alignment_Error_Packets Register (Offset = 798h) [Reset = 0h]

Rx_Alignment_Error_Packets is shown in Figure 43-130 and described in Table 43-184.

Return to the Summary Table.

This register provides the number of packets received by DWC_ether_qos with alignment (dribble) error. It is valid only in 10/100 mode.

Figure 43-130. Rx_Alignment_Error_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXALGNERR																															
R-0h																															

Table 43-184. Rx_Alignment_Error_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXALGNERR	R	0h	Rx Alignment Error Packets This field indicates the number of packets received with alignment (dribble) error. It is valid only in 10/100 mode.

43.7.3.92 Rx_Runt_Error_Packets Register (Offset = 79Ch) [Reset = 0h]

Rx_Runt_Error_Packets is shown in Figure 43-131 and described in Table 43-185.

Return to the Summary Table.

This register provides the number of packets received by DWC_ether_qos with runt (length less than 64 bytes and CRC error) error.

Figure 43-131. Rx_Runt_Error_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXRUNTERR																															
R-0h																															

Table 43-185. Rx_Runt_Error_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXRUNTERR	R	0h	Rx Runt Error Packets This field indicates the number of packets received with runt (length less than 64 bytes and CRC error) error.

43.7.3.93 Rx_Jabber_Error_Packets Register (Offset = 7A0h) [Reset = 0h]

Rx_Jabber_Error_Packets is shown in Figure 43-132 and described in Table 43-186.

Return to the Summary Table.

This register provides the number of giant packets received by DWC_ether_qos with length (including CRC) greater than 1,518 bytes (1,522 bytes for VLAN tagged) and with CRC error. If Jumbo Packet mode is enabled, packets of length greater than 9,018 bytes (9,022 bytes for VLAN tagged) are considered as giant packets.

Figure 43-132. Rx_Jabber_Error_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXJABERR																															
R-0h																															

Table 43-186. Rx_Jabber_Error_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXJABERR	R	0h	Rx Jabber Error Packets This field indicates the number of giant packets received with length (including CRC) greater than 1,518 bytes (1,522 bytes for VLAN tagged) and with CRC error. If Jumbo Packet mode is enabled, packets of length greater than 9,018 bytes (9,022 bytes for VLAN tagged) are considered as giant packets.

43.7.3.94 Rx_Undersize_Packets_Good Register (Offset = 7A4h) [Reset = 0h]

Rx_Undersize_Packets_Good is shown in Figure 43-133 and described in Table 43-187.

Return to the Summary Table.

This register provides the number of packets received by DWC_ether_qos with length less than 64 bytes, without any errors.

Figure 43-133. Rx_Undersize_Packets_Good Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXUNDERSZG																															
R-0h																															

Table 43-187. Rx_Undersize_Packets_Good Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXUNDERSZG	R	0h	Rx Undersize Packets Good This field indicates the number of packets received with length less than 64 bytes, without any errors.

43.7.3.95 Rx_Oversize_Packets_Good Register (Offset = 7A8h) [Reset = 0h]

Rx_Oversize_Packets_Good is shown in Figure 43-134 and described in Table 43-188.

Return to the Summary Table.

This register provides the number of packets received by DWC_ether_qos without errors, with length greater than the maxsize (1,518 bytes or 1,522 bytes for VLAN tagged packets
2000 bytes if enabled in the S2KP bit of the MAC_Configuration register).

Figure 43-134. Rx_Oversize_Packets_Good Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXOVERSZG																															
R-0h																															

Table 43-188. Rx_Oversize_Packets_Good Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXOVERSZG	R	0h	Rx Oversize Packets Good This field indicates the number of packets received without errors, with length greater than the maxsize (1,518 bytes or 1,522 bytes for VLAN tagged packets 2000 bytes if enabled in the S2KP bit of the MAC_Configuration register).

43.7.3.96 Rx_64Octets_Packets_Good_Bad Register (Offset = 7ACh) [Reset = 0h]

Rx_64Octets_Packets_Good_Bad is shown in Figure 43-135 and described in Table 43-189.

Return to the Summary Table.

This register provides the number of good and bad packets received by DWC_ether_qos with length 64 bytes, exclusive of the preamble.

Figure 43-135. Rx_64Octets_Packets_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX64OCTGB																															
R-0h																															

Table 43-189. Rx_64Octets_Packets_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RX64OCTGB	R	0h	Rx 64 Octets Packets Good Bad This field indicates the number of good and bad packets received with length 64 bytes, exclusive of the preamble.

43.7.3.97 Rx_65To127Octets_Packets_Good_Bad Register (Offset = 7B0h) [Reset = 0h]

Rx_65To127Octets_Packets_Good_Bad is shown in Figure 43-136 and described in Table 43-190.

Return to the Summary Table.

This register provides the number of good and bad packets received by DWC_ether_qos with length between 65 and 127 (inclusive) bytes, exclusive of the preamble.

Figure 43-136. Rx_65To127Octets_Packets_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX65_127OCTGB																															
R-0h																															

Table 43-190. Rx_65To127Octets_Packets_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RX65_127OCTGB	R	0h	Rx 65-127 Octets Packets Good Bad This field indicates the number of good and bad packets received with length between 65 and 127 (inclusive) bytes, exclusive of the preamble.

43.7.3.98 Rx_128To255Octets_Packets_Good_Bad Register (Offset = 7B4h) [Reset = 0h]

Rx_128To255Octets_Packets_Good_Bad is shown in Figure 43-137 and described in Table 43-191.

Return to the Summary Table.

This register provides the number of good and bad packets received by DWC_ether_qos with length between 128 and 255 (inclusive) bytes, exclusive of the preamble.

Figure 43-137. Rx_128To255Octets_Packets_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX128_255OCTGB																															
R-0h																															

Table 43-191. Rx_128To255Octets_Packets_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RX128_255OCTGB	R	0h	Rx 128-255 Octets Packets Good Bad This field indicates the number of good and bad packets received with length between 128 and 255 (inclusive) bytes, exclusive of the preamble.

43.7.3.99 Rx_256To511Octets_Packets_Good_Bad Register (Offset = 7B8h) [Reset = 0h]

Rx_256To511Octets_Packets_Good_Bad is shown in Figure 43-138 and described in Table 43-192.

Return to the Summary Table.

This register provides the number of good and bad packets received by DWC_ether_qos with length between 256 and 511 (inclusive) bytes, exclusive of the preamble.

Figure 43-138. Rx_256To511Octets_Packets_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX256_511OCTGB																															
R-0h																															

Table 43-192. Rx_256To511Octets_Packets_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RX256_511OCTGB	R	0h	Rx 256-511 Octets Packets Good Bad This field indicates the number of good and bad packets received with length between 256 and 511 (inclusive) bytes, exclusive of the preamble.

43.7.3.100 Rx_512To1023Octets_Packets_Good_Bad Register (Offset = 7BCh) [Reset = 0h]

Rx_512To1023Octets_Packets_Good_Bad is shown in Figure 43-139 and described in Table 43-193.

Return to the Summary Table.

This register provides the number of good and bad packets received by DWC_ether_qos with length between 512 and 1023 (inclusive) bytes, exclusive of the preamble.

Figure 43-139. Rx_512To1023Octets_Packets_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX512_1023OCTGB																															
R-0h																															

Table 43-193. Rx_512To1023Octets_Packets_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RX512_1023OCTGB	R	0h	RX 512-1023 Octets Packets Good Bad This field indicates the number of good and bad packets received with length between 512 and 1023 (inclusive) bytes, exclusive of the preamble.

43.7.3.101 Rx_1024ToMaxOctets_Packets_Good_Bad Register (Offset = 7C0h) [Reset = 0h]

Rx_1024ToMaxOctets_Packets_Good_Bad is shown in Figure 43-140 and described in Table 43-194.

Return to the Summary Table.

This register provides the number of good and bad packets received by DWC_ether_qos with length between 1024 and maxsize (inclusive) bytes, exclusive of the preamble.

Figure 43-140. Rx_1024ToMaxOctets_Packets_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX1024_MAXOCTGB																															
R-0h																															

Table 43-194. Rx_1024ToMaxOctets_Packets_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RX1024_MAXOCTGB	R	0h	Rx 1024-Max Octets Good Bad This field indicates the number of good and bad packets received with length between 1024 and maxsize (inclusive) bytes, exclusive of the preamble.

43.7.3.102 Rx_Unicast_Packets_Good Register (Offset = 7C4h) [Reset = 0h]

Rx_Unicast_Packets_Good is shown in Figure 43-141 and described in Table 43-195.

Return to the Summary Table.

This register provides the number of good unicast packets received by DWC_ether_qos.

Figure 43-141. Rx_Unicast_Packets_Good Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXUCASTG																															
R-0h																															

Table 43-195. Rx_Unicast_Packets_Good Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXUCASTG	R	0h	Rx Unicast Packets Good This field indicates the number of good unicast packets received.

43.7.3.103 Rx_Length_Error_Packets Register (Offset = 7C8h) [Reset = 0h]

Rx_Length_Error_Packets is shown in Figure 43-142 and described in Table 43-196.

Return to the Summary Table.

This register provides the number of packets received by DWC_ether_qos with length error (Length Type field not equal to packet size), for all packets with valid length field.

Figure 43-142. Rx_Length_Error_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXLENERR																															
R-0h																															

Table 43-196. Rx_Length_Error_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXLENERR	R	0h	Rx Length Error Packets This field indicates the number of packets received with length error (Length Type field not equal to packet size), for all packets with valid length field.

43.7.3.104 Rx_Out_Of_Range_Type_Packets Register (Offset = 7CCh) [Reset = 0h]

Rx_Out_Of_Range_Type_Packets is shown in Figure 43-143 and described in Table 43-197.

Return to the Summary Table.

This register provides the number of packets received by DWC_ether_qos with length field not equal to the valid packet size (greater than 1,500 but less than 1,536).

Figure 43-143. Rx_Out_Of_Range_Type_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXOUTOFRNG																															
R-0h																															

Table 43-197. Rx_Out_Of_Range_Type_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXOUTOFRNG	R	0h	Rx Out of Range Type Packet This field indicates the number of packets received with length field not equal to the valid packet size (greater than 1,500 but less than 1,536).

43.7.3.105 Rx_Pause_Packets Register (Offset = 7D0h) [Reset = 0h]

Rx_Pause_Packets is shown in Figure 43-144 and described in Table 43-198.

Return to the Summary Table.

This register provides the number of good and valid Pause packets received by DWC_ether_qos.

Figure 43-144. Rx_Pause_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXPAUSEPKT																															
R-0h																															

Table 43-198. Rx_Pause_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXPAUSEPKT	R	0h	Rx Pause Packets This field indicates the number of good and valid Pause packets received.

43.7.3.106 Rx_FIFO_Overflow_Packets Register (Offset = 7D4h) [Reset = 0h]

Rx_FIFO_Overflow_Packets is shown in Figure 43-145 and described in Table 43-199.

Return to the Summary Table.

This register provides the number of missed received packets because of FIFO overflow in DWC_ether_qos.

Figure 43-145. Rx_FIFO_Overflow_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXFIFOOVFL																															
R-0h																															

Table 43-199. Rx_FIFO_Overflow_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXFIFOOVFL	R	0h	Rx FIFO Overflow Packets This field indicates the number of missed received packets because of FIFO overflow.

43.7.3.107 Rx_VLAN_Packets_Good_Bad Register (Offset = 7D8h) [Reset = 0h]

Rx_VLAN_Packets_Good_Bad is shown in Figure 43-146 and described in Table 43-200.

Return to the Summary Table.

This register provides the number of good and bad VLAN packets received by DWC_ether_qos.

Figure 43-146. Rx_VLAN_Packets_Good_Bad Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXVLANPKTGB																															
R-0h																															

Table 43-200. Rx_VLAN_Packets_Good_Bad Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXVLANPKTGB	R	0h	Rx VLAN Packets Good Bad This field indicates the number of good and bad VLAN packets received.

43.7.3.108 Rx_Watchdog_Error_Packets Register (Offset = 7DCh) [Reset = 0h]

Rx_Watchdog_Error_Packets is shown in Figure 43-147 and described in Table 43-201.

Return to the Summary Table.

This register provides the number of packets received by DWC_ether_qos with error because of watchdog timeout error (packets with a data load larger than 2,048 bytes (when JE and WD bits are reset in MAC_Configuration register), 10,240 bytes (when JE bit is set and WD bit is reset in MAC_Configuration register), 16,384 bytes (when WD bit is set in MAC_Configuration register) or the value programmed in the MAC_Watchdog_Timeout register).

Figure 43-147. Rx_Watchdog_Error_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXWDGERR																															
R-0h																															

Table 43-201. Rx_Watchdog_Error_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXWDGERR	R	0h	Rx Watchdog Error Packets This field indicates the number of packets received with error because of watchdog timeout error (packets with a data load larger than 2,048 bytes (when JE and WD bits are reset in MAC_Configuration register), 10,240 bytes (when JE bit is set and WD bit is reset in MAC_Configuration register), 16,384 bytes (when WD bit is set in MAC_Configuration register) or the value programmed in the MAC_Watchdog_Timeout register).

43.7.3.109 Rx_Receive_Error_Packets Register (Offset = 7E0h) [Reset = 0h]

Rx_Receive_Error_Packets is shown in Figure 43-148 and described in Table 43-202.

Return to the Summary Table.

This register provides the number of packets received by DWC_ether_qos with Receive error or Packet Extension error on the GMII or MII interface.

Figure 43-148. Rx_Receive_Error_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXRCVERR																															
R-0h																															

Table 43-202. Rx_Receive_Error_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXRCVERR	R	0h	Rx Receive Error Packets This field indicates the number of packets received with Receive error or Packet Extension error on the GMII or MII interface.

43.7.3.110 Rx_Control_Packets_Good Register (Offset = 7E4h) [Reset = 0h]

Rx_Control_Packets_Good is shown in Figure 43-149 and described in Table 43-203.

Return to the Summary Table.

This register provides the number of good control packets received by DWC_ether_qos.

Figure 43-149. Rx_Control_Packets_Good Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXCTRLG																															
R-0h																															

Table 43-203. Rx_Control_Packets_Good Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXCTRLG	R	0h	Rx Control Packets Good This field indicates the number of good control packets received.

43.7.3.111 Tx_LPI_USEC_Cntr Register (Offset = 7ECh) [Reset = 0h]

Tx_LPI_USEC_Cntr is shown in Figure 43-150 and described in Table 43-204.

Return to the Summary Table.

This register provides the number of microseconds Tx LPI is asserted by DWC_ether_qos.

Figure 43-150. Tx_LPI_USEC_Cntr Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXLPIUSC																															
R-0h																															

Table 43-204. Tx_LPI_USEC_Cntr Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXLPIUSC	R	0h	Tx LPI Microseconds Counter This field indicates the number of microseconds Tx LPI is asserted. For every Tx LPI Entry and Exit, the Timer value can have an error of +/- 1 microsecond.

43.7.3.112 Tx_LPI_Tran_Cntr Register (Offset = 7F0h) [Reset = 0h]

Tx_LPI_Tran_Cntr is shown in Figure 43-151 and described in Table 43-205.

Return to the Summary Table.

This register provides the number of times DWC_ether_qos has entered Tx LPI.

Figure 43-151. Tx_LPI_Tran_Cntr Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXLPITRC																															
R-0h																															

Table 43-205. Tx_LPI_Tran_Cntr Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXLPITRC	R	0h	Tx LPI Transition counter This field indicates the number of times Tx LPI Entry has occurred. Even if Tx LPI Entry occurs in Automate Mode (because of LPITXA bit set in the LPI Control and Status register), the counter will increment.

43.7.3.113 Rx_LPI_USEC_Cntr Register (Offset = 7F4h) [Reset = 0h]

Rx_LPI_USEC_Cntr is shown in Figure 43-152 and described in Table 43-206.

Return to the Summary Table.

This register provides the number of microseconds Rx LPI is sampled by DWC_ether_qos.

Figure 43-152. Rx_LPI_USEC_Cntr Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXLPIUSC																															
R-0h																															

Table 43-206. Rx_LPI_USEC_Cntr Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXLPIUSC	R	0h	Rx LPI Microseconds Counter This field indicates the number of microseconds Rx LPI is asserted. For every Rx LPI Entry and Exit, the Timer value can have an error of +/- 1 microsecond.

43.7.3.114 Rx_LPI_Tran_Cntr Register (Offset = 7F8h) [Reset = 0h]

Rx_LPI_Tran_Cntr is shown in Figure 43-153 and described in Table 43-207.

Return to the Summary Table.

This register provides the number of times DWC_ether_qos has entered Rx LPI.

Figure 43-153. Rx_LPI_Tran_Cntr Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXLPITRC																															
R-0h																															

Table 43-207. Rx_LPI_Tran_Cntr Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXLPITRC	R	0h	Rx LPI Transition counter This field indicates the number of times Rx LPI Entry has occurred.

43.7.3.115 MMC_IPC_Rx_Interrupt_Mask Register (Offset = 800h) [Reset = 0h]

MMC_IPC_Rx_Interrupt_Mask is shown in Figure 43-154 and described in Table 43-208.

Return to the Summary Table.

This register maintains the mask for the interrupt generated from the receive IPC statistic counters. The MMC Receive Checksum Off load Interrupt Mask register maintains the masks for the interrupts generated when the receive IPC (Checksum Off load) statistic counters reach half their maximum value, and when they reach their maximum values. This register is 32 bits wide.

Figure 43-154. MMC_IPC_Rx_Interrupt_Mask Register

31	30	29	28	27	26	25	24
RESERVED		RXICMPEROIM	RXICMPGOIM	RXTCPEROIM	RXTCPGOIM	RXUDPEROIM	RXUDPGOIM
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RXIPV6NOPAYOIM	RXIPV6HEROIM	RXIPV6GOIM	RXIPV4UDSBLOIM	RXIPV4FRAGOIM	RXIPV4NOPAYOIM	RXIPV4HEROIM	RXIPV4GOIM
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED		RXICMPERPIM	RXICMPGPIM	RXTCPERPIM	RXTCPGPIM	RXUDPERPIM	RXUDPGPIM
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RXIPV6NOPAYPIM	RXIPV6HERPIM	RXIPV6GPIM	RXIPV4UDSBLPIM	RXIPV4FRAGPIM	RXIPV4NOPAYPIM	RXIPV4HERPIM	RXIPV4GPIM
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 43-208. MMC_IPC_Rx_Interrupt_Mask Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved.
29	RXICMPEROIM	R/W	0h	MMC Receive ICMP Error Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxicmp_err_octets counter reaches half of the maximum value or the maximum value. 0h = MMC Receive ICMP Error Octet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive ICMP Error Octet Counter Interrupt Mask is enabled : 0x1
28	RXICMPGOIM	R/W	0h	MMC Receive ICMP Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxicmp_gd_octets counter reaches half of the maximum value or the maximum value. 0h = MMC Receive ICMP Good Octet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive ICMP Good Octet Counter Interrupt Mask is enabled : 0x1
27	RXTCPEROIM	R/W	0h	MMC Receive TCP Error Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxtcp_err_octets counter reaches half of the maximum value or the maximum value. 0h = MMC Receive TCP Error Octet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive TCP Error Octet Counter Interrupt Mask is enabled : 0x1
26	RXTCPGOIM	R/W	0h	MMC Receive TCP Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxtcp_gd_octets counter reaches half of the maximum value or the maximum value. 0h = MMC Receive TCP Good Octet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive TCP Good Octet Counter Interrupt Mask is enabled : 0x1

Table 43-208. MMC_IPC_Rx_Interrupt_Mask Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	RXUDPEROIM	R/W	0h	MMC Receive UDP Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxudp_err_octets counter reaches half of the maximum value or the maximum value. 0h = MMC Receive UDP Good Octet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive UDP Good Octet Counter Interrupt Mask is enabled : 0x1
24	RXUDPGOIM	R/W	0h	MMC Receive IPV6 No Payload Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxudp_gd_octets counter reaches half of the maximum value or the maximum value. 0h = MMC Receive IPV6 No Payload Octet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive IPV6 No Payload Octet Counter Interrupt Mask is enabled : 0x1
23	RXIPV6NOPAYOIM	R/W	0h	MMC Receive IPV6 Header Error Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv6_nopay_octets counter reaches half of the maximum value or the maximum value. 0h = MMC Receive IPV6 Header Error Octet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive IPV6 Header Error Octet Counter Interrupt Mask is enabled : 0x1
22	RXIPV6HEROIM	R/W	0h	MMC Receive IPV6 Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv6_hdrerr_octets counter reaches half of the maximum value or the maximum value. 0h = MMC Receive IPV6 Good Octet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive IPV6 Good Octet Counter Interrupt Mask is enabled : 0x1
21	RXIPV6GOIM	R/W	0h	MMC Receive IPV6 Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv6_gd_octets counter reaches half of the maximum value or the maximum value. 0h = MMC Receive IPV6 Good Octet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive IPV6 Good Octet Counter Interrupt Mask is enabled : 0x1
20	RXIPV4UDSBLOIM	R/W	0h	MMC Receive IPV4 UDP Checksum Disabled Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_udsbl_octets counter reaches half of the maximum value or the maximum value. 0h = MMC Receive IPV4 UDP Checksum Disabled Octet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive IPV4 UDP Checksum Disabled Octet Counter Interrupt Mask is enabled : 0x1
19	RXIPV4FRAGOIM	R/W	0h	MMC Receive IPV4 Fragmented Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_frag_octets counter reaches half of the maximum value or the maximum value. 0h = MMC Receive IPV4 Fragmented Octet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive IPV4 Fragmented Octet Counter Interrupt Mask is enabled : 0x1
18	RXIPV4NOPAYOIM	R/W	0h	MMC Receive IPV4 No Payload Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_nopay_octets counter reaches half of the maximum value or the maximum value. 0h = MMC Receive IPV4 No Payload Octet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive IPV4 No Payload Octet Counter Interrupt Mask is enabled : 0x1

Table 43-208. MMC_IPC_Rx_Interrupt_Mask Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	RXIPV4HEROIM	R/W	0h	MMC Receive IPV4 Header Error Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_hdrerr_octets counter reaches half of the maximum value or the maximum value. 0h = MMC Receive IPV4 Header Error Octet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive IPV4 Header Error Octet Counter Interrupt Mask is enabled : 0x1
16	RXIPV4GOIM	R/W	0h	MMC Receive IPV4 Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_gd_octets counter reaches half of the maximum value or the maximum value. 0h = MMC Receive IPV4 Good Octet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive IPV4 Good Octet Counter Interrupt Mask is enabled : 0x1
15-14	RESERVED	R	0h	Reserved.
13	RXICMPERPIM	R/W	0h	MMC Receive ICMP Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxicmp_err_pkts counter reaches half of the maximum value or the maximum value. 0h = MMC Receive ICMP Error Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive ICMP Error Packet Counter Interrupt Mask is enabled : 0x1
12	RXICMPGPIM	R/W	0h	MMC Receive ICMP Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxicmp_gd_pkts counter reaches half of the maximum value or the maximum value. 0h = MMC Receive ICMP Good Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive ICMP Good Packet Counter Interrupt Mask is enabled : 0x1
11	RXTCPERPIM	R/W	0h	MMC Receive TCP Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxtcp_err_pkts counter reaches half of the maximum value or the maximum value. 0h = MMC Receive TCP Error Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive TCP Error Packet Counter Interrupt Mask is enabled : 0x1
10	RXTCPGPIM	R/W	0h	MMC Receive TCP Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxtcp_gd_pkts counter reaches half of the maximum value or the maximum value. 0h = MMC Receive TCP Good Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive TCP Good Packet Counter Interrupt Mask is enabled : 0x1
9	RXUDPERPIM	R/W	0h	MMC Receive UDP Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxudp_err_pkts counter reaches half of the maximum value or the maximum value. 0h = MMC Receive UDP Error Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive UDP Error Packet Counter Interrupt Mask is enabled : 0x1
8	RXUDPGPIM	R/W	0h	MMC Receive UDP Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxudp_gd_pkts counter reaches half of the maximum value or the maximum value. 0h = MMC Receive UDP Good Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive UDP Good Packet Counter Interrupt Mask is enabled : 0x1

Table 43-208. MMC_IPC_Rx_Interrupt_Mask Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	RXIPV6NOPAYPIM	R/W	0h	MMC Receive IPV6 No Payload Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv6_nopay_pkts counter reaches half of the maximum value or the maximum value. 0h = MMC Receive IPV6 No Payload Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive IPV6 No Payload Packet Counter Interrupt Mask is enabled : 0x1
6	RXIPV6HERPIM	R/W	0h	MMC Receive IPV6 Header Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv6_hdrerr_pkts counter reaches half of the maximum value or the maximum value. 0h = MMC Receive IPV6 Header Error Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive IPV6 Header Error Packet Counter Interrupt Mask is enabled : 0x1
5	RXIPV6GPIM	R/W	0h	MMC Receive IPV6 Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv6_gd_pkts counter reaches half of the maximum value or the maximum value. 0h = MMC Receive IPV6 Good Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive IPV6 Good Packet Counter Interrupt Mask is enabled : 0x1
4	RXIPV4UDSBLPIM	R/W	0h	MMC Receive IPV4 UDP Checksum Disabled Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_udsbl_pkts counter reaches half of the maximum value or the maximum value. 0h = MMC Receive IPV4 UDP Checksum Disabled Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive IPV4 UDP Checksum Disabled Packet Counter Interrupt Mask is enabled : 0x1
3	RXIPV4FRAGPIM	R/W	0h	MMC Receive IPV4 Fragmented Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_frag_pkts counter reaches half of the maximum value or the maximum value. 0h = MMC Receive IPV4 Fragmented Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive IPV4 Fragmented Packet Counter Interrupt Mask is enabled : 0x1
2	RXIPV4NOPAYPIM	R/W	0h	MMC Receive IPV4 No Payload Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_nopay_pkts counter reaches half of the maximum value or the maximum value. 0h = MMC Receive IPV4 No Payload Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive IPV4 No Payload Packet Counter Interrupt Mask is enabled : 0x1
1	RXIPV4HERPIM	R/W	0h	MMC Receive IPV4 Header Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_hdrerr_pkts counter reaches half of the maximum value or the maximum value. 0h = MMC Receive IPV4 Header Error Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive IPV4 Header Error Packet Counter Interrupt Mask is enabled : 0x1
0	RXIPV4GPIM	R/W	0h	MMC Receive IPV4 Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_gd_pkts counter reaches half of the maximum value or the maximum value. 0h = MMC Receive IPV4 Good Packet Counter Interrupt Mask is disabled : 0x0 1h = MMC Receive IPV4 Good Packet Counter Interrupt Mask is enabled : 0x1

43.7.3.116 MMC_IPC_Rx_Interrupt Register (Offset = 808h) [Reset = 0h]

MMC_IPC_Rx_Interrupt is shown in Figure 43-155 and described in Table 43-209.

Return to the Summary Table.

This register maintains the interrupt that the receive IPC statistic counters generate.
The MMC Receive Checksum Offload Interrupt register maintains the interrupts generated when receive IPC statistic counters reach half their maximum values (0x8000_0000 for 32 bit counter and 0x8000 for 16 bit counter), and when they cross their maximum values (0xFFFF_FFFF for 32 bit counter and 0xFFFF for 16 bit counter). When Counter Stop Rollover is set, the interrupts are set but the counter remains at all-ones.
The MMC Receive Checksum Offload Interrupt register is 32 bit wide. When the MMC IPC counter that caused the interrupt is read, its corresponding interrupt bit is cleared. The counter's least-significant byte lane (Bits[7:0]) must be read to clear the interrupt bit.

Figure 43-155. MMC_IPC_Rx_Interrupt Register

31	30	29	28	27	26	25	24
RESERVED		RXICMPEROIS	RXICMPGOIS	RXTCPEROIS	RXTCPGOIS	RXUDPEROIS	RXUDPGOIS
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
RXIPV6NOPAYOIS	RXIPV6HEROIS	RXIPV6GOIS	RXIPV4UDSBLOIS	RXIPV4FRAGOIS	RXIPV4NOPAYOIS	RXIPV4HEROIS	RXIPV4GOIS
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED		RXICMPERPIS	RXICMPGPIS	RXTCPERPIS	RXTCPGPIS	RXUDPERPIS	RXUDPGPIS
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RXIPV6NOPAYPIS	RXIPV6HERPIS	RXIPV6GPIS	RXIPV4UDSBLPIS	RXIPV4FRAGPIS	RXIPV4NOPAYPIS	RXIPV4HERPIS	RXIPV4GPIS
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 43-209. MMC_IPC_Rx_Interrupt Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved.
29	RXICMPEROIS	R	0h	MMC Receive ICMP Error Octet Counter Interrupt Status This bit is set when the rxicmp_err_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive ICMP Error Octet Counter Interrupt Status not detected : 0x0 1h = MMC Receive ICMP Error Octet Counter Interrupt Status detected : 0x1
28	RXICMPGOIS	R	0h	MMC Receive ICMP Good Octet Counter Interrupt Status This bit is set when the rxicmp_gd_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive ICMP Good Octet Counter Interrupt Status not detected : 0x0 1h = MMC Receive ICMP Good Octet Counter Interrupt Status detected : 0x1

Table 43-209. MMC_IPC_Rx_Interrupt Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	RXTCPEROIS	R	0h	MMC Receive TCP Error Octet Counter Interrupt Status This bit is set when the rxtcp_err_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive TCP Error Octet Counter Interrupt Status not detected : 0x0 1h = MMC Receive TCP Error Octet Counter Interrupt Status detected : 0x1
26	RXTCPGOIS	R	0h	MMC Receive TCP Good Octet Counter Interrupt Status This bit is set when the rxtcp_gd_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive TCP Good Octet Counter Interrupt Status not detected : 0x0 1h = MMC Receive TCP Good Octet Counter Interrupt Status detected : 0x1
25	RXUDPEROIS	R	0h	MMC Receive UDP Error Octet Counter Interrupt Status This bit is set when the rxudp_err_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive UDP Error Octet Counter Interrupt Status not detected : 0x0 1h = MMC Receive UDP Error Octet Counter Interrupt Status detected : 0x1
24	RXUDPGOIS	R	0h	MMC Receive UDP Good Octet Counter Interrupt Status This bit is set when the rxudp_gd_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive UDP Good Octet Counter Interrupt Status not detected : 0x0 1h = MMC Receive UDP Good Octet Counter Interrupt Status detected : 0x1
23	RXIPV6NOPAYOIS	R	0h	MMC Receive IPV6 No Payload Octet Counter Interrupt Status This bit is set when the rxipv6_nopay_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive IPV6 No Payload Octet Counter Interrupt Status not detected : 0x0 1h = MMC Receive IPV6 No Payload Octet Counter Interrupt Status detected : 0x1
22	RXIPV6HEROIS	R	0h	MMC Receive IPV6 Header Error Octet Counter Interrupt Status This bit is set when the rxipv6_hdrerr_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive IPV6 Header Error Octet Counter Interrupt Status not detected : 0x0 1h = MMC Receive IPV6 Header Error Octet Counter Interrupt Status detected : 0x1
21	RXIPV6GOIS	R	0h	MMC Receive IPV6 Good Octet Counter Interrupt Status This bit is set when the rxipv6_gd_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive IPV6 Good Octet Counter Interrupt Status not detected : 0x0 1h = MMC Receive IPV6 Good Octet Counter Interrupt Status detected : 0x1

Table 43-209. MMC_IPC_Rx_Interrupt Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	RXIPV4UDSBLOIS	R	0h	MMC Receive IPV4 UDP Checksum Disabled Octet Counter Interrupt Status This bit is set when the rxipv4_udsbl_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive IPV4 UDP Checksum Disabled Octet Counter Interrupt Status not detected : 0x0 1h = MMC Receive IPV4 UDP Checksum Disabled Octet Counter Interrupt Status detected : 0x1
19	RXIPV4FRAGOIS	R	0h	MMC Receive IPV4 Fragmented Octet Counter Interrupt Status This bit is set when the rxipv4_frag_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive IPV4 Fragmented Octet Counter Interrupt Status not detected : 0x0 1h = MMC Receive IPV4 Fragmented Octet Counter Interrupt Status detected : 0x1
18	RXIPV4NOPAYOIS	R	0h	MMC Receive IPV4 No Payload Octet Counter Interrupt Status This bit is set when the rxipv4_nopay_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive IPV4 No Payload Octet Counter Interrupt Status not detected : 0x0 1h = MMC Receive IPV4 No Payload Octet Counter Interrupt Status detected : 0x1
17	RXIPV4HEROIS	R	0h	MMC Receive IPV4 Header Error Octet Counter Interrupt Status This bit is set when the rxipv4_hdrerr_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive IPV4 Header Error Octet Counter Interrupt Status not detected : 0x0 1h = MMC Receive IPV4 Header Error Octet Counter Interrupt Status detected : 0x1
16	RXIPV4GOIS	R	0h	MMC Receive IPV4 Good Octet Counter Interrupt Status This bit is set when the rxipv4_gd_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive IPV4 Good Octet Counter Interrupt Status not detected : 0x0 1h = MMC Receive IPV4 Good Octet Counter Interrupt Status detected : 0x1
15-14	RESERVED	R	0h	Reserved.
13	RXICMPERPIS	R	0h	MMC Receive ICMP Error Packet Counter Interrupt Status This bit is set when the rxicmp_err_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive ICMP Error Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive ICMP Error Packet Counter Interrupt Status detected : 0x1

Table 43-209. MMC_IPC_Rx_Interrupt Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	RXICMPGPIS	R	0h	MMC Receive ICMP Good Packet Counter Interrupt Status This bit is set when the rxicmp_gd_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive ICMP Good Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive ICMP Good Packet Counter Interrupt Status detected : 0x1
11	RXTCPERPIS	R	0h	MMC Receive TCP Error Packet Counter Interrupt Status This bit is set when the rxtcp_err_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive TCP Error Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive TCP Error Packet Counter Interrupt Status detected : 0x1
10	RXTCPGPIS	R	0h	MMC Receive TCP Good Packet Counter Interrupt Status This bit is set when the rxtcp_gd_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive TCP Good Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive TCP Good Packet Counter Interrupt Status detected : 0x1
9	RXUDPERPIS	R	0h	MMC Receive UDP Error Packet Counter Interrupt Status This bit is set when the rxudp_err_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive UDP Error Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive UDP Error Packet Counter Interrupt Status detected : 0x1
8	RXUDPGPIS	R	0h	MC Receive UDP Good Packet Counter Interrupt Status This bit is set when the rxudp_gd_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive UDP Good Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive UDP Good Packet Counter Interrupt Status detected : 0x1
7	RXIPV6NOPAYPIS	R	0h	MMC Receive IPV6 No Payload Packet Counter Interrupt Status This bit is set when the rxipv6_nopay_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive IPV6 No Payload Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive IPV6 No Payload Packet Counter Interrupt Status detected : 0x1
6	RXIPV6HERPIS	R	0h	MMC Receive IPV6 Header Error Packet Counter Interrupt Status This bit is set when the rxipv6_hdrerr_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive IPV6 Header Error Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive IPV6 Header Error Packet Counter Interrupt Status detected : 0x1

Table 43-209. MMC_IPC_Rx_Interrupt Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	RXIPV6GPIS	R	0h	MMC Receive IPV6 Good Packet Counter Interrupt Status This bit is set when the rxipv6_gd_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive IPV6 Good Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive IPV6 Good Packet Counter Interrupt Status detected : 0x1
4	RXIPV4UDSBLPIS	R	0h	MMC Receive IPV4 UDP Checksum Disabled Packet Counter Interrupt Status This bit is set when the rxipv4_udsbl_pkts counter reaches half of the maximum value or the maximum value. 0h = MMC Receive IPV4 UDP Checksum Disabled Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive IPV4 UDP Checksum Disabled Packet Counter Interrupt Status detected : 0x1
3	RXIPV4FRAGPIS	R	0h	MMC Receive IPV4 Fragmented Packet Counter Interrupt Status This bit is set when the rxipv4_frag_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive IPV4 Fragmented Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive IPV4 Fragmented Packet Counter Interrupt Status detected : 0x1
2	RXIPV4NOPAYPIS	R	0h	MMC Receive IPV4 No Payload Packet Counter Interrupt Status This bit is set when the rxipv4_nopay_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive IPV4 No Payload Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive IPV4 No Payload Packet Counter Interrupt Status detected : 0x1
1	RXIPV4HERPIS	R	0h	MMC Receive IPV4 Header Error Packet Counter Interrupt Status This bit is set when the rxipv4_hdrerr_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive IPV4 Header Error Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive IPV4 Header Error Packet Counter Interrupt Status detected : 0x1
0	RXIPV4GPIS	R	0h	MMC Receive IPV4 Good Packet Counter Interrupt Status This bit is set when the rxipv4_gd_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = MMC Receive IPV4 Good Packet Counter Interrupt Status not detected : 0x0 1h = MMC Receive IPV4 Good Packet Counter Interrupt Status detected : 0x1

43.7.3.117 RxIPv4_Good_Packets Register (Offset = 810h) [Reset = 0h]

RxIPv4_Good_Packets is shown in Figure 43-156 and described in Table 43-210.

Return to the Summary Table.

This register provides the number of good IPv4 datagrams received by DWC_ether_qos with the TCP, UDP, or ICMP payload.

Figure 43-156. RxIPv4_Good_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV4GDPKT																															
R-0h																															

Table 43-210. RxIPv4_Good_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXIPV4GDPKT	R	0h	RxIPv4 Good Packets This field indicates the number of good IPv4 datagrams received with the TCP, UDP, or ICMP payload.

43.7.3.118 RxIPv4_Header_Error_Packets Register (Offset = 814h) [Reset = 0h]

RxIPv4_Header_Error_Packets is shown in Figure 43-157 and described in Table 43-211.

Return to the Summary Table.

RxIPv4 Header Error Packets
This register provides the number of IPv4 datagrams received by DWC_ether_qos with header (checksum, length, or version mismatch) errors.

Figure 43-157. RxIPv4_Header_Error_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV4HDRERRPKT																															
R-0h																															

Table 43-211. RxIPv4_Header_Error_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXIPV4HDRERRPKT	R	0h	RxIPv4 Header Error Packets This field indicates the number of IPv4 datagrams received with header (checksum, length, or version mismatch) errors.

43.7.3.119 RxIPv4_No_Payload_Packets Register (Offset = 818h) [Reset = 0h]

RxIPv4_No_Payload_Packets is shown in Figure 43-158 and described in Table 43-212.

Return to the Summary Table.

This register provides the number of IPv4 datagram packets received by DWC_ether_qos that did not have a TCP, UDP, or ICMP payload.

Figure 43-158. RxIPv4_No_Payload_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV4NOPAYPKT																															
R-0h																															

Table 43-212. RxIPv4_No_Payload_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXIPV4NOPAYPKT	R	0h	RxIPv4 Payload Packets This field indicates the number of IPv4 datagram packets received that did not have a TCP, UDP, or ICMP payload.

43.7.3.120 RxIPv4_Fragmented_Packets Register (Offset = 81Ch) [Reset = 0h]

RxIPv4_Fragmented_Packets is shown in Figure 43-159 and described in Table 43-213.

Return to the Summary Table.

This register provides the number of good IPv4 datagrams received by DWC_ether_qos with fragmentation.

Figure 43-159. RxIPv4_Fragmented_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV4FRAGPKT																															
R-0h																															

Table 43-213. RxIPv4_Fragmented_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXIPV4FRAGPKT	R	0h	RxIPv4 Fragmented Packets This field indicates the number of good IPv4 datagrams received with fragmentation.

43.7.3.121 RxIPv4_UDP_Checksum_Disabled_Packets Register (Offset = 820h) [Reset = 0h]

RxIPv4_UDP_Checksum_Disabled_Packets is shown in Figure 43-160 and described in Table 43-214.

Return to the Summary Table.

This register provides the number of good IPv4 datagrams received by DWC_ether_qos that had a UDP payload with checksum disabled.

Figure 43-160. RxIPv4_UDP_Checksum_Disabled_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV4UDSBLPKT																															
R-0h																															

Table 43-214. RxIPv4_UDP_Checksum_Disabled_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXIPV4UDSBLPKT	R	0h	RxIPv4 UDP Checksum Disabled Packets This field indicates the number of good IPv4 datagrams received that had a UDP payload with checksum disabled.

43.7.3.122 RxIPv6_Good_Packets Register (Offset = 824h) [Reset = 0h]

RxIPv6_Good_Packets is shown in Figure 43-161 and described in Table 43-215.

Return to the Summary Table.

This register provides the number of good IPv6 datagrams received by DWC_ether_qos with the TCP, UDP, or ICMP payload.

Figure 43-161. RxIPv6_Good_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV6GDPKT																															
R-0h																															

Table 43-215. RxIPv6_Good_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXIPV6GDPKT	R	0h	RxIPv6 Good Packets This field indicates the number of good IPv6 datagrams received with the TCP, UDP, or ICMP payload.

43.7.3.123 RxIPv6_Header_Error_Packets Register (Offset = 828h) [Reset = 0h]

RxIPv6_Header_Error_Packets is shown in Figure 43-162 and described in Table 43-216.

Return to the Summary Table.

This register provides the number of IPv6 datagrams received by DWC_ether_qos with header (length or version mismatch) errors.

Figure 43-162. RxIPv6_Header_Error_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV6HDRERRPKT																															
R-0h																															

Table 43-216. RxIPv6_Header_Error_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXIPV6HDRERRPKT	R	0h	RxIPv6 Header Error Packets This field indicates the number of IPv6 datagrams received with header (length or version mismatch) errors.

43.7.3.124 RxIPv6_No_Payload_Packets Register (Offset = 82Ch) [Reset = 0h]

RxIPv6_No_Payload_Packets is shown in Figure 43-163 and described in Table 43-217.

Return to the Summary Table.

This register provides the number of IPv6 datagram packets received by DWC_ether_qos that did not have a TCP, UDP, or ICMP payload. This includes all IPv6 datagrams with fragmentation or security extension headers.

Figure 43-163. RxIPv6_No_Payload_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV6NOPAYPKT																															
R-0h																															

Table 43-217. RxIPv6_No_Payload_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXIPV6NOPAYPKT	R	0h	RxIPv6 Payload Packets This field indicates the number of IPv6 datagram packets received that did not have a TCP, UDP, or ICMP payload. This includes all IPv6 datagrams with fragmentation or security extension headers.

43.7.3.125 RxUDP_Good_Packets Register (Offset = 830h) [Reset = 0h]

RxUDP_Good_Packets is shown in Figure 43-164 and described in Table 43-218.

Return to the Summary Table.

This register provides the number of good IP datagrams received by DWC_ether_qos with a good UDP payload. This counter is not updated when the RxIPv4_UDP_Checksum_Disabled_Packets counter is incremented.

Figure 43-164. RxUDP_Good_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXUDPGDPKT																															
R-0h																															

Table 43-218. RxUDP_Good_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXUDPGDPKT	R	0h	RxUDP Good Packets This field indicates the number of good IP datagrams received with a good UDP payload. This counter is not updated when the RxIPv4_UDP_Checksum_Disabled_Packets counter is incremented.

43.7.3.126 RxUDP_Error_Packets Register (Offset = 834h) [Reset = 0h]

RxUDP_Error_Packets is shown in Figure 43-165 and described in Table 43-219.

Return to the Summary Table.

This register provides the number of good IP datagrams received by DWC_ether_qos whose UDP payload has a checksum error.

Figure 43-165. RxUDP_Error_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXUDPERRPKT																															
R-0h																															

Table 43-219. RxUDP_Error_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXUDPERRPKT	R	0h	RxUDP Error Packets This field indicates the number of good IP datagrams received whose UDP payload has a checksum error.

43.7.3.127 RxTCP_Good_Packets Register (Offset = 838h) [Reset = 0h]

RxTCP_Good_Packets is shown in Figure 43-166 and described in Table 43-220.

Return to the Summary Table.

This register provides the number of good IP datagrams received by DWC_ether_qos with a good TCP payload.

Figure 43-166. RxTCP_Good_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXTCPGDPKT																															
R-0h																															

Table 43-220. RxTCP_Good_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXTCPGDPKT	R	0h	RxTCP Good Packets This field indicates the number of good IP datagrams received with a good TCP payload.

43.7.3.128 RxTCP_Error_Packets Register (Offset = 83Ch) [Reset = 0h]

RxTCP_Error_Packets is shown in Figure 43-167 and described in Table 43-221.

Return to the Summary Table.

This register provides the number of good IP datagrams received by DWC_ether_qos whose TCP payload has a checksum error.

Figure 43-167. RxTCP_Error_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXTCPERRPKT																															
R-0h																															

Table 43-221. RxTCP_Error_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXTCPERRPKT	R	0h	RxTCP Error Packets This field indicates the number of good IP datagrams received whose TCP payload has a checksum error.

43.7.3.129 RxICMP_Good_Packets Register (Offset = 840h) [Reset = 0h]

RxICMP_Good_Packets is shown in Figure 43-168 and described in Table 43-222.

Return to the Summary Table.

This register provides the number of good IP datagrams received by DWC_ether_qos with a good ICMP payload.

Figure 43-168. RxICMP_Good_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXICMPGDPKT																															
R-0h																															

Table 43-222. RxICMP_Good_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXICMPGDPKT	R	0h	RxICMP Good Packets This field indicates the number of good IP datagrams received with a good ICMP payload.

43.7.3.130 RxICMP_Error_Packets Register (Offset = 844h) [Reset = 0h]

RxICMP_Error_Packets is shown in Figure 43-169 and described in Table 43-223.

Return to the Summary Table.

This register provides the number of good IP datagrams received by DWC_ether_qos whose ICMP payload has a checksum error.

Figure 43-169. RxICMP_Error_Packets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXICMPERRPKT																															
R-0h																															

Table 43-223. RxICMP_Error_Packets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXICMPERRPKT	R	0h	RxICMP Error Packets This field indicates the number of good IP datagrams received whose ICMP payload has a checksum error.

43.7.3.131 RxIPv4_Good_Octets Register (Offset = 850h) [Reset = 0h]

RxIPv4_Good_Octets is shown in Figure 43-170 and described in Table 43-224.

Return to the Summary Table.

This register provides the number of bytes received by DWC_ether_qos in good IPv4 datagrams encapsulating TCP, UDP, or ICMP data. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

Figure 43-170. RxIPv4_Good_Octets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV4GDOCT																															
R-0h																															

Table 43-224. RxIPv4_Good_Octets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXIPV4GDOCT	R	0h	RxIPv4 Good Octets This field indicates the number of bytes received in good IPv4 datagrams encapsulating TCP, UDP, or ICMP data. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

43.7.3.132 RxIPv4_Header_Error_Octets Register (Offset = 854h) [Reset = 0h]

RxIPv4_Header_Error_Octets is shown in Figure 43-171 and described in Table 43-225.

Return to the Summary Table.

This register provides the number of bytes received by DWC_ether_qos in IPv4 datagrams with header errors (checksum, length, version mismatch). The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

Figure 43-171. RxIPv4_Header_Error_Octets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV4HDRERROCT																															
R-0h																															

Table 43-225. RxIPv4_Header_Error_Octets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXIPV4HDRERROCT	R	0h	RxIPv4 Header Error Octets This field indicates the number of bytes received in IPv4 datagrams with header errors (checksum, length, version mismatch). The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

43.7.3.133 RxIPv4_No_Payload_Octets Register (Offset = 858h) [Reset = 0h]

RxIPv4_No_Payload_Octets is shown in Figure 43-172 and described in Table 43-226.

Return to the Summary Table.

This register provides the number of bytes received by DWC_ether_qos in IPv4 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

Figure 43-172. RxIPv4_No_Payload_Octets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV4NOPAYOCT																															
R-0h																															

Table 43-226. RxIPv4_No_Payload_Octets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXIPV4NOPAYOCT	R	0h	RxIPv4 Payload Octets This field indicates the number of bytes received in IPv4 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

43.7.3.134 RxIPv4_Fragmented_Octets Register (Offset = 85Ch) [Reset = 0h]

RxIPv4_Fragmented_Octets is shown in Figure 43-173 and described in Table 43-227.

Return to the Summary Table.

This register provides the number of bytes received by DWC_ether_qos in fragmented IPv4 datagrams. The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

Figure 43-173. RxIPv4_Fragmented_Octets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV4FRAGOCT																															
R-0h																															

Table 43-227. RxIPv4_Fragmented_Octets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXIPV4FRAGOCT	R	0h	RxIPv4 Fragmented Octets This field indicates the number of bytes received in fragmented IPv4 datagrams. The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

43.7.3.135 RxIPv4_UDP_Checksum_Disable_Octets Register (Offset = 860h) [Reset = 0h]

RxIPv4_UDP_Checksum_Disable_Octets is shown in Figure 43-174 and described in Table 43-228.

Return to the Summary Table.

This register provides the number of bytes received by DWC_ether_qos in a UDP segment that had the UDP checksum disabled. This counter does not count IP Header bytes. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

Figure 43-174. RxIPv4_UDP_Checksum_Disable_Octets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV4UDSBLOCT																															
R-0h																															

Table 43-228. RxIPv4_UDP_Checksum_Disable_Octets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXIPV4UDSBLOCT	R	0h	RxIPv4 UDP Checksum Disable Octets This field indicates the number of bytes received in a UDP segment that had the UDP checksum disabled. This counter does not count IP Header bytes. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

43.7.3.136 RxIPv6_Good_Octets Register (Offset = 864h) [Reset = 0h]

RxIPv6_Good_Octets is shown in Figure 43-175 and described in Table 43-229.

Return to the Summary Table.

This register provides the number of bytes received by DWC_ether_qos in good IPv6 datagrams encapsulating TCP, UDP, or ICMP data. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

Figure 43-175. RxIPv6_Good_Octets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV6GDOCT																															
R-0h																															

Table 43-229. RxIPv6_Good_Octets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXIPV6GDOCT	R	0h	RxIPv6 Good Octets This field indicates the number of bytes received in good IPv6 datagrams encapsulating TCP, UDP, or ICMP data. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

43.7.3.137 RxIPv6_Header_Error_Octets Register (Offset = 868h) [Reset = 0h]

RxIPv6_Header_Error_Octets is shown in Figure 43-176 and described in Table 43-230.

Return to the Summary Table.

This register provides the number of bytes received by DWC_ether_qos in IPv6 datagrams with header errors (length, version mismatch). The value in the Length field of IPv6 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

Figure 43-176. RxIPv6_Header_Error_Octets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV6HDRERROCT																															
R-0h																															

Table 43-230. RxIPv6_Header_Error_Octets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXIPV6HDRERROCT	R	0h	RxIPv6 Header Error Octets This field indicates the number of bytes received in IPv6 datagrams with header errors (length, version mismatch). The value in the Length field of IPv6 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

43.7.3.138 RxIPv6_No_Payload_Octets Register (Offset = 86Ch) [Reset = 0h]

RxIPv6_No_Payload_Octets is shown in Figure 43-177 and described in Table 43-231.

Return to the Summary Table.

This register provides the number of bytes received by DWC_ether_qos in IPv6 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the Length field of IPv6 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

Figure 43-177. RxIPv6_No_Payload_Octets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIPV6NOPAYOCT																															
R-0h																															

Table 43-231. RxIPv6_No_Payload_Octets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXIPV6NOPAYOCT	R	0h	RxIPv6 Payload Octets This field indicates the number of bytes received in IPv6 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the Length field of IPv6 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

43.7.3.139 RxUDP_Good_Octets Register (Offset = 870h) [Reset = 0h]

RxUDP_Good_Octets is shown in Figure 43-178 and described in Table 43-232.

Return to the Summary Table.

This register provides the number of bytes received by DWC_ether_qos in a good UDP segment. This counter does not count IP header bytes.

Figure 43-178. RxUDP_Good_Octets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXUDPGDOCT																															
R-0h																															

Table 43-232. RxUDP_Good_Octets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXUDPGDOCT	R	0h	RxUDP Good Octets This field indicates the number of bytes received in a good UDP segment. This counter does not count IP header bytes.

43.7.3.140 RxUDP_Error_Octets Register (Offset = 874h) [Reset = 0h]

RxUDP_Error_Octets is shown in Figure 43-179 and described in Table 43-233.

Return to the Summary Table.

This register provides the number of bytes received by DWC_ether_qos in a UDP segment that had checksum errors. This counter does not count IP header bytes.

Figure 43-179. RxUDP_Error_Octets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXUDPERROCT																															
R-0h																															

Table 43-233. RxUDP_Error_Octets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXUDPERROCT	R	0h	RxUDP Error Octets This field indicates the number of bytes received in a UDP segment that had checksum errors. This counter does not count IP header bytes.

43.7.3.141 RxTCP_Good_Octets Register (Offset = 878h) [Reset = 0h]

RxTCP_Good_Octets is shown in Figure 43-180 and described in Table 43-234.

Return to the Summary Table.

This register provides the number of bytes received by DWC_ether_qos in a good TCP segment. This counter does not count IP header bytes.

Figure 43-180. RxTCP_Good_Octets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXTCPGDOCT																															
R-0h																															

Table 43-234. RxTCP_Good_Octets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXTCPGDOCT	R	0h	RxTCP Good Octets This field indicates the number of bytes received in a good TCP segment. This counter does not count IP header bytes.

43.7.3.142 RxTCP_Error_Octets Register (Offset = 87Ch) [Reset = 0h]

RxTCP_Error_Octets is shown in Figure 43-181 and described in Table 43-235.

Return to the Summary Table.

This register provides the number of bytes received by DWC_ether_qos in a TCP segment that had checksum errors. This counter does not count IP header bytes.

Figure 43-181. RxTCP_Error_Octets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXTCPERROCT																															
R-0h																															

Table 43-235. RxTCP_Error_Octets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXTCPERROCT	R	0h	RxTCP Error Octets This field indicates the number of bytes received in a TCP segment that had checksum errors. This counter does not count IP header bytes.

43.7.3.143 RxICMP_Good_Octets Register (Offset = 880h) [Reset = 0h]

RxICMP_Good_Octets is shown in Figure 43-182 and described in Table 43-236.

Return to the Summary Table.

This register provides the number of bytes received by DWC_ether_qos in a good ICMP segment. This counter does not count IP header bytes.

Figure 43-182. RxICMP_Good_Octets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXICMPGDOCT																															
R-0h																															

Table 43-236. RxICMP_Good_Octets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXICMPGDOCT	R	0h	RxICMP Good Octets This field indicates the number of bytes received in a good ICMP segment. This counter does not count IP header bytes.

43.7.3.144 RxICMP_Error_Octets Register (Offset = 884h) [Reset = 0h]

RxICMP_Error_Octets is shown in Figure 43-183 and described in Table 43-237.

Return to the Summary Table.

This register provides the number of bytes received by DWC_ether_qos in a ICMP segment that had checksum errors. This counter does not count IP header bytes.

Figure 43-183. RxICMP_Error_Octets Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXICMPERROCT																															
R-0h																															

Table 43-237. RxICMP_Error_Octets Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXICMPERROCT	R	0h	RxICMP Error Octets This field indicates the number of bytes received in a ICMP segment that had checksum errors. This counter does not count IP header bytes.

43.7.3.145 MAC_L3_L4_Control0 Register (Offset = 900h) [Reset = 0h]

MAC_L3_L4_Control0 is shown in Figure 43-184 and described in Table 43-238.

Return to the Summary Table.

The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Figure 43-184. MAC_L3_L4_Control0 Register

31	30	29	28	27	26	25	24
RESERVED			DMCHEN0	RESERVED			DMCHN0
R-0h			R/W-0h	R-0h			R/W-0h
23	22	21	20	19	18	17	16
RESERVED		L4DPIM0	L4DPM0	L4SPIM0	L4SPM0	RESERVED	L4PEN0
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h
15	14	13	12	11	10	9	8
L3HDBM0					L3HSBM0		
R/W-0h					R/W-0h		
7	6	5	4	3	2	1	0
L3HSBM0		L3DAIM0	L3DAM0	L3SAIM0	L3SAM0	RESERVED	L3PEN0
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h

Table 43-238. MAC_L3_L4_Control0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved.
28	DMCHEN0	R/W	0h	DMA Channel Select Enable When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When this bit is reset, the DMA channel is not decided by this filter. 0h = DMA Channel Select is disabled : 0x0 1h = DMA Channel Select is enabled : 0x1
27-25	RESERVED	R	0h	Reserved.
24	DMCHN0	R/W	0h	DMA Channel Number When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22	RESERVED	R	0h	Reserved.
21	L4DPIM0	R/W	0h	Layer 4 Destination Port Inverse Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high. 0h = Layer 4 Destination Port Inverse Match is disabled : 0x0 1h = Layer 4 Destination Port Inverse Match is enabled : 0x1
20	L4DPM0	R/W	0h	Layer 4 Destination Port Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching. 0h = Layer 4 Destination Port Match is disabled : 0x0 1h = Layer 4 Destination Port Match is enabled : 0x1

Table 43-238. MAC_L3_L4_Control0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	L4SPIM0	R/W	0h	Layer 4 Source Port Inverse Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high. 0h = Layer 4 Source Port Inverse Match is disabled : 0x0 1h = Layer 4 Source Port Inverse Match is enabled : 0x1
18	L4SPM0	R/W	0h	Layer 4 Source Port Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching. 0h = Layer 4 Source Port Match is disabled : 0x0 1h = Layer 4 Source Port Match is enabled : 0x1
17	RESERVED	R	0h	Reserved.
16	L4PEN0	R/W	0h	Layer 4 Protocol Enable When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set. 0h = Layer 4 Protocol is disabled : 0x0 1h = Layer 4 Protocol is enabled : 0x1
15-11	L3HDBM0	R/W	0h	Layer 3 IP DA Higher Bits Match IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSb[0] is masked. - 2: Two LSbs [1:0] are masked - .. - 127: All bits except MSb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
10-6	L3HSBM0	R/W	0h	Layer 3 IP SA Higher Bits Match IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.

Table 43-238. MAC_L3_L4_Control0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	L3DAIM0	R/W	0h	Layer 3 IP DA Inverse Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high. 0h = Layer 3 IP DA Inverse Match is disabled : 0x0 1h = Layer 3 IP DA Inverse Match is enabled : 0x1
4	L3DAM0	R/W	0h	Layer 3 IP DA Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering. 0h = Layer 3 IP DA Match is disabled : 0x0 1h = Layer 3 IP DA Match is enabled : 0x1
3	L3SAIM0	R/W	0h	Layer 3 IP SA Inverse Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set. 0h = Layer 3 IP SA Inverse Match is disabled : 0x0 1h = Layer 3 IP SA Inverse Match is enabled : 0x1
2	L3SAM0	R/W	0h	Layer 3 IP SA Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering. 0h = Layer 3 IP SA Match is disabled : 0x0 1h = Layer 3 IP SA Match is enabled : 0x1
1	RESERVED	R	0h	Reserved.
0	L3PEN0	R/W	0h	Layer 3 Protocol Enable When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set. 0h = Layer 3 Protocol is disabled : 0x0 1h = Layer 3 Protocol is enabled : 0x1

43.7.3.146 MAC_Layer4_Address0 Register (Offset = 904h) [Reset = 0h]

MAC_Layer4_Address0 is shown in Figure 43-185 and described in Table 43-239.

Return to the Summary Table.

The Layer 4 Address 0 register and registers 580 through 667 are reserved (RO with default value) if Enable Layer 3 and Layer 4 Packet Filter option is not selected while configuring the core.
You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option while configuring the core. When you select this option, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform consecutive writes to same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.

Figure 43-185. MAC_Layer4_Address0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L4DP0																L4SP0															
R/W-0h																R/W-0h															

Table 43-239. MAC_Layer4_Address0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	L4DP0	R/W	0h	Layer 4 Destination Port Number Field When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.
15-0	L4SP0	R/W	0h	Layer 4 Source Port Number Field When the L4PEN0 bit is reset and the L4SPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4SPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.

43.7.3.147 MAC_Layer3_Addr0_Reg0 Register (Offset = 910h) [Reset = 0h]

MAC_Layer3_Addr0_Reg0 is shown in Figure 43-186 and described in Table 43-240.

Return to the Summary Table.

For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Figure 43-186. MAC_Layer3_Addr0_Reg0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A00																															
R/W-0h																															

Table 43-240. MAC_Layer3_Addr0_Reg0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	L3A00	R/W	0h	Layer 3 Address 0 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

43.7.3.148 MAC_Layer3_Addr1_Reg0 Register (Offset = 914h) [Reset = 0h]

MAC_Layer3_Addr1_Reg0 is shown in Figure 43-187 and described in Table 43-241.

Return to the Summary Table.

For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Figure 43-187. MAC_Layer3_Addr1_Reg0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A10																															
R/W-0h																															

Table 43-241. MAC_Layer3_Addr1_Reg0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	L3A10	R/W	0h	Layer 3 Address 1 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

43.7.3.149 MAC_Layer3_Addr2_Reg0 Register (Offset = 918h) [Reset = 0h]

MAC_Layer3_Addr2_Reg0 is shown in Figure 43-188 and described in Table 43-242.

Return to the Summary Table.

The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Figure 43-188. MAC_Layer3_Addr2_Reg0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A20																															
R/W-0h																															

Table 43-242. MAC_Layer3_Addr2_Reg0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	L3A20	R/W	0h	Layer 3 Address 2 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

43.7.3.150 MAC_Layer3_Addr3_Reg0 Register (Offset = 91Ch) [Reset = 0h]

MAC_Layer3_Addr3_Reg0 is shown in Figure 43-189 and described in Table 43-243.

Return to the Summary Table.

The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Figure 43-189. MAC_Layer3_Addr3_Reg0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A30																															
R/W-0h																															

Table 43-243. MAC_Layer3_Addr3_Reg0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	L3A30	R/W	0h	Layer 3 Address 3 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

43.7.3.151 MAC_L3_L4_Control1 Register (Offset = 930h) [Reset = 0h]

MAC_L3_L4_Control1 is shown in Figure 43-190 and described in Table 43-244.

Return to the Summary Table.

The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Figure 43-190. MAC_L3_L4_Control1 Register

31	30	29	28	27	26	25	24
RESERVED			DMCHEN1	RESERVED			DMCHN1
R-0h			R/W-0h	R-0h			R/W-0h
23	22	21	20	19	18	17	16
RESERVED		L4DPIM1	L4DPM1	L4SPIM1	L4SPM1	RESERVED	L4PEN1
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h
15	14	13	12	11	10	9	8
L3HDBM1					L3HSBM1		
R/W-0h					R/W-0h		
7	6	5	4	3	2	1	0
L3HSBM1		L3DAIM1	L3DAM1	L3SAIM1	L3SAM1	RESERVED	L3PEN1
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h

Table 43-244. MAC_L3_L4_Control1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved.
28	DMCHEN1	R/W	0h	DMA Channel Select Enable When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When this bit is reset, the DMA channel is not decided by this filter. 0h = DMA Channel Select is disabled : 0x0 1h = DMA Channel Select is enabled : 0x1
27-25	RESERVED	R	0h	Reserved.
24	DMCHN1	R/W	0h	DMA Channel Number When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22	RESERVED	R	0h	Reserved.
21	L4DPIM1	R/W	0h	Layer 4 Destination Port Inverse Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high. 0h = Layer 4 Destination Port Inverse Match is disabled : 0x0 1h = Layer 4 Destination Port Inverse Match is enabled : 0x1
20	L4DPM1	R/W	0h	Layer 4 Destination Port Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching. 0h = Layer 4 Destination Port Match is disabled : 0x0 1h = Layer 4 Destination Port Match is enabled : 0x1

Table 43-244. MAC_L3_L4_Control1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	L4SPIM1	R/W	0h	Layer 4 Source Port Inverse Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high. 0h = Layer 4 Source Port Inverse Match is disabled : 0x0 1h = Layer 4 Source Port Inverse Match is enabled : 0x1
18	L4SPM1	R/W	0h	Layer 4 Source Port Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching. 0h = Layer 4 Source Port Match is disabled : 0x0 1h = Layer 4 Source Port Match is enabled : 0x1
17	RESERVED	R	0h	Reserved.
16	L4PEN1	R/W	0h	Layer 4 Protocol Enable When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set. 0h = Layer 4 Protocol is disabled : 0x0 1h = Layer 4 Protocol is enabled : 0x1
15-11	L3HDBM1	R/W	0h	Layer 3 IP DA Higher Bits Match IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSb[0] is masked. - 2: Two LSbs [1:0] are masked - .. - 127: All bits except MSb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
10-6	L3HSBM1	R/W	0h	Layer 3 IP SA Higher Bits Match IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.

Table 43-244. MAC_L3_L4_Control1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	L3DAIM1	R/W	0h	Layer 3 IP DA Inverse Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high. 0h = Layer 3 IP DA Inverse Match is disabled : 0x0 1h = Layer 3 IP DA Inverse Match is enabled : 0x1
4	L3DAM1	R/W	0h	Layer 3 IP DA Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering. 0h = Layer 3 IP DA Match is disabled : 0x0 1h = Layer 3 IP DA Match is enabled : 0x1
3	L3SAIM1	R/W	0h	Layer 3 IP SA Inverse Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set. 0h = Layer 3 IP SA Inverse Match is disabled : 0x0 1h = Layer 3 IP SA Inverse Match is enabled : 0x1
2	L3SAM1	R/W	0h	Layer 3 IP SA Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering. 0h = Layer 3 IP SA Match is disabled : 0x0 1h = Layer 3 IP SA Match is enabled : 0x1
1	RESERVED	R	0h	Reserved.
0	L3PEN1	R/W	0h	Layer 3 Protocol Enable When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set. 0h = Layer 3 Protocol is disabled : 0x0 1h = Layer 3 Protocol is enabled : 0x1

43.7.3.152 MAC_Layer4_Address1 Register (Offset = 934h) [Reset = 0h]

MAC_Layer4_Address1 is shown in Figure 43-191 and described in Table 43-245.

Return to the Summary Table.

The Layer 4 Address 0 register and registers 580 through 667 are reserved (RO with default value) if Enable Layer 3 and Layer 4 Packet Filter option is not selected while configuring the core.
You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option while configuring the core. When you select this option, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform consecutive writes to same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.

Figure 43-191. MAC_Layer4_Address1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L4DP1																L4SP1															
R/W-0h																R/W-0h															

Table 43-245. MAC_Layer4_Address1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	L4DP1	R/W	0h	Layer 4 Destination Port Number Field When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.
15-0	L4SP1	R/W	0h	Layer 4 Source Port Number Field When the L4PEN0 bit is reset and the L4SPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4SPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.

43.7.3.153 MAC_Layer3_Addr0_Reg1 Register (Offset = 940h) [Reset = 0h]

MAC_Layer3_Addr0_Reg1 is shown in Figure 43-192 and described in Table 43-246.

Return to the Summary Table.

For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Figure 43-192. MAC_Layer3_Addr0_Reg1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A01																															
R/W-0h																															

Table 43-246. MAC_Layer3_Addr0_Reg1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	L3A01	R/W	0h	Layer 3 Address 0 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

43.7.3.154 MAC_Layer3_Addr1_Reg1 Register (Offset = 944h) [Reset = 0h]

MAC_Layer3_Addr1_Reg1 is shown in Figure 43-193 and described in Table 43-247.

Return to the Summary Table.

For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Figure 43-193. MAC_Layer3_Addr1_Reg1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A11																															
R/W-0h																															

Table 43-247. MAC_Layer3_Addr1_Reg1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	L3A11	R/W	0h	Layer 3 Address 1 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

43.7.3.155 MAC_Layer3_Addr2_Reg1 Register (Offset = 948h) [Reset = 0h]

MAC_Layer3_Addr2_Reg1 is shown in Figure 43-194 and described in Table 43-248.

Return to the Summary Table.

The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Figure 43-194. MAC_Layer3_Addr2_Reg1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A21																															
R/W-0h																															

Table 43-248. MAC_Layer3_Addr2_Reg1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	L3A21	R/W	0h	Layer 3 Address 2 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

43.7.3.156 MAC_Layer3_Addr3_Reg1 Register (Offset = 94Ch) [Reset = 0h]

MAC_Layer3_Addr3_Reg1 is shown in Figure 43-195 and described in Table 43-249.

Return to the Summary Table.

The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Figure 43-195. MAC_Layer3_Addr3_Reg1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A31																															
R/W-0h																															

Table 43-249. MAC_Layer3_Addr3_Reg1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	L3A31	R/W	0h	Layer 3 Address 3 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

43.7.3.157 MAC_L3_L4_Control2 Register (Offset = 960h) [Reset = 0h]

MAC_L3_L4_Control2 is shown in Figure 43-196 and described in Table 43-250.

Return to the Summary Table.

The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Figure 43-196. MAC_L3_L4_Control2 Register

31	30	29	28	27	26	25	24
RESERVED			DMCHEN2	RESERVED			DMCHN2
R-0h			R/W-0h	R-0h			R/W-0h
23	22	21	20	19	18	17	16
RESERVED		L4DPIM2	L4DPM2	L4SPIM2	L4SPM2	RESERVED	L4PEN2
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h
15	14	13	12	11	10	9	8
L3HDBM2					L3HSBM2		
R/W-0h					R/W-0h		
7	6	5	4	3	2	1	0
L3HSBM2		L3DAIM2	L3DAM2	L3SAIM2	L3SAM2	RESERVED	L3PEN2
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h

Table 43-250. MAC_L3_L4_Control2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved.
28	DMCHEN2	R/W	0h	DMA Channel Select Enable When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When this bit is reset, the DMA channel is not decided by this filter. 0h = DMA Channel Select is disabled : 0x0 1h = DMA Channel Select is enabled : 0x1
27-25	RESERVED	R	0h	Reserved.
24	DMCHN2	R/W	0h	DMA Channel Number When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22	RESERVED	R	0h	Reserved.
21	L4DPIM2	R/W	0h	Layer 4 Destination Port Inverse Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high. 0h = Layer 4 Destination Port Inverse Match is disabled : 0x0 1h = Layer 4 Destination Port Inverse Match is enabled : 0x1
20	L4DPM2	R/W	0h	Layer 4 Destination Port Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching. 0h = Layer 4 Destination Port Match is disabled : 0x0 1h = Layer 4 Destination Port Match is enabled : 0x1

Table 43-250. MAC_L3_L4_Control2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	L4SPIM2	R/W	0h	Layer 4 Source Port Inverse Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high. 0h = Layer 4 Source Port Inverse Match is disabled : 0x0 1h = Layer 4 Source Port Inverse Match is enabled : 0x1
18	L4SPM2	R/W	0h	Layer 4 Source Port Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching. 0h = Layer 4 Source Port Match is disabled : 0x0 1h = Layer 4 Source Port Match is enabled : 0x1
17	RESERVED	R	0h	Reserved.
16	L4PEN2	R/W	0h	Layer 4 Protocol Enable When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set. 0h = Layer 4 Protocol is disabled : 0x0 1h = Layer 4 Protocol is enabled : 0x1
15-11	L3HDBM2	R/W	0h	Layer 3 IP DA Higher Bits Match IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSb[0] is masked. - 2: Two LSbs [1:0] are masked - .. - 127: All bits except MSb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
10-6	L3HSBM2	R/W	0h	Layer 3 IP SA Higher Bits Match IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.

Table 43-250. MAC_L3_L4_Control2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	L3DAIM2	R/W	0h	Layer 3 IP DA Inverse Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high. 0h = Layer 3 IP DA Inverse Match is disabled : 0x0 1h = Layer 3 IP DA Inverse Match is enabled : 0x1
4	L3DAM2	R/W	0h	Layer 3 IP DA Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering. 0h = Layer 3 IP DA Match is disabled : 0x0 1h = Layer 3 IP DA Match is enabled : 0x1
3	L3SAIM2	R/W	0h	Layer 3 IP SA Inverse Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set. 0h = Layer 3 IP SA Inverse Match is disabled : 0x0 1h = Layer 3 IP SA Inverse Match is enabled : 0x1
2	L3SAM2	R/W	0h	Layer 3 IP SA Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering. 0h = Layer 3 IP SA Match is disabled : 0x0 1h = Layer 3 IP SA Match is enabled : 0x1
1	RESERVED	R	0h	Reserved.
0	L3PEN2	R/W	0h	Layer 3 Protocol Enable When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set. 0h = Layer 3 Protocol is disabled : 0x0 1h = Layer 3 Protocol is enabled : 0x1

43.7.3.158 MAC_Layer4_Address2 Register (Offset = 964h) [Reset = 0h]

MAC_Layer4_Address2 is shown in Figure 43-197 and described in Table 43-251.

Return to the Summary Table.

The Layer 4 Address 0 register and registers 580 through 667 are reserved (RO with default value) if Enable Layer 3 and Layer 4 Packet Filter option is not selected while configuring the core.
You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option while configuring the core. When you select this option, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform consecutive writes to same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.

Figure 43-197. MAC_Layer4_Address2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L4DP2																L4SP2															
R/W-0h																R/W-0h															

Table 43-251. MAC_Layer4_Address2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	L4DP2	R/W	0h	Layer 4 Destination Port Number Field When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.
15-0	L4SP2	R/W	0h	Layer 4 Source Port Number Field When the L4PEN0 bit is reset and the L4SPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4SPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.

43.7.3.159 MAC_Layer3_Addr0_Reg2 Register (Offset = 970h) [Reset = 0h]

MAC_Layer3_Addr0_Reg2 is shown in Figure 43-198 and described in Table 43-252.

Return to the Summary Table.

For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Figure 43-198. MAC_Layer3_Addr0_Reg2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A02																															
R/W-0h																															

Table 43-252. MAC_Layer3_Addr0_Reg2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	L3A02	R/W	0h	Layer 3 Address 0 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

43.7.3.160 MAC_Layer3_Addr1_Reg2 Register (Offset = 974h) [Reset = 0h]

MAC_Layer3_Addr1_Reg2 is shown in Figure 43-199 and described in Table 43-253.

Return to the Summary Table.

For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Figure 43-199. MAC_Layer3_Addr1_Reg2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A12																															
R/W-0h																															

Table 43-253. MAC_Layer3_Addr1_Reg2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	L3A12	R/W	0h	Layer 3 Address 1 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

43.7.3.161 MAC_Layer3_Addr2_Reg2 Register (Offset = 978h) [Reset = 0h]

MAC_Layer3_Addr2_Reg2 is shown in Figure 43-200 and described in Table 43-254.

Return to the Summary Table.

The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Figure 43-200. MAC_Layer3_Addr2_Reg2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A22																															
R/W-0h																															

Table 43-254. MAC_Layer3_Addr2_Reg2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	L3A22	R/W	0h	Layer 3 Address 2 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

43.7.3.162 MAC_Layer3_Addr3_Reg2 Register (Offset = 97Ch) [Reset = 0h]

MAC_Layer3_Addr3_Reg2 is shown in Figure 43-201 and described in Table 43-255.

Return to the Summary Table.

The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Figure 43-201. MAC_Layer3_Addr3_Reg2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A32																															
R/W-0h																															

Table 43-255. MAC_Layer3_Addr3_Reg2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	L3A32	R/W	0h	Layer 3 Address 3 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

43.7.3.163 MAC_L3_L4_Control3 Register (Offset = 990h) [Reset = 0h]

MAC_L3_L4_Control3 is shown in Figure 43-202 and described in Table 43-256.

Return to the Summary Table.

The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.

Figure 43-202. MAC_L3_L4_Control3 Register

31	30	29	28	27	26	25	24
RESERVED			DMCHEN3	RESERVED			DMCHN3
R-0h			R/W-0h	R-0h			R/W-0h
23	22	21	20	19	18	17	16
RESERVED		L4DPIM3	L4DPM3	L4SPIM3	L4SPM3	RESERVED	L4PEN3
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h
15	14	13	12	11	10	9	8
L3HDBM3					L3HSBM3		
R/W-0h					R/W-0h		
7	6	5	4	3	2	1	0
L3HSBM3		L3DAIM3	L3DAM3	L3SAIM3	L3SAM3	RESERVED	L3PEN3
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h

Table 43-256. MAC_L3_L4_Control3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved.
28	DMCHEN3	R/W	0h	DMA Channel Select Enable When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When this bit is reset, the DMA channel is not decided by this filter. 0h = DMA Channel Select is disabled : 0x0 1h = DMA Channel Select is enabled : 0x1
27-25	RESERVED	R	0h	Reserved.
24	DMCHN3	R/W	0h	DMA Channel Number When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration.
23-22	RESERVED	R	0h	Reserved.
21	L4DPIM3	R/W	0h	Layer 4 Destination Port Inverse Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high. 0h = Layer 4 Destination Port Inverse Match is disabled : 0x0 1h = Layer 4 Destination Port Inverse Match is enabled : 0x1
20	L4DPM3	R/W	0h	Layer 4 Destination Port Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching. 0h = Layer 4 Destination Port Match is disabled : 0x0 1h = Layer 4 Destination Port Match is enabled : 0x1

Table 43-256. MAC_L3_L4_Control3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	L4SPIM3	R/W	0h	Layer 4 Source Port Inverse Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high. 0h = Layer 4 Source Port Inverse Match is disabled : 0x0 1h = Layer 4 Source Port Inverse Match is enabled : 0x1
18	L4SPM3	R/W	0h	Layer 4 Source Port Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching. 0h = Layer 4 Source Port Match is disabled : 0x0 1h = Layer 4 Source Port Match is enabled : 0x1
17	RESERVED	R	0h	Reserved.
16	L4PEN3	R/W	0h	Layer 4 Protocol Enable When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set. 0h = Layer 4 Protocol is disabled : 0x0 1h = Layer 4 Protocol is enabled : 0x1
15-11	L3HDBM3	R/W	0h	Layer 3 IP DA Higher Bits Match IPv4 Packets: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: - 0: No bits are masked. - 1: LSb[0] is masked. - 2: Two LSbs [1:0] are masked - .. - 127: All bits except MSb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set.
10-6	L3HSBM3	R/W	0h	Layer 3 IP SA Higher Bits Match IPv4 Packets: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: - 0: No bits are masked. - 1: LSb[0] is masked - 2: Two LSbs [1:0] are masked - .. - 31: All bits except MSb are masked. IPv6 Packets: This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high.

Table 43-256. MAC_L3_L4_Control3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	L3DAIM3	R/W	0h	Layer 3 IP DA Inverse Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high. 0h = Layer 3 IP DA Inverse Match is disabled : 0x0 1h = Layer 3 IP DA Inverse Match is enabled : 0x1
4	L3DAM3	R/W	0h	Layer 3 IP DA Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering. 0h = Layer 3 IP DA Match is disabled : 0x0 1h = Layer 3 IP DA Match is enabled : 0x1
3	L3SAIM3	R/W	0h	Layer 3 IP SA Inverse Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set. 0h = Layer 3 IP SA Inverse Match is disabled : 0x0 1h = Layer 3 IP SA Inverse Match is enabled : 0x1
2	L3SAM3	R/W	0h	Layer 3 IP SA Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When the L3PEN0 bit is set, you should set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering. 0h = Layer 3 IP SA Match is disabled : 0x0 1h = Layer 3 IP SA Match is enabled : 0x1
1	RESERVED	R	0h	Reserved.
0	L3PEN3	R/W	0h	Layer 3 Protocol Enable When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set. 0h = Layer 3 Protocol is disabled : 0x0 1h = Layer 3 Protocol is enabled : 0x1

43.7.3.164 MAC_Layer4_Address3 Register (Offset = 994h) [Reset = 0h]

MAC_Layer4_Address3 is shown in Figure 43-203 and described in Table 43-257.

Return to the Summary Table.

The Layer 4 Address 0 register and registers 580 through 667 are reserved (RO with default value) if Enable Layer 3 and Layer 4 Packet Filter option is not selected while configuring the core.
You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option while configuring the core. When you select this option, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform consecutive writes to same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.

Figure 43-203. MAC_Layer4_Address3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L4DP3																L4SP3															
R/W-0h																R/W-0h															

Table 43-257. MAC_Layer4_Address3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	L4DP3	R/W	0h	Layer 4 Destination Port Number Field When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets.
15-0	L4SP3	R/W	0h	Layer 4 Source Port Number Field When the L4PEN0 bit is reset and the L4SPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4SPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets.

43.7.3.165 MAC_Layer3_Addr0_Reg3 Register (Offset = 9A0h) [Reset = 0h]

MAC_Layer3_Addr0_Reg3 is shown in Figure 43-204 and described in Table 43-258.

Return to the Summary Table.

For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Figure 43-204. MAC_Layer3_Addr0_Reg3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A03																															
R/W-0h																															

Table 43-258. MAC_Layer3_Addr0_Reg3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	L3A03	R/W	0h	Layer 3 Address 0 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets.

43.7.3.166 MAC_Layer3_Addr1_Reg3 Register (Offset = 9A4h) [Reset = 0h]

MAC_Layer3_Addr1_Reg3 is shown in Figure 43-205 and described in Table 43-259.

Return to the Summary Table.

For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Figure 43-205. MAC_Layer3_Addr1_Reg3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A13																															
R/W-0h																															

Table 43-259. MAC_Layer3_Addr1_Reg3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	L3A13	R/W	0h	Layer 3 Address 1 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets.

43.7.3.167 MAC_Layer3_Addr2_Reg3 Register (Offset = 9A8h) [Reset = 0h]

MAC_Layer3_Addr2_Reg3 is shown in Figure 43-206 and described in Table 43-260.

Return to the Summary Table.

The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.

Figure 43-206. MAC_Layer3_Addr2_Reg3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A23																															
R/W-0h																															

Table 43-260. MAC_Layer3_Addr2_Reg3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	L3A23	R/W	0h	Layer 3 Address 2 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

43.7.3.168 MAC_Layer3_Addr3_Reg3 Register (Offset = 9ACh) [Reset = 0h]

MAC_Layer3_Addr3_Reg3 is shown in Figure 43-207 and described in Table 43-261.

Return to the Summary Table.

The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.

Figure 43-207. MAC_Layer3_Addr3_Reg3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L3A33																															
R/W-0h																															

Table 43-261. MAC_Layer3_Addr3_Reg3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	L3A33	R/W	0h	Layer 3 Address 3 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used.

43.7.3.169 MAC_Timestamp_Control Register (Offset = B00h) [Reset = 2000h]

MAC_Timestamp_Control is shown in Figure 43-208 and described in Table 43-262.

Return to the Summary Table.

This register controls the operation of the System Time generator and processing of PTP packets for timestamping in the Receiver.

Figure 43-208. MAC_Timestamp_Control Register

31	30	29	28	27	26	25	24
RESERVED			AV8021ASMEN	RESERVED			TXTSSTSM
R-0h			R/W-0h	R-0h			R/W-0h
23	22	21	20	19	18	17	16
RESERVED			ESTI	CSC	TSENMACADDR	SNAPTYPSEL	
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
TSMSTRENA	TSEVNTENA	TSIPV4ENA	TSIPV6ENA	TSIPENA	TSVER2ENA	TSCTRLSSR	TSENALL
R/W-0h	R/W-0h	R/W-1h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED		TSADDREG	RESERVED	TSUPDT	TSINIT	TSCFUPDT	TSENA
R-0h		R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 43-262. MAC_Timestamp_Control Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved.
28	AV8021ASMEN	R/W	0h	AV 802.1AS Mode Enable When this bit is set, the MAC processes only untagged PTP over Ethernet packets for providing PTP status and capturing timestamp snapshots, that is, IEEE 802.1AS mode of operation. When PTP offload feature is enabled, for the purpose of PTP offload, the transport specific field in the PTP header is generated and checked based on the value of this bit. 0h = AV 802.1AS Mode is disabled : 0x0 1h = AV 802.1AS Mode is enabled : 0x1
27-25	RESERVED	R	0h	Reserved.
24	TXTSSTSM	R/W	0h	Transmit Timestamp Status Mode When this bit is set, the MAC overwrites the earlier transmit timestamp status even if it is not read by the software. The MAC indicates this by setting the TXTSSMIS bit of the MAC_Tx_Timestamp_Status_Nanoseconds register. When this bit is reset, the MAC ignores the timestamp status of current packet if the timestamp status of previous packet is not read by the software. The MAC indicates this by setting the TXTSSMIS bit of the MAC_Tx_Timestamp_Status_Nanoseconds register. 0h = Transmit Timestamp Status Mode is disabled : 0x0 1h = Transmit Timestamp Status Mode is enabled : 0x1
23-21	RESERVED	R	0h	Reserved.

Table 43-262. MAC_Timestamp_Control Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	ESTI	R/W	0h	External System Time Input When this bit is set, the MAC uses the external 64-bit reference System Time input for the following: - To take the timestamp provided as status - To insert the timestamp in transmit PTP packets when One-step Timestamp or Timestamp Offload feature is enabled. When this bit is reset, the MAC uses the internal reference System Time. 0h = External System Time Input is disabled : 0x0 1h = External System Time Input is enabled : 0x1
19	CSC	R/W	0h	Enable checksum correction during OST for PTP over UDP/IPv4 packets When this bit is set, the last two bytes of PTP message sent over UDP/IPv4 is updated to keep the UDP checksum correct, for changes made to origin timestamp and/or correction field as part of one step timestamp operation. The application shall form the packet with these two dummy bytes. When reset, no updates are done to keep the UDP checksum correct. The application shall form the packet with UDP checksum set to 0. 0h = checksum correction during OST for PTP over UDP/IPv4 packets is disabled : 0x0 1h = checksum correction during OST for PTP over UDP/IPv4 packets is enabled : 0x1
18	TSENMACADDR	R/W	0h	Enable MAC Address for PTP Packet Filtering When this bit is set, the DA MAC address (that matches any MAC Address register) is used to filter the PTP packets when PTP is directly sent over Ethernet. When this bit is set, received PTP packets with DA containing a special multicast or unicast address that matches the one programmed in MAC address registers are considered for processing as indicated below, when PTP is directly sent over Ethernet. For normal time stamping operation, MAC address registers 0 to 31 is considered for unicast destination address matching. For PTP offload, only MAC address register 0 is considered for unicast destination address matching. 0h = MAC Address for PTP Packet Filtering is disabled : 0x0 1h = MAC Address for PTP Packet Filtering is enabled : 0x1
17-16	SNAPTYPSEL	R/W	0h	Select PTP packets for Taking Snapshots These bits, along with Bits 15 and 14, decide the set of PTP packet types for which snapshot needs to be taken. The encoding is given in Timestamp Snapshot Dependency on Register Bits Table.
15	TSMSTRENA	R/W	0h	Enable Snapshot for Messages Relevant to Master When this bit is set, the snapshot is taken only for the messages that are relevant to the master node. Otherwise, the snapshot is taken for the messages relevant to the slave node. 0h = Snapshot for Messages Relevant to Master is disabled : 0x0 1h = Snapshot for Messages Relevant to Master is enabled : 0x1
14	TSEVNTENA	R/W	0h	Enable Timestamp Snapshot for Event Messages When this bit is set, the timestamp snapshot is taken only for event messages (SYNC, Delay_Req, Pdelay_Req, or Pdelay_Resp). When this bit is reset, the snapshot is taken for all messages except Announce, Management, and Signaling. For more information about the timestamp snapshots, see Timestamp Snapshot Dependency on Register Bits Table. 0h = Timestamp Snapshot for Event Messages is disabled : 0x0 1h = Timestamp Snapshot for Event Messages is enabled : 0x1

Table 43-262. MAC_Timestamp_Control Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	TSIPV4ENA	R/W	1h	Enable Processing of PTP Packets Sent over IPv4-UDP When this bit is set, the MAC receiver processes the PTP packets encapsulated in IPv4-UDP packets. When this bit is reset, the MAC ignores the PTP transported over IPv4-UDP packets. This bit is set by default. 0h = Processing of PTP Packets Sent over IPv4-UDP is disabled : 0x0 1h = Processing of PTP Packets Sent over IPv4-UDP is enabled : 0x1
12	TSIPV6ENA	R/W	0h	Enable Processing of PTP Packets Sent over IPv6-UDP When this bit is set, the MAC receiver processes the PTP packets encapsulated in IPv6-UDP packets. When this bit is clear, the MAC ignores the PTP transported over IPv6-UDP packets. 0h = Processing of PTP Packets Sent over IPv6-UDP is disabled : 0x0 1h = Processing of PTP Packets Sent over IPv6-UDP is enabled : 0x1
11	TSIPENA	R/W	0h	Enable Processing of PTP over Ethernet Packets When this bit is set, the MAC receiver processes the PTP packets encapsulated directly in the Ethernet packets. When this bit is reset, the MAC ignores the PTP over Ethernet packets. 0h = Processing of PTP over Ethernet Packets is disabled : 0x0 1h = Processing of PTP over Ethernet Packets is enabled : 0x1
10	TSVER2ENA	R/W	0h	Enable PTP Packet Processing for Version 2 Format When this bit is set, the IEEE 1588 version 2 format is used to process the PTP packets. When this bit is reset, the IEEE 1588 version 1 format is used to process the PTP packets. The IEEE 1588 formats are described in 'PTP Processing and Control'. 0h = PTP Packet Processing for Version 2 Format is disabled : 0x0 1h = PTP Packet Processing for Version 2 Format is enabled : 0x1
9	TSCTRLSSR	R/W	0h	Timestamp Digital or Binary Rollover Control When this bit is set, the Timestamp Low register rolls over after 0x3B9A_C9FF value (that is, 1 nanosecond accuracy) and increments the timestamp (High) seconds. When this bit is reset, the rollover value of sub-second register is 0x7FFF_FFFF. The sub-second increment must be programmed correctly depending on the PTP reference clock frequency and the value of this bit. 0h = Timestamp Digital or Binary Rollover Control is disabled : 0x0 1h = Timestamp Digital or Binary Rollover Control is enabled : 0x1
8	TSENALL	R/W	0h	Enable Timestamp for All Packets When this bit is set, the timestamp snapshot is enabled for all packets received by the MAC. 0h = Timestamp for All Packets disabled : 0x0 1h = Timestamp for All Packets enabled : 0x1
7-6	RESERVED	R	0h	Reserved.
5	TSADDREG	R/W	0h	Update Addend Register When this bit is set, the content of the Timestamp Addend register is updated in the PTP block for fine correction. This bit is cleared when the update is complete. This bit should be zero before it is set. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0h = Addend Register is not updated : 0x0 1h = Addend Register is updated : 0x1
4	RESERVED	R	0h	Reserved.

Table 43-262. MAC_Timestamp_Control Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	TSUPDT	R/W	0h	Update Timestamp When this bit is set, the system time is updated (added or subtracted) with the value specified in MAC_System_Time_Seconds_Update and MAC_System_Time_Nanoseconds_Update. This bit should be zero before updating it. This bit is reset when the update is complete in hardware. The Timestamp Higher Word register (if enabled during core configuration) is not updated. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0h = Timestamp is not updated : 0x0 1h = Timestamp is updated : 0x1
2	TSINIT	R/W	0h	Initialize Timestamp When this bit is set, the system time is initialized (overwritten) with the value specified in the MAC Register 80 (System Time Seconds Update Register) and MAC Register 81 (System Time Nanoseconds Update Register). This bit should be zero before it is updated. This bit is reset when the initialization is complete. The Timestamp Higher Word register (if enabled during core configuration) can only be initialized. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0h = Timestamp is not initialized : 0x0 1h = Timestamp is initialized : 0x1
1	TSCFUPDT	R/W	0h	Fine or Coarse Timestamp Update When this bit is set, the Fine method is used to update system timestamp. When this bit is reset, Coarse method is used to update the system timestamp. 0h = Coarse method is used to update system timestamp : 0x0 1h = Fine method is used to update system timestamp : 0x1
0	TSENA	R/W	0h	Enable Timestamp When this bit is set, the timestamp is added for Transmit and Receive packets. When disabled, timestamp is not added for transmit and receive packets and the Timestamp Generator is also suspended. You need to initialize the Timestamp (system time) after enabling this mode. On the Receive side, the MAC processes the 1588 packets only if this bit is set. 0h = Timestamp is disabled : 0x0 1h = Timestamp is enabled : 0x1

43.7.3.170 MAC_Sub_Second_Increment Register (Offset = B04h) [Reset = 0h]

MAC_Sub_Second_Increment is shown in Figure 43-209 and described in Table 43-263.

Return to the Summary Table.

This register specifies the value to be added to the internal system time register every cycle of clk_ptp_ref_i clock.

Figure 43-209. MAC_Sub_Second_Increment Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								SSINC								SNSINC								RESERVED							
R-0h								R/W-0h								R/W-0h								R-0h							

Table 43-263. MAC_Sub_Second_Increment Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved.
23-16	SSINC	R/W	0h	Sub-second Increment Value The value programmed in this field is accumulated every clock cycle (of clk_ptp_i) with the contents of the sub-second register. For example, when the PTP clock is 50 MHz (period is 20 ns), you should program 20 (0x14) when the System Time Nanoseconds register has an accuracy of 1 ns [Bit 9 (TSCTRLSSR) is set in MAC_Timestamp_Control]. When TSCTRLSSR is clear, the Nanoseconds register has a resolution of ~0.465 ns. In this case, you should program a value of 43 (0x2B) which is derived by 20 ns/0.465.
15-8	SNSINC	R/W	0h	Sub-nanosecond Increment Value This field contains the sub-nanosecond increment value, represented in nanoseconds multiplied by 2^8. This value is accumulated with the sub-nanoseconds field of the subsecond register. For example, when TSCTRLSSR field in the MAC_Timestamp_Control register is set. and if the required increment is 5.3ns, then SSINC should be 0x05 and SNSINC should be 0x4C.
7-0	RESERVED	R	0h	Reserved.

43.7.3.171 MAC_System_Time_Seconds Register (Offset = B08h) [Reset = 0h]

MAC_System_Time_Seconds is shown in Figure 43-210 and described in Table 43-264.

Return to the Summary Table.

The System Time Seconds register, along with System Time Nanoseconds register, indicates the current value of the system time maintained by the MAC. Though it is updated on a continuous basis, there is some delay from the actual time because of clock domain transfer latencies (from clk_ptp_ref_i to CSR clock).

Figure 43-210. MAC_System_Time_Seconds Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSS																															
R-0h																															

Table 43-264. MAC_System_Time_Seconds Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TSS	R	0h	Timestamp Second The value in this field indicates the current value in seconds of the System Time maintained by the MAC.

43.7.3.172 MAC_System_Time_Nanoseconds Register (Offset = B0Ch) [Reset = 0h]

MAC_System_Time_Nanoseconds is shown in Figure 43-211 and described in Table 43-265.

Return to the Summary Table.

The System Time Nanoseconds register, along with System Time Seconds register, indicates the current value of the system time maintained by the MAC.

Figure 43-211. MAC_System_Time_Nanoseconds Register

31	30	29	28	27	26	25	24
RESERVED	TSSS						
R-0h	R-0h						
23	22	21	20	19	18	17	16
TSSS							
R-0h							
15	14	13	12	11	10	9	8
TSSS							
R-0h							
7	6	5	4	3	2	1	0
TSSS							
R-0h							

Table 43-265. MAC_System_Time_Nanoseconds Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved.
30-0	TSSS	R	0h	Timestamp Sub Seconds The value in this field has the sub-second representation of time, with an accuracy of 0.46 ns. When Bit 9 is set in MAC_Timestamp_Control, each bit represents 1 ns. The maximum value is 0x3B9A_C9FF after which it rolls-over to zero.

43.7.3.173 MAC_System_Time_Seconds_Update Register (Offset = B10h) [Reset = 0h]

MAC_System_Time_Seconds_Update is shown in Figure 43-212 and described in Table 43-266.

Return to the Summary Table.

The System Time Seconds Update register, along with the System Time Nanoseconds Update register, initializes or updates the system time maintained by the MAC. You must write both registers before setting the TSINIT or TSUPDT bits in EMAC_REGS/EQOS_MAC/MAC_Timestamp_Control.

Figure 43-212. MAC_System_Time_Seconds_Update Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSS																															
R/W-0h																															

Table 43-266. MAC_System_Time_Seconds_Update Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TSS	R/W	0h	Timestamp Seconds The value in this field is the seconds part of the update. When ADDSUB is reset, this field must be programmed with the seconds part of the update value. When ADDSUB is set, this field must be programmed with the complement of the seconds part of the update value. For example, if 2.000000001 seconds need to be subtracted from the system time, the TSS field in the MAC_Timestamp_Seconds_Update register must be 0xFFFF_FFFE (that is, 2^{32} - 2).

43.7.3.174 MAC_System_Time_Nanoseconds_Update Register (Offset = B14h) [Reset = 0h]

MAC_System_Time_Nanoseconds_Update is shown in Figure 43-213 and described in Table 43-267.

Return to the Summary Table.

MAC System Time Nanoseconds Update register.

Figure 43-213. MAC_System_Time_Nanoseconds_Update Register

31	30	29	28	27	26	25	24
ADDSUB	TSSS						
R/W-0h	R/W-0h						
23	22	21	20	19	18	17	16
TSSS							
R/W-0h							
15	14	13	12	11	10	9	8
TSSS							
R/W-0h							
7	6	5	4	3	2	1	0
TSSS							
R/W-0h							

Table 43-267. MAC_System_Time_Nanoseconds_Update Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ADDSUB	R/W	0h	Add or Subtract Time When this bit is set, the time value is subtracted with the contents of the update register. When this bit is reset, the time value is added with the contents of the update register. 0h = Add time : 0x0 1h = Subtract time : 0x1
30-0	TSSS	R/W	0h	Timestamp Sub Seconds The value in this field is the sub-seconds part of the update. When ADDSUB is reset, this field must be programmed with the sub-seconds part of the update value, with an accuracy based on the TSCTRLSSR bit of the MAC_Timestamp_Control register. When ADDSUB is set, this field must be programmed with the complement of the sub-seconds part of the update value as described below. When TSCTRLSSR bit in MAC_Timestamp_Control is set, the programmed value must be 10^9 - <sub-second value>. When TSCTRLSSR bit in MAC_Timestamp_Control is reset, the programmed value must be 2^{31} - <sub-second_value>. When the TSCTRLSSR bit is reset in the MAC_Timestamp_Control register, each bit represents an accuracy of 0.46 ns. When the TSCTRLSSR bit is set in the MAC_Timestamp_Control register, each bit represents 1 ns and the programmed value should not exceed 0x3B9A_C9FF. For example, if 2.000000001 seconds need to be subtracted from the system time, then the TSSS field in the MAC_Timestamp_Nanoseconds_Update register must be 0x7FFF_FFFF (that is, 2^{31} - 1), when TSCTRLSSR bit in MAC_Timestamp_Control is reset and 0x3B9A_C9FF (that is, 10^9 - 1), when TSCTRLSSR bit in MAC_Timestamp_Control is set.

43.7.3.175 MAC_Timestamp_Addend Register (Offset = B18h) [Reset = 0h]

MAC_Timestamp_Addend is shown in Figure 43-214 and described in Table 43-268.

Return to the Summary Table.

Timestamp Addend register. This register value is used only when the system time is configured for Fine Update mode (TSCFUPDT bit in the MAC_Timestamp_Control register). The content of this register is added to a 32-bit accumulator in every clock cycle (of clk_ptp_ref_i) and the system time is updated whenever the accumulator overflows.

Figure 43-214. MAC_Timestamp_Addend Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSAR																															
R/W-0h																															

Table 43-268. MAC_Timestamp_Addend Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TSAR	R/W	0h	Timestamp Addend Register This field indicates the 32-bit time value to be added to the Accumulator register to achieve time synchronization.

43.7.3.176 MAC_System_Time_Higher_Word_Seconds Register (Offset = B1Ch) [Reset = 0h]

MAC_System_Time_Higher_Word_Seconds is shown in Figure 43-215 and described in Table 43-269.

Return to the Summary Table.

System Time - Higher Word Seconds register.

Figure 43-215. MAC_System_Time_Higher_Word_Seconds Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																TSHWR															
R-0h																R/W-0h															

Table 43-269. MAC_System_Time_Higher_Word_Seconds Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved.
15-0	TSHWR	R/W	0h	Timestamp Higher Word Register This field contains the most-significant 16-bits of timestamp seconds value. This register is optional. You can add this register by selecting the Add IEEE 1588 Higher Word Register option. This register is directly written to initialize the value and it is incremented when there is an overflow from 32-bits of the System Time Seconds register. Access restriction applies. Updated based on the event. Setting 1 sets. Setting 0 clears.

43.7.3.177 MAC_Timestamp_Status Register (Offset = B20h) [Reset = 0h]

MAC_Timestamp_Status is shown in Figure 43-216 and described in Table 43-270.

Return to the Summary Table.

Timestamp Status register. All bits except Bits[27:25] gets cleared when the application reads this register.

Figure 43-216. MAC_Timestamp_Status Register

31	30	29	28	27	26	25	24
RESERVED				ATSNS			ATSSTM
R-0h				R-0h			R-0h
23	22	21	20	19	18	17	16
RESERVED				ATSSTN			
R-0h				R-0h			
15	14	13	12	11	10	9	8
TXTSSIS	RESERVED					TSTRGTERR3	TSTARGT3
R-0h	R-0h					R-0h	R-0h
7	6	5	4	3	2	1	0
TSTRGTERR2	TSTARGT2	TSTRGTERR1	TSTARGT1	TSTRGTERR0	AUXTSTRIG	TSTARGT0	TSSOVF
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 43-270. MAC_Timestamp_Status Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved.
29-25	ATSNS	R	0h	Number of Auxiliary Timestamp Snapshots This field indicates the number of Snapshots available in the FIFO. A value equal to the selected depth of FIFO (4, 8, or 16) indicates that the Auxiliary Snapshot FIFO is full. These bits are cleared (to 00000) when the Auxiliary snapshot FIFO clear bit is set. This bit is valid only if the Add IEEE 1588 Auxiliary Snapshot option is selected.
24	ATSSTM	R	0h	Auxiliary Timestamp Snapshot Trigger Missed This bit is set when the Auxiliary timestamp snapshot FIFO is full and external trigger was set. This indicates that the latest snapshot is not stored in the FIFO. This bit is valid only if the Add IEEE 1588 Auxiliary Snapshot option is selected. 0h = Auxiliary Timestamp Snapshot Trigger Missed status not detected : 0x0 1h = Auxiliary Timestamp Snapshot Trigger Missed status detected : 0x1
23-20	RESERVED	R	0h	Reserved.
19-16	ATSSTN	R	0h	Auxiliary Timestamp Snapshot Trigger Identifier These bits identify the Auxiliary trigger inputs for which the timestamp available in the Auxiliary Snapshot Register is applicable. When more than one bit is set at the same time, it means that corresponding auxiliary triggers were sampled at the same clock. These bits are applicable only if the number of Auxiliary snapshots is more than one. One bit is assigned for each trigger as shown in the following list: - Bit 16: Auxiliary trigger 0 - Bit 17: Auxiliary trigger 1 - Bit 18: Auxiliary trigger 2 - Bit 19: Auxiliary trigger 3 The software can read this register to find the triggers that are set when the timestamp is taken. Access restriction applies. Clears on read. Self-set to 1 on internal event.

Table 43-270. MAC_Timestamp_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15	TXTSSIS	R	0h	Tx Timestamp Status Interrupt Status In non-EQOS_CORE configurations when drop transmit status is enabled in MTL, this bit is set when the captured transmit timestamp is updated in the MAC_Tx_Timestamp_Status_Nanoseconds and MAC_Tx_Timestamp_Status_Seconds registers. When PTP offload feature is enabled, this bit is set when the captured transmit timestamp is updated in the MAC_Tx_Timestamp_Status_Nanoseconds and MAC_Tx_Timestamp_Status_Seconds registers, for PTO generated Delay Request and Pdelay request packets. This bit is cleared when the MAC_Tx_Timestamp_Status_Seconds register is read (or write to MAC_Tx_Timestamp_Status_Seconds register when RCWE bit of MAC_CSR_SW_Ctrl register is set). 0h = Tx Timestamp Status Interrupt status not detected : 0x0 1h = Tx Timestamp Status Interrupt status detected : 0x1
14-10	RESERVED	R	0h	Reserved.
9	TSTRGTERR3	R	0h	Timestamp Target Time Error This bit is set when the latest target time programmed in the MAC_PPS3_Target_Time_Seconds and MAC_PPS3_Target_Time_Nanoseconds registers elapses. This bit is cleared when the application reads this bit. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = Timestamp Target Time Error status not detected : 0x0 1h = Timestamp Target Time Error status detected : 0x1
8	TSTARGT3	R	0h	Timestamp Target Time Reached for Target Time PPS3 When this bit is set, it indicates that the value of system time is greater than or equal to the value specified in the MAC_PPS3_Target_Time_Seconds and MAC_PPS3_Target_Time_Nanoseconds registers. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = Timestamp Target Time Reached for Target Time PPS3 status not detected : 0x0 1h = Timestamp Target Time Reached for Target Time PPS3 status detected : 0x1
7	TSTRGTERR2	R	0h	Timestamp Target Time Error This bit is set when the latest target time programmed in the MAC_PPS2_Target_Time_Seconds and MAC_PPS2_Target_Time_Nanoseconds registers elapses. This bit is cleared when the application reads this bit. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = Timestamp Target Time Error status not detected : 0x0 1h = Timestamp Target Time Error status detected : 0x1
6	TSTARGT2	R	0h	Timestamp Target Time Reached for Target Time PPS2 When set, this bit indicates that the value of system time is greater than or equal to the value specified in the MAC_PPS2_Target_Time_Seconds and MAC_PPS2_Target_Time_Nanoseconds registers. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = Timestamp Target Time Reached for Target Time PPS2 status not detected : 0x0 1h = Timestamp Target Time Reached for Target Time PPS2 status detected : 0x1

Table 43-270. MAC_Timestamp_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	TSTRGTERR1	R	0h	Timestamp Target Time Error This bit is set when the latest target time programmed in the MAC_PPS1_Target_Time_Seconds and MAC_PPS1_Target_Time_Nanoseconds registers elapses. This bit is cleared when the application reads this bit. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = Timestamp Target Time Error status not detected : 0x0 1h = Timestamp Target Time Error status detected : 0x1
4	TSTARGT1	R	0h	Timestamp Target Time Reached for Target Time PPS1 When set, this bit indicates that the value of system time is greater than or equal to the value specified in the MAC_PPS1_Target_Time_Seconds and MAC_PPS1_Target_Time_Nanoseconds registers. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = Timestamp Target Time Reached for Target Time PPS1 status not detected : 0x0 1h = Timestamp Target Time Reached for Target Time PPS1 status detected : 0x1
3	TSTRGTERR0	R	0h	Timestamp Target Time Error This bit is set when the latest target time programmed in the MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds registers elapses. This bit is cleared when the application reads this bit. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = Timestamp Target Time Error status not detected : 0x0 1h = Timestamp Target Time Error status detected : 0x1
2	AUXTSTRIG	R	0h	Auxiliary Timestamp Trigger Snapshot This bit is set high when the auxiliary snapshot is written to the FIFO. This bit is valid only if the Add IEEE 1588 Auxiliary Snapshot option is selected. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = Auxiliary Timestamp Trigger Snapshot status not detected : 0x0 1h = Auxiliary Timestamp Trigger Snapshot status detected : 0x1
1	TSTARGT0	R	0h	Timestamp Target Time Reached When set, this bit indicates that the value of system time is greater than or equal to the value specified in the MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds registers. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = Timestamp Target Time Reached status not detected : 0x0 1h = Timestamp Target Time Reached status detected : 0x1
0	TSSOVF	R	0h	Timestamp Seconds Overflow When this bit is set, it indicates that the seconds value of the timestamp (when supporting version 2 format) has overflowed beyond 32'hFFFF_FFFF. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. 0h = Timestamp Seconds Overflow status not detected : 0x0 1h = Timestamp Seconds Overflow status detected : 0x1

43.7.3.178 MAC_Tx_Timestamp_Status_Nanoseconds Register (Offset = B30h) [Reset = 0h]

MAC_Tx_Timestamp_Status_Nanoseconds is shown in Figure 43-217 and described in Table 43-271.

Return to the Summary Table.

This register contains the nanosecond part of timestamp captured for Transmit packets when Tx status is disabled.
The MAC_Tx_Timestamp_Status_Nanoseconds register, along with MAC_Tx_Timestamp_Status_Seconds, gives the 64-bit timestamp captured for the PTP packet successfully transmitted by the MAC. This value is considered to be read by the application when the last byte of MAC_Tx_Timestamp_Status_Nanoseconds is read. In the little-endian mode, this means when bits[31:24] are read
in big-endian mode, bits[7:0] are read.
If the application does not read these registers and timestamp of another packet is captured, then either the current timestamp is lost (overwritten) or the new timestamp is lost (dropped), depending on the setting of the TXTSSTSM bit of the MAC_Timestamp_Control register. The status bit TXTSC bit [15] in MAC_Timestamp_Status register is set whenever the MAC transmitter captures the timestamp.

Figure 43-217. MAC_Tx_Timestamp_Status_Nanoseconds Register

31	30	29	28	27	26	25	24
TXTSSMIS	TXTSSLO						
R-0h	R-0h						
23	22	21	20	19	18	17	16
TXTSSLO							
R-0h							
15	14	13	12	11	10	9	8
TXTSSLO							
R-0h							
7	6	5	4	3	2	1	0
TXTSSLO							
R-0h							

Table 43-271. MAC_Tx_Timestamp_Status_Nanoseconds Register Field Descriptions

Bit	Field	Type	Reset	Description
31	TXTSSMIS	R	0h	Transmit Timestamp Status Missed When this bit is set, it indicates one of the following: - The timestamp of the current packet is ignored if TXTSSTSM bit of the MAC_Timestamp_Control register is reset - The timestamp of the previous packet is overwritten with timestamp of the current packet if TXTSSTSM bit of the MAC_Timestamp_Control register is set. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = Transmit Timestamp Status Missed status not detected : 0x0 1h = Transmit Timestamp Status Missed status detected : 0x1
30-0	TXTSSLO	R	0h	Transmit Timestamp Status Low This field contains the 31 bits of the Nanoseconds field of the Transmit packet's captured timestamp.

43.7.3.179 MAC_Tx_Timestamp_Status_Seconds Register (Offset = B34h) [Reset = 0h]

MAC_Tx_Timestamp_Status_Seconds is shown in Figure 43-218 and described in Table 43-272.

Return to the Summary Table.

The register contains the higher 32 bits of the timestamp (in seconds) captured when a PTP packet is transmitted.

Figure 43-218. MAC_Tx_Timestamp_Status_Seconds Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXTSSHI																															
R-0h																															

Table 43-272. MAC_Tx_Timestamp_Status_Seconds Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXTSSHI	R	0h	Transmit Timestamp Status High This field contains the lower 32 bits of the Seconds field of Transmit packet's captured timestamp.

43.7.3.180 MAC_Auxiliary_Control Register (Offset = B40h) [Reset = 0h]

MAC_Auxiliary_Control is shown in Figure 43-219 and described in Table 43-273.

Return to the Summary Table.

The Auxiliary Timestamp Control register controls the Auxiliary Timestamp snapshot.

Figure 43-219. MAC_Auxiliary_Control Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	ATSEN1	ATSEN0	RESERVED			ATSFC
R-0h	R-0h	R/W-0h	R/W-0h	R-0h			R/W-0h

Table 43-273. MAC_Auxiliary_Control Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved.
7	RESERVED	R	0h	Reserved.
6	RESERVED	R	0h	Reserved.
5	ATSEN1	R/W	0h	Auxiliary Snapshot 1 Enable This bit controls the capturing of Auxiliary Snapshot Trigger 1. When this bit is set, the auxiliary snapshot of the event on ptp_aux_trig_i[1] input is enabled. When this bit is reset, the events on this input are ignored. 0h = Auxiliary Snapshot $i is disabled : 0x0 1h = Auxiliary Snapshot $i is enabled : 0x1
4	ATSEN0	R/W	0h	Auxiliary Snapshot 0 Enable This bit controls the capturing of Auxiliary Snapshot Trigger 0. When this bit is set, the auxiliary snapshot of the event on ptp_aux_trig_i[0] input is enabled. When this bit is reset, the events on this input are ignored. 0h = Auxiliary Snapshot $i is disabled : 0x0 1h = Auxiliary Snapshot $i is enabled : 0x1
3-1	RESERVED	R	0h	Reserved.
0	ATSFC	R/W	0h	Auxiliary Snapshot FIFO Clear When set, this bit resets the pointers of the Auxiliary Snapshot FIFO. This bit is cleared when the pointers are reset and the FIFO is empty. When this bit is high, the auxiliary snapshots are stored in the FIFO. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0h = Auxiliary Snapshot FIFO Clear is disabled : 0x0 1h = Auxiliary Snapshot FIFO Clear is enabled : 0x1

43.7.3.181 MAC_Auxiliary_Timestamp_Nanoseconds Register (Offset = B48h) [Reset = 0h]

MAC_Auxiliary_Timestamp_Nanoseconds is shown in Figure 43-220 and described in Table 43-274.

Return to the Summary Table.

The Auxiliary Timestamp Nanoseconds register, along with MAC_Auxiliary_Timestamp_Seconds, gives the 64-bit timestamp stored as auxiliary snapshot. These two registers form the read port of a 64-bit wide FIFO with a depth of 4, 8, or 16 as selected while configuring the core.
You can store multiple snapshots in this FIFO. Bits[29:25] in MAC_Timestamp_Status indicate the fill-level of the FIFO. The top of the FIFO is removed only when the last byte of MAC Register 91 (Auxiliary Timestamp - Seconds Register) is read. In the little-endian mode, this means when Bits[31:24] are read and in big-endian mode, Bits[7:0] are read.

Figure 43-220. MAC_Auxiliary_Timestamp_Nanoseconds Register

31	30	29	28	27	26	25	24
RESERVED	AUXTSLO						
R-0h	R-0h						
23	22	21	20	19	18	17	16
AUXTSLO							
R-0h							
15	14	13	12	11	10	9	8
AUXTSLO							
R-0h							
7	6	5	4	3	2	1	0
AUXTSLO							
R-0h							

Table 43-274. MAC_Auxiliary_Timestamp_Nanoseconds Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved.
30-0	AUXTSLO	R	0h	Auxiliary Timestamp Contains the lower 31 bits (nanoseconds field) of the auxiliary timestamp.

43.7.3.182 MAC_Auxiliary_Timestamp_Seconds Register (Offset = B4Ch) [Reset = 0h]

MAC_Auxiliary_Timestamp_Seconds is shown in Figure 43-221 and described in Table 43-275.

Return to the Summary Table.

The Auxiliary Timestamp - Seconds register contains the lower 32 bits of the Seconds field of the auxiliary timestamp register.

Figure 43-221. MAC_Auxiliary_Timestamp_Seconds Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AUXTSHI																															
R-0h																															

Table 43-275. MAC_Auxiliary_Timestamp_Seconds Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	AUXTSHI	R	0h	Auxiliary Timestamp Contains the lower 32 bits of the Seconds field of the auxiliary timestamp.

43.7.3.183 MAC_Timestamp_Ingress_Asym_Corr Register (Offset = B50h) [Reset = 0h]

MAC_Timestamp_Ingress_Asym_Corr is shown in Figure 43-222 and described in Table 43-276.

Return to the Summary Table.

The MAC Timestamp Ingress Asymmetry Correction register contains the Ingress Asymmetry Correction value to be used while updating correction field in PDelay_Resp PTP messages.

Figure 43-222. MAC_Timestamp_Ingress_Asym_Corr Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OSTIAC																															
R/W-0h																															

Table 43-276. MAC_Timestamp_Ingress_Asym_Corr Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	OSTIAC	R/W	0h	One-Step Timestamp Ingress Asymmetry Correction This field contains the ingress path asymmetry value to be added to correctionField of Pdelay_Resp PTP packet. The programmed value should be in units of nanoseconds and multiplied by 2^{16}. For example, 2.5 ns is represented as 0x00028000. The value can also be negative, which is represented in 2's complement form with bit 31 representing the sign bit.

43.7.3.184 MAC_Timestamp_Egress_Asym_Corr Register (Offset = B54h) [Reset = 0h]

MAC_Timestamp_Egress_Asym_Corr is shown in Figure 43-223 and described in Table 43-277.

Return to the Summary Table.

The MAC Timestamp Egress Asymmetry Correction register contains the Egress Asymmetry Correction value to be used while updating the correction field in PDelay_Req PTP messages.

Figure 43-223. MAC_Timestamp_Egress_Asym_Corr Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OSTEAC																															
R/W-0h																															

Table 43-277. MAC_Timestamp_Egress_Asym_Corr Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	OSTEAC	R/W	0h	One-Step Timestamp Egress Asymmetry Correction This field contains the egress path asymmetry value to be subtracted from correctionField of Pdelay_Resp PTP packet. The programmed value must be the negated value in units of nanoseconds multiplied by 2^{16}. For example, if the required correction is +2.5 ns, the programmed value must be 0xFFFD_8000, which is the 2's complement of 0x0002_8000(2.5 * 216). Similarly, if the required correction is -3.3 ns, the programmed value is 0x0003_4CCC (3.3 * 216).

43.7.3.185 MAC_Timestamp_Ingress_Corr_Nanosecond Register (Offset = B58h) [Reset = 0h]

MAC_Timestamp_Ingress_Corr_Nanosecond is shown in Figure 43-224 and described in Table 43-278.

Return to the Summary Table.

This register contains the correction value in nanoseconds to be used with the captured timestamp value in the ingress path.

Figure 43-224. MAC_Timestamp_Ingress_Corr_Nanosecond Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSIC																															
R/W-0h																															

Table 43-278. MAC_Timestamp_Ingress_Corr_Nanosecond Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TSIC	R/W	0h	Timestamp Ingress Correction This field contains the ingress path correction value as defined by the Ingress Correction expression.

43.7.3.186 MAC_Timestamp_Egress_Corr_Nanosecond Register (Offset = B5Ch) [Reset = 0h]

MAC_Timestamp_Egress_Corr_Nanosecond is shown in Figure 43-225 and described in Table 43-279.

Return to the Summary Table.

This register contains the correction value in nanoseconds to be used with the captured timestamp value in the egress path.

Figure 43-225. MAC_Timestamp_Egress_Corr_Nanosecond Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSEC																															
R/W-0h																															

Table 43-279. MAC_Timestamp_Egress_Corr_Nanosecond Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TSEC	R/W	0h	Timestamp Egress Correction This field contains the nanoseconds part of the egress path correction value as defined by the Egress Correction expression.

43.7.3.187 MAC_Timestamp_Ingress_Corr_Subnanosec Register (Offset = B60h) [Reset = 0h]

MAC_Timestamp_Ingress_Corr_Subnanosec is shown in Figure 43-226 and described in Table 43-280.

Return to the Summary Table.

This register contains the sub-nanosecond part of the correction value to be used with the captured timestamp value, for ingress direction.

Figure 43-226. MAC_Timestamp_Ingress_Corr_Subnanosec Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																TSICSNS								RESERVED							
R-0h																R/W-0h								R-0h							

Table 43-280. MAC_Timestamp_Ingress_Corr_Subnanosec Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved.
15-8	TSICSNS	R/W	0h	Timestamp Ingress Correction, sub-nanoseconds This field contains the sub-nanoseconds part of the ingress path correction value as defined by the "Ingress Correction" expression.
7-0	RESERVED	R	0h	Reserved.

43.7.3.188 MAC_Timestamp_Egress_Corr_Subnanosec Register (Offset = B64h) [Reset = 0h]

MAC_Timestamp_Egress_Corr_Subnanosec is shown in Figure 43-227 and described in Table 43-281.

Return to the Summary Table.

This register contains the sub-nanosecond part of the correction value to be used with the captured timestamp value, for egress direction.

Figure 43-227. MAC_Timestamp_Egress_Corr_Subnanosec Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																TSECSNS								RESERVED							
R-0h																R/W-0h								R-0h							

Table 43-281. MAC_Timestamp_Egress_Corr_Subnanosec Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved.
15-8	TSECSNS	R/W	0h	Timestamp Egress Correction, sub-nanoseconds This field contains the sub-nanoseconds part of the egress path correction value as defined by the "Egress Correction" expression.
7-0	RESERVED	R	0h	Reserved.

43.7.3.189 MAC_PPS_Control Register (Offset = B70h) [Reset = 0h]

MAC_PPS_Control is shown in Figure 43-228 and described in Table 43-282.

Return to the Summary Table.

PPS Control register.
Bits[30:24] of this register are valid only when four Flexible PPS outputs are selected. Bits[22:16] are valid only when three or more Flexible PPS outputs are selected. Bits[14:8] are valid only when two or more Flexible PPS outputs are selected. Bits[6:4] are valid only when Flexible PPS feature is selected.

Figure 43-228. MAC_PPS_Control Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED		RESERVED		RESERVED		
R-0h	R-0h		R-0h		R-0h		
23	22	21	20	19	18	17	16
RESERVED	RESERVED		RESERVED		RESERVED		
R-0h	R-0h		R-0h		R-0h		
15	14	13	12	11	10	9	8
RESERVED	TRGTMODSEL1		RESERVED		PPSCMD1		
R-0h	R/W-0h		R-0h		R/W-0h		
7	6	5	4	3	2	1	0
RESERVED	TRGTMODSEL0		PPSEN0	PPSCTRL_PPSCMD			
R-0h	R/W-0h		R/W-0h	R/W-0h			

Table 43-282. MAC_PPS_Control Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved.
30-29	RESERVED	R	0h	Reserved.
28-27	RESERVED	R	0h	Reserved.
26-24	RESERVED	R	0h	Reserved.
23	RESERVED	R	0h	Reserved.
22-21	RESERVED	R	0h	Reserved.
20-19	RESERVED	R	0h	Reserved.
18-16	RESERVED	R	0h	Reserved.
15	RESERVED	R	0h	Reserved.
14-13	TRGTMODSEL1	R/W	0h	Target Time Register Mode for PPS1 Output This field indicates the Target Time registers (MAC_PPS1_Target_Time_Seconds and MAC_PPS1_Target_Time_Nanoseconds) mode for PPS1 output signal. 0h = Target Time registers are programmed only for generating the interrupt event. The Flexible PPS function must not be enabled in this mode, otherwise spurious transitions may be observed on the corresponding ptp_pps_o output port : 0x0 1h = Reserved : 0x1 2h = Target Time registers are programmed for generating the interrupt event and starting or stopping the PPS0 output signal generation : 0x2 3h = Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No interrupt is asserted : 0x3
12-11	RESERVED	R	0h	Reserved.

Table 43-282. MAC_PPS_Control Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10-8	PPSCMD1	R/W	0h	Flexible PPS1 Output Control This field controls the flexible PPS1 output (ptp_pps_o[1]) signal. This field is similar to the PPSCMD0 field. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect.
7	RESERVED	R	0h	Reserved.
6-5	TRGTMODSEL0	R/W	0h	Target Time Register Mode for PPS0 Output This field indicates the Target Time registers (MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds) mode for PPS0 output signal: 0h = Target Time registers are programmed only for generating the interrupt event. The Flexible PPS function must not be enabled in this mode, otherwise spurious transitions may be observed on the corresponding ptp_pps_o output port : 0x0 1h = Reserved : 0x1 2h = Target Time registers are programmed for generating the interrupt event and starting or stopping the PPS0 output signal generation : 0x2 3h = Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No interrupt is asserted : 0x3
4	PPSEN0	R/W	0h	Flexible PPS Output Mode Enable When this bit is set, Bits[3:0] function as PPSCMD. When this bit is reset, Bits[3:0] function as PPSCTRL (Fixed PPS mode). 0h = Flexible PPS Output Mode is disabled : 0x0 1h = Flexible PPS Output Mode is enabled : 0x1

Table 43-282. MAC_PPS_Control Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	PPSCTRL_PPSCMD	R/W	0h	PPS Output Frequency Control This field controls the frequency of the PPS0 output (ptp_pps_o[0]) signal. The default value of PPSCTRL is 0000, and the PPS output is 1 pulse (of width clk_ptp_i) every second. For other values of PPSCTRL, the PPS output becomes a generated clock of following frequencies: - 0001: The binary rollover is 2 Hz, and the digital rollover is 1 Hz. - 0010: The binary rollover is 4 Hz, and the digital rollover is 2 Hz. - 0011: The binary rollover is 8 Hz, and the digital rollover is 4 Hz. - 0100: The binary rollover is 16 Hz, and the digital rollover is 8 Hz. - .. - 1111: The binary rollover is 32.768 KHz and the digital rollover is 16.384 KHz. Note: In the binary rollover mode, the PPS output (ptp_pps_o) has a duty cycle of 50 percent with these frequencies. In the digital rollover mode, the PPS output frequency is an average number. The actual clock is of different frequency that gets synchronized every second. For example: - When PPSCTRL = 0001, the PPS (1 Hz) has a low period of 537 ms and a high period of 463 ms - When PPSCTRL = 0010, the PPS (2 Hz) is a sequence of One clock of 50 percent duty cycle and 537 ms period Second clock of 463 ms period (268 ms low and 195 ms high) - When PPSCTRL = 0011, the PPS (4 Hz) is a sequence of Three clocks of 50 percent duty cycle and 268 ms period Fourth clock of 195 ms period (134 ms low and 61 ms high) This behavior is because of the non-linear toggling of bits in the digital rollover mode in the MAC_System_Time_Nanoseconds register. or Flexible PPS Output (ptp_pps_o[0]) Control Programming these bits with a non-zero value instructs the MAC to initiate an event. When the command is transferred or synchronized to the PTP clock domain, these bits get cleared automatically. The software should ensure that these bits are programmed only when they are 'all-zero'. The following list describes the values of PPSCMD0: - 0000: No Command - 0001: START Single Pulse This command generates single pulse rising at the start point defined in Target Time Registers (register 455 and 456) and of a duration defined in the PPS0 Width Register. - 0010: START Pulse Train This command generates the train of pulses rising at the start point defined in the Target Time Registers and of a duration defined in the PPS0 Width Register and repeated at interval defined in the PPS Interval Register. By default, the PPS pulse train is free-running unless stopped by the 'Stop Pulse train at time' or 'Stop Pulse Train immediately' commands. - 0011: Cancel START This command cancels the START Single Pulse and START Pulse Train commands if the system time has not crossed the programmed start time. - 0100: STOP Pulse train at time This command stops the train of pulses initiated by the START Pulse Train command (PPSCMD = 0010) after the time programmed in the Target Time registers elapses. - 0101: STOP Pulse Train immediately This command immediately stops the train of pulses initiated by the START Pulse Train command (PPSCMD = 0010). - 0110: Cancel STOP Pulse train This command cancels the STOP pulse train at time command if the programmed stop time has not elapsed. The PPS pulse train becomes free-running on the successful execution of this command.

Table 43-282. MAC_PPS_Control Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
				- 0111-1111: Reserved Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect.

43.7.3.190 MAC_PPS0_Target_Time_Seconds Register (Offset = B80h) [Reset = 0h]

MAC_PPS0_Target_Time_Seconds is shown in Figure 43-229 and described in Table 43-283.

Return to the Summary Table.

The PPS Target Time Seconds register, along with PPS Target Time Nanoseconds register, is used to schedule an interrupt event [Bit 1 of MAC_Timestamp_Status] when the system time exceeds the value programmed in these registers.

Figure 43-229. MAC_PPS0_Target_Time_Seconds Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSTRH0																															
R/W-0h																															

Table 43-283. MAC_PPS0_Target_Time_Seconds Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TSTRH0	R/W	0h	PPS Target Time Seconds Register This field stores the time in seconds. When the timestamp value matches or exceeds both Target Timestamp registers, the MAC starts or stops the PPS signal output and generates an interrupt (if enabled) based on Target Time mode selected for the corresponding PPS output in the MAC_PPS_Control register.

43.7.3.191 MAC_PPS0_Target_Time_Nanoseconds Register (Offset = B84h) [Reset = 0h]

MAC_PPS0_Target_Time_Nanoseconds is shown in Figure 43-230 and described in Table 43-284.

Return to the Summary Table.

PPS0 Target Time Nanoseconds register.

Figure 43-230. MAC_PPS0_Target_Time_Nanoseconds Register

31	30	29	28	27	26	25	24
TRGTBUSY0	TTSL0						
R/W-0h	R/W-0h						
23	22	21	20	19	18	17	16
TTSL0							
R/W-0h							
15	14	13	12	11	10	9	8
TTSL0							
R/W-0h							
7	6	5	4	3	2	1	0
TTSL0							
R/W-0h							

Table 43-284. MAC_PPS0_Target_Time_Nanoseconds Register Field Descriptions

Bit	Field	Type	Reset	Description
31	TRGTBUSY0	R/W	0h	PPS Target Time Register Busy The MAC sets this bit when the PPSCMD0 field in the MAC_PPS_Control register is programmed to 010 or 011. Programming the PPSCMD0 field to 010 or 011 instructs the MAC to synchronize the Target Time Registers to the PTP clock domain. The MAC clears this bit after synchronizing the Target Time Registers to the PTP clock domain The application must not update the Target Time Registers when this bit is read as 1. Otherwise, the synchronization of the previous programmed time gets corrupted. 0h = PPS Target Time Register Busy status is not detected : 0x0 1h = PPS Target Time Register Busy is detected : 0x1
30-0	TTSL0	R/W	0h	Target Time Low for PPS Register This register stores the time in (signed) nanoseconds. When the value of the timestamp matches the value in both Target Timestamp registers, the MAC starts or stops the PPS signal output and generates an interrupt (if enabled) based on the TRGTMODSEL0 field (Bits [6:5]) in MAC_PPS_Control. When the TSCTRLSSR bit is reset in the MAC_Timestamp_Control register, this value should be (time in ns / 0.465). The actual start or stop time of the PPS signal output may have an error margin up to one unit of sub-second increment value. When the TSCTRLSSR bit is set in the MAC_Timestamp_Control register, this value should not exceed 0x3B9A_C9FF. The actual start or stop time of the PPS signal output may have an error margin up to one unit of sub-second increment value. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect.

43.7.3.192 MAC_PPS0_Interval Register (Offset = B88h) [Reset = 0h]

MAC_PPS0_Interval is shown in Figure 43-231 and described in Table 43-285.

Return to the Summary Table.

The PPS0 Interval register contains the number of units of sub-second increment value between the rising edges of PPS0 signal output (ptp_pps_o[0]).

Figure 43-231. MAC_PPS0_Interval Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PPSINT0																															
R/W-0h																															

Table 43-285. MAC_PPS0_Interval Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PPSINT0	R/W	0h	PPS Output Signal Interval These bits store the interval between the rising edges of PPS0 signal output. The interval is stored in terms of number of units of sub-second increment value. You need to program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20 ns), and desired interval between the rising edges of PPS0 signal output is 100 ns (that is, 5 units of sub-second increment value), you should program value 4 (5-1) in this register.

43.7.3.193 MAC_PPS0_Width Register (Offset = B8Ch) [Reset = 0h]

MAC_PPS0_Width is shown in Figure 43-232 and described in Table 43-286.

Return to the Summary Table.

The PPS0 Width register contains the number of units of sub-second increment value between the rising and corresponding falling edges of PPS0 signal output (ptp_pps_o[0]).

Figure 43-232. MAC_PPS0_Width Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PPSWIDTH0																															
R/W-0h																															

Table 43-286. MAC_PPS0_Width Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PPSWIDTH0	R/W	0h	PPS Output Signal Width These bits store the width between the rising edge and corresponding falling edge of PPS0 signal output. The width is stored in terms of number of units of sub-second increment value. You need to program one value less than the required interval. For example, if PTP reference clock is 50 MHz (period of 20 ns), and width between the rising and corresponding falling edges of PPS0 signal output is 80 ns (that is, four units of sub-second increment value), you should program value 3 (4-1) in this register. Note: The value programmed in this register must be lesser than the value programmed in MAC_PPS0_Interval.

43.7.3.194 MAC_PPS1_Target_Time_Seconds Register (Offset = B90h) [Reset = 0h]

MAC_PPS1_Target_Time_Seconds is shown in Figure 43-233 and described in Table 43-287.

Return to the Summary Table.

The PPS Target Time Seconds register, along with PPS Target Time Nanoseconds register, is used to schedule an interrupt event [Bit 1 of MAC_Timestamp_Status] when the system time exceeds the value programmed in these registers.

Figure 43-233. MAC_PPS1_Target_Time_Seconds Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSTRH1																															
R/W-0h																															

Table 43-287. MAC_PPS1_Target_Time_Seconds Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TSTRH1	R/W	0h	PPS Target Time Seconds Register This field stores the time in seconds. When the timestamp value matches or exceeds both Target Timestamp registers, the MAC starts or stops the PPS signal output and generates an interrupt (if enabled) based on Target Time mode selected for the corresponding PPS output in the MAC_PPS_Control register.

43.7.3.195 MAC_PPS1_Target_Time_Nanoseconds Register (Offset = B94h) [Reset = 0h]

MAC_PPS1_Target_Time_Nanoseconds is shown in Figure 43-234 and described in Table 43-288.

Return to the Summary Table.

PPS0 Target Time Nanoseconds register.

Figure 43-234. MAC_PPS1_Target_Time_Nanoseconds Register

31	30	29	28	27	26	25	24
TRGTBUSY1	TTSL1						
R/W-0h	R/W-0h						
23	22	21	20	19	18	17	16
TTSL1							
R/W-0h							
15	14	13	12	11	10	9	8
TTSL1							
R/W-0h							
7	6	5	4	3	2	1	0
TTSL1							
R/W-0h							

Table 43-288. MAC_PPS1_Target_Time_Nanoseconds Register Field Descriptions

Bit	Field	Type	Reset	Description
31	TRGTBUSY1	R/W	0h	PPS Target Time Register Busy The MAC sets this bit when the PPSCMD0 field in the MAC_PPS_Control register is programmed to 010 or 011. Programming the PPSCMD0 field to 010 or 011 instructs the MAC to synchronize the Target Time Registers to the PTP clock domain. The MAC clears this bit after synchronizing the Target Time Registers to the PTP clock domain The application must not update the Target Time Registers when this bit is read as 1. Otherwise, the synchronization of the previous programmed time gets corrupted. 0h = PPS Target Time Register Busy status is not detected : 0x0 1h = PPS Target Time Register Busy is detected : 0x1
30-0	TTSL1	R/W	0h	Target Time Low for PPS Register This register stores the time in (signed) nanoseconds. When the value of the timestamp matches the value in both Target Timestamp registers, the MAC starts or stops the PPS signal output and generates an interrupt (if enabled) based on the TRGTMODSEL0 field (Bits [6:5]) in MAC_PPS_Control. When the TSCTRLSSR bit is reset in the MAC_Timestamp_Control register, this value should be (time in ns / 0.465). The actual start or stop time of the PPS signal output may have an error margin up to one unit of sub-second increment value. When the TSCTRLSSR bit is set in the MAC_Timestamp_Control register, this value should not exceed 0x3B9A_C9FF. The actual start or stop time of the PPS signal output may have an error margin up to one unit of sub-second increment value. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect.

43.7.3.196 MAC_PPS1_Interval Register (Offset = B98h) [Reset = 0h]

MAC_PPS1_Interval is shown in Figure 43-235 and described in Table 43-289.

Return to the Summary Table.

The PPS0 Interval register contains the number of units of sub-second increment value between the rising edges of PPS0 signal output (ptp_pps_o[0]).

Figure 43-235. MAC_PPS1_Interval Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PPSINT1																															
R/W-0h																															

Table 43-289. MAC_PPS1_Interval Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PPSINT1	R/W	0h	PPS Output Signal Interval These bits store the interval between the rising edges of PPS0 signal output. The interval is stored in terms of number of units of sub-second increment value. You need to program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20 ns), and desired interval between the rising edges of PPS0 signal output is 100 ns (that is, 5 units of sub-second increment value), you should program value 4 (5-1) in this register.

43.7.3.197 MAC_PPS1_Width Register (Offset = B9Ch) [Reset = 0h]

MAC_PPS1_Width is shown in Figure 43-236 and described in Table 43-290.

Return to the Summary Table.

The PPS0 Width register contains the number of units of sub-second increment value between the rising and corresponding falling edges of PPS0 signal output (ptp_pps_o[0]).

Figure 43-236. MAC_PPS1_Width Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PPSWIDTH1																															
R/W-0h																															

Table 43-290. MAC_PPS1_Width Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PPSWIDTH1	R/W	0h	PPS Output Signal Width These bits store the width between the rising edge and corresponding falling edge of PPS0 signal output. The width is stored in terms of number of units of sub-second increment value. You need to program one value less than the required interval. For example, if PTP reference clock is 50 MHz (period of 20 ns), and width between the rising and corresponding falling edges of PPS0 signal output is 80 ns (that is, four units of sub-second increment value), you should program value 3 (4-1) in this register. Note: The value programmed in this register must be lesser than the value programmed in MAC_PPS0_Interval.

43.7.3.198 MAC_PTO_Control Register (Offset = BC0h) [Reset = 0h]

MAC_PTO_Control is shown in Figure 43-237 and described in Table 43-291.

Return to the Summary Table.

This register controls the PTP Offload Engine operation. This register is available only when the Enable PTP Timestamp Offload feature is selected.

Figure 43-237. MAC_PTO_Control Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
DN							
R/W-0h							
7	6	5	4	3	2	1	0
PDRDIS	DRRDIS	APDREQTRIG	ASYNCTRIG	RESERVED	APDREQEN	ASYNCEN	PTOEN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h

Table 43-291. MAC_PTO_Control Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved.
15-8	DN	R/W	0h	Domain Number This field indicates the domain Number in which the PTP node is operating.
7	PDRDIS	R/W	0h	Disable Peer Delay Response response generation When this bit is set, the Peer Delay Response (Pdelay_Resp) response is not be generated for received Peer Delay Request (Pdelay_Req) request packet, as required by the programmed mode. Note: Setting this bit to 1 affects the normal PTP Offload operation and the time synchronization. So, this bit must be set only if there is problem with Pdelay_Resp generation in Hardware and/or Pdelay_Resp generation is handled by Software. 0h = Peer Delay Response response generation is enabled : 0x0 1h = Peer Delay Response response generation is disabled : 0x1
6	DRRDIS	R/W	0h	Disable PTO Delay Request/Response response generation When this bit is set, the Delay Request and Delay response is not generated for received SYNC and Delay request packet respectively, as required by the programmed mode. 0h = PTO Delay Request/Response response generation is enabled : 0x0 1h = PTO Delay Request/Response response generation is disabled : 0x1
5	APDREQTRIG	R/W	0h	Automatic PTP Pdelay_Req message Trigger When this bit is set, one PTP Pdelay_Req message is transmitted. This bit is automatically cleared after the PTP Pdelay_Req message is transmitted. The application should set the APDREQEN bit for this operation. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0h = Automatic PTP Pdelay_Req message Trigger is disabled : 0x0 1h = Automatic PTP Pdelay_Req message Trigger is enabled : 0x1

Table 43-291. MAC_PTO_Control Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	ASYNCTRIG	R/W	0h	Automatic PTP SYNC message Trigger When this bit is set, one PTP SYNC message is transmitted. This bit is automatically cleared after the PTP SYNC message is transmitted. The application should set the ASYNCEN bit for this operation. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0h = Automatic PTP SYNC message Trigger is disabled : 0x0 1h = Automatic PTP SYNC message Trigger is enabled : 0x1
3	RESERVED	R	0h	Reserved.
2	APDREQEN	R/W	0h	Automatic PTP Pdelay_Req message Enable When this bit is set, PTP Pdelay_Req message is generated periodically based on interval programmed or trigger from application, when the MAC is programmed to be in Peer-to-Peer Transparent mode. 0h = Automatic PTP Pdelay_Req message is disabled : 0x0 1h = Automatic PTP Pdelay_Req message is enabled : 0x1
1	ASYNCEN	R/W	0h	Automatic PTP SYNC message Enable When this bit is set, PTP SYNC message is generated periodically based on interval programmed or trigger from application, when the MAC is programmed to be in Clock Master mode. 0h = Automatic PTP SYNC message is disabled : 0x0 1h = Automatic PTP SYNC message is enabled : 0x1
0	PTOEN	R/W	0h	PTP Offload Enable When this bit is set, the PTP Offload feature is enabled. 0h = PTP Offload feature is disabled : 0x0 1h = PTP Offload feature is enabled : 0x1

43.7.3.199 MAC_Source_Port_Identity0 Register (Offset = BC4h) [Reset = 0h]

MAC_Source_Port_Identity0 is shown in Figure 43-238 and described in Table 43-292.

Return to the Summary Table.

This register contains Bits[31:0] of the 80-bit Source Port Identity of the PTP node. This register is available only when the Enable PTP Timestamp Offload feature is selected.

Figure 43-238. MAC_Source_Port_Identity0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI0																															
R/W-0h																															

Table 43-292. MAC_Source_Port_Identity0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SPI0	R/W	0h	Source Port Identity 0 This field indicates bits [31:0] of sourcePortIdentity of PTP node.

43.7.3.200 MAC_Source_Port_Identity1 Register (Offset = BC8h) [Reset = 0h]

MAC_Source_Port_Identity1 is shown in Figure 43-239 and described in Table 43-293.

Return to the Summary Table.

This register contains Bits[63:32] of the 80-bit Source Port Identity of the PTP node. This register is available only when the Enable PTP Timestamp Offload feature is selected.

Figure 43-239. MAC_Source_Port_Identity1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI1																															
R/W-0h																															

Table 43-293. MAC_Source_Port_Identity1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SPI1	R/W	0h	Source Port Identity 1 This field indicates bits [63:32] of sourcePortIdentity of PTP node.

43.7.3.201 MAC_Source_Port_Identity2 Register (Offset = BCCh) [Reset = 0h]

MAC_Source_Port_Identity2 is shown in Figure 43-240 and described in Table 43-294.

Return to the Summary Table.

This register contains Bits[79:64] of the 80-bit Source Port Identity of the PTP node. This register is available only when the Enable PTP Timestamp Offload feature is selected.

Figure 43-240. MAC_Source_Port_Identity2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																SPI2															
R-0h																R/W-0h															

Table 43-294. MAC_Source_Port_Identity2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved.
15-0	SPI2	R/W	0h	Source Port Identity 2 This field indicates bits [79:64] of sourcePortIdentity of PTP node.

43.7.3.202 MAC_Log_Message_Interval Register (Offset = BD0h) [Reset = 0h]

MAC_Log_Message_Interval is shown in Figure 43-241 and described in Table 43-295.

Return to the Summary Table.

This register contains the periodic intervals for automatic PTP packet generation. This register is available only when the Enable PTP Timestamp Offload feature is selected.

Figure 43-241. MAC_Log_Message_Interval Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LMPDRI								RESERVED							
R/W-0h								R-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					DRSYNCR			LSI							
R-0h					R/W-0h			R/W-0h							

Table 43-295. MAC_Log_Message_Interval Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	LMPDRI	R/W	0h	Log Min Pdelay_Req Interval This field indicates logMinPdelayReqInterval of PTP node. This is used to schedule the periodic Pdelay request packet transmission. Allowed values are -15 to 15.Negative value must be represented in 2's-complement form. For example, if the required value is -1, the value programmed must be 0xFF.
23-11	RESERVED	R	0h	Reserved.
10-8	DRSYNCR	R/W	0h	Delay_Req to SYNC Ratio In Slave mode, it is used for controlling frequency of Delay_Req messages transmitted. - 0: DelayReq generated for every received SYNC - 1: DelayReq generated every alternate reception of SYNC - 2: for every 4 SYNC messages - 3: for every 8 SYNC messages - 4: for every 16 SYNC messages - 5: for every 32 SYNC messages - 6-7: Reserved The master sends this information (logMinDelayReqInterval) in the DelayResp PTP messages to the slave. The DWC_ether_qos Receiver processes this value from the received DelayResp messages and updates this field accordingly. In the Slave mode, the host must not write/update this register unless it has to override the received value. In Master mode, the sum of this field and logSyncInterval (LSI) field is provided in the logMinDelayReqInterval field of the generated multicast Delay_Resp PTP message. Access restriction applies. Updated based on the event. Setting 1 sets. Setting 0 clears. 0h = DelayReq generated for every received SYNC : 0x0 1h = DelayReq generated every alternate reception of SYNC : 0x1 2h = for every 4 SYNC messages : 0x2 3h = for every 8 SYNC messages : 0x3 4h = for every 16 SYNC messages : 0x4 5h = for every 32 SYNC messages : 0x5 6h = Reserved : 0x6
7-0	LSI	R/W	0h	Log Sync Interval This field indicates the periodicity of the automatically generated SYNC message when the PTP node is Master. Allowed values are -15 to 15. Negative value must be represented in 2's-complement form. For example, if the required value is -1, the value programmed must be 0xFF.

43.7.3.203 MTL_Operation_Mode Register (Offset = C00h) [Reset = 0h]

MTL_Operation_Mode is shown in Figure 43-242 and described in Table 43-296.

Return to the Summary Table.

The Operation Mode register establishes the Transmit and Receive operating modes and commands.

Figure 43-242. MTL_Operation_Mode Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						CNTCLR	CNTPRST
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	SCHALG		RESERVED		RAA	DTXSTS	RESERVED
R-0h	R/W-0h		R-0h		R/W-0h	R/W-0h	R-0h

Table 43-296. MTL_Operation_Mode Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved.
9	CNTCLR	R/W	0h	Counters Reset When this bit is set, all counters are reset. This bit is cleared automatically after 1 clock cycle. If this bit is set along with CNT_PRESET bit, CNT_PRESET has precedence. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0h = Counters are not reset : 0x0 1h = All counters are reset : 0x1
8	CNTPRST	R/W	0h	Counters Preset When this bit is set, - MTL_TxQ[0-7]_Underflow register is initialized/preset to 12'h7F0. - Missed Packet and Overflow Packet counters in MTL_RxQ[0-7]_Missed_Packet_Overflow_Cnt register is initialized/ preset to 12'h7F0. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0h = Counters Preset is disabled : 0x0 1h = Counters Preset is enabled : 0x1
7	RESERVED	R	0h	Reserved.
6-5	SCHALG	R/W	0h	Tx Scheduling Algorithm This field indicates the algorithm for Tx scheduling: 0h = WRR algorithm : 0x0 1h = WFQ algorithm when DCB feature is selected.Otherwise, Reserved : 0x1 2h = DWRR algorithm when DCB feature is selected.Otherwise, Reserved : 0x2 3h = Strict priority algorithm : 0x3
4-3	RESERVED	R	0h	Reserved.

Table 43-296. MTL_Operation_Mode Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	RAA	R/W	0h	Receive Arbitration Algorithm This field is used to select the arbitration algorithm for the Rx side. - 0: Strict priority (SP) Queue 0 has the lowest priority and the last queue has the highest priority. - 1: Weighted Strict Priority (WSP) 0h = Strict priority (SP) : 0x0 1h = Weighted Strict Priority (WSP) : 0x1
1	DTXSTS	R/W	0h	Drop Transmit Status When this bit is set, the Tx packet status received from the MAC is dropped in the MTL. When this bit is reset, the Tx packet status received from the MAC is forwarded to the application. 0h = Drop Transmit Status is disabled : 0x0 1h = Drop Transmit Status is enabled : 0x1
0	RESERVED	R	0h	Reserved.

43.7.3.204 MTL_DBG_CTL Register (Offset = C08h) [Reset = 0h]

MTL_DBG_CTL is shown in Figure 43-243 and described in Table 43-297.

Return to the Summary Table.

The FIFO Debug Access Control and Status register controls the operation mode of FIFO debug access.

Figure 43-243. MTL_DBG_CTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
STSIE	PKTIE	FIFOSEL		FIFOWREN	FIFORDEN	RSTSEL	RSTALL
R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	PKTSTATE		RESERVED	BYTEEN		DBGMOD	FDBGEN
R-0h	R/W-0h		R-0h	R/W-0h		R/W-0h	R/W-0h

Table 43-297. MTL_DBG_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved.
15	STSIE	R/W	0h	Transmit Status Available Interrupt Status Enable When this bit is set, an interrupt is generated when Transmit status is available in slave mode. 0h = Transmit Packet Available Interrupt Status is disabled : 0x0 1h = Transmit Packet Available Interrupt Status is enabled : 0x1
14	PKTIE	R/W	0h	Receive Packet Available Interrupt Status Enable When this bit is set, an interrupt is generated when EOP of received packet is written to the Rx FIFO. 0h = Receive Packet Available Interrupt Status is disabled : 0x0 1h = Receive Packet Available Interrupt Status is enabled : 0x1
13-12	FIFOSEL	R/W	0h	FIFO Selected for Access This field indicates the FIFO selected for debug access: 0h = Tx FIFO : 0x0 1h = Tx Status FIFO (only read access when SLVMOD is set) : 0x1 2h = TSO FIFO (cannot be accessed when SLVMOD is set) : 0x2 3h = Rx FIFO : 0x3
11	FIFOWREN	R/W	0h	FIFO Write Enable When this bit is set, it enables the Write operation on selected FIFO when FIFO Debug Access is enabled. This bit must not be written to 1 when FIFO Debug Access is not enabled, that is FDBGEN bit is 0. Access restriction applies. Self-cleared. Setting 0 clears. Setting 1 sets. 0h = FIFO Write is disabled : 0x0 1h = FIFO Write is enabled : 0x1

Table 43-297. MTL_DBG_CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	FIFORDEN	R/W	0h	FIFO Read Enable When this bit is set, it enables the Read operation on selected FIFO when FIFO Debug Access is enabled. This bit must not be written to 1 when FIFO Debug Access is not enabled, that is FDBGEN bit is 0. Access restriction applies. Self-cleared. Setting 0 clears. Setting 1 sets. 0h = FIFO Read is disabled : 0x0 1h = FIFO Read is enabled : 0x1
9	RSTSEL	R/W	0h	Reset Pointers of Selected FIFO When this bit is set, the pointers of the currently-selected FIFO are reset when FIFO Debug Access is enabled. This bit must not be written to 1 when FIFO Debug Access is not enabled, that is FDBGEN bit is 0. Access restriction applies. Self-cleared. Setting 0 clears. Setting 1 sets. 0h = Reset Pointers of Selected FIFO is disabled : 0x0 1h = Reset Pointers of Selected FIFO is enabled : 0x1
8	RSTALL	R/W	0h	Reset All Pointers When this bit is set, the pointers of all FIFOs are reset when FIFO Debug Access is enabled. This bit must not be written to 1 when FIFO Debug Access is not enabled, that is FDBGEN bit is 0. Access restriction applies. Self-cleared. Setting 0 clears. Setting 1 sets. 0h = Reset All Pointers is disabled : 0x0 1h = Reset All Pointers is enabled : 0x1
7	RESERVED	R	0h	Reserved.
6-5	PKTSTATE	R/W	0h	Encoded Packet State This field is used to write the control information to the Tx FIFO or Rx FIFO. Tx FIFO: - 00: Packet Data - 01: Control Word - 10: SOP Data - 11: EOP Data Rx FIFO: - 00: Packet Data - 01: Normal Status - 10: Last Status - 11: EOP 0h = Packet Data : 0x0 1h = Control Word/Normal Status : 0x1 2h = SOP Data/Last Status : 0x2 3h = EOP Data/EOP : 0x3
4	RESERVED	R	0h	Reserved.
3-2	BYTEEN	R/W	0h	Byte Enables This field indicates the number of data bytes valid in the data register during Write operation. This is valid only when PKTSTATE is 2'b10 (EOP) and Tx FIFO or Rx FIFO is selected. 0h = Byte 0 valid : 0x0 1h = Byte 0 and Byte 1 are valid : 0x1 2h = Byte 0, Byte 1, and Byte 2 are valid : 0x2 3h = All four bytes are valid : 0x3

Table 43-297. MTL_DBG_CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	DBGMOD	R/W	0h	Debug Mode Access to FIFO When this bit is set, it indicates that the current access to the FIFO is read, write, and debug access. In this mode, the following access types are allowed: - Read and Write access to Tx FIFO, TSO FIFO, and Rx FIFO - Read access is allowed to Tx Status FIFO. When this bit is reset, it indicates that the current access to the FIFO is slave access bypassing the DMA. In this mode, the following access are allowed: - Write access to the Tx FIFO - Read access to the Rx FIFO and Tx Status FIFO 0h = Debug Mode Access to FIFO is disabled : 0x0 1h = Debug Mode Access to FIFO is enabled : 0x1
0	FDBGEN	R/W	0h	FIFO Debug Access Enable When this bit is set, it indicates that the debug mode access to the FIFO is enabled. When this bit is reset, it indicates that the FIFO can be accessed only through a master interface. 0h = FIFO Debug Access is disabled : 0x0 1h = FIFO Debug Access is enabled : 0x1

43.7.3.205 MTL_DBG_STS Register (Offset = C0Ch) [Reset = 18h]

MTL_DBG_STS is shown in Figure 43-244 and described in Table 43-298.

Return to the Summary Table.

The FIFO Debug Status register contains the status of FIFO debug access.

Figure 43-244. MTL_DBG_STS Register

31	30	29	28	27	26	25	24
LOCR							
R-0h							
23	22	21	20	19	18	17	16
LOCR							
R-0h							
15	14	13	12	11	10	9	8
LOCR	RESERVED					STSI	PKTI
R-0h	R-0h					R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED			BYTEEN		PKTSTATE		FIFOBUSY
R-0h			R-3h		R-0h		R-0h

Table 43-298. MTL_DBG_STS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	LOCR	R	0h	Remaining Locations in the FIFO Slave Access Mode: This field indicates the space available in selected FIFO. Debug Access Mode: This field contains the Write or Read pointer value of the selected FIFO during Write or Read operation, respectively. Reset: In single Tx Queue configurations, (DWC_EQOS_TXFIFO_SIZE/(DWC_EQOS_DATAWIDTH/8)), Otherwise 0000H
14-10	RESERVED	R	0h	Reserved.
9	STSI	R/W	0h	Transmit Status Available Interrupt Status When set, this bit indicates that the Slave mode Tx packet is transmitted, and the status is available in Tx Status FIFO. This bit is reset when 1 is written to this bit. 0h = Transmit Status Available Interrupt Status not detected : 0x0 1h = Transmit Status Available Interrupt Status detected : 0x1
8	PKTI	R/W	0h	Receive Packet Available Interrupt Status When set, this bit indicates that MAC layer has written the EOP of received packet to the Rx FIFO. This bit is reset when 1 is written to this bit. 0h = Receive Packet Available Interrupt Status not detected : 0x0 1h = Receive Packet Available Interrupt Status detected : 0x1
7-5	RESERVED	R	0h	Reserved.
4-3	BYTEEN	R	3h	Byte Enables This field indicates the number of data bytes valid in the data register during Read operation. This is valid only when PKTSTATE is 2'b10 (EOP) and Tx FIFO or Rx FIFO is selected. 0h = Byte 0 valid : 0x0 1h = Byte 0 and Byte 1 are valid : 0x1 2h = Byte 0, Byte 1, and Byte 2 are valid : 0x2 3h = All four bytes are valid : 0x3

Table 43-298. MTL_DBG_STS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-1	PKTSTATE	R	0h	Encoded Packet State This field is used to get the control or status information of the selected FIFO. Tx FIFO: - 00: Packet Data - 01: Control Word - 10: SOP Data - 11: EOP Data Rx FIFO: - 00: Packet Data - 01: Normal Status - 10: Last Status - 11: EOP This field is applicable only for Tx FIFO and Rx FIFO during Read operation. 0h = Packet Data : 0x0 1h = Control Word/Normal Status : 0x1 2h = SOP Data/Last Status : 0x2 3h = EOP Data/EOP : 0x3
0	FIFOBUSY	R	0h	FIFO Busy When set, this bit indicates that a FIFO operation is in progress in the MAC and content of the following fields is not valid: - All other fields of this register - All fields of the MTL_FIFO_Debug_Data register 0h = FIFO Busy not detected : 0x0 1h = FIFO Busy detected : 0x1

43.7.3.206 MTL_FIFO_Debug_Data Register (Offset = C10h) [Reset = 0h]

MTL_FIFO_Debug_Data is shown in Figure 43-245 and described in Table 43-299.

Return to the Summary Table.

The FIFO Debug Data register contains the data to be written to or read from the FIFOs.

Figure 43-245. MTL_FIFO_Debug_Data Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FDBGDATA																															
R/W-0h																															

Table 43-299. MTL_FIFO_Debug_Data Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FDBGDATA	R/W	0h	FIFO Debug Data During debug or slave access write operation, this field contains the data to be written to the Tx FIFO, Rx FIFO, or TSO FIFO. During debug or slave access read operation, this field contains the data read from the Tx FIFO, Rx FIFO, TSO FIFO, or Tx Status FIFO.

43.7.3.207 MTL_Interrupt_Status Register (Offset = C20h) [Reset = 0h]

MTL_Interrupt_Status is shown in Figure 43-246 and described in Table 43-300.

Return to the Summary Table.

The software driver (application) reads this register during interrupt service routine or polling to determine the interrupt status of MTL queues and the MAC.

Figure 43-246. MTL_Interrupt_Status Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED					RESERVED	DBGIS	RESERVED
R-0h					R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	Q1IS	Q0IS
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 43-300. MTL_Interrupt_Status Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved.
18	RESERVED	R	0h	Reserved.
17	DBGIS	R	0h	Debug Interrupt status This bit indicates an interrupt event during the slave access. To reset this bit, the application must read the FIFO Debug Access Status register to get the exact cause of the interrupt and clear its source. 0h = Debug Interrupt status not detected : 0x0 1h = Debug Interrupt status detected : 0x1
16	RESERVED	R	0h	Reserved.
15-8	RESERVED	R	0h	Reserved.
7	RESERVED	R	0h	Reserved.
6	RESERVED	R	0h	Reserved.
5	RESERVED	R	0h	Reserved.
4	RESERVED	R	0h	Reserved.
3	RESERVED	R	0h	Reserved.
2	RESERVED	R	0h	Reserved.
1	Q1IS	R	0h	Queue 1 Interrupt status This bit indicates that there is an interrupt from Queue 1. To reset this bit, the application must read the MTL_Q1_Interrupt_Control_Status register to get the exact cause of the interrupt and clear its source. 0h = Queue 1 Interrupt status not detected : 0x0 1h = Queue 1 Interrupt status detected : 0x1

Table 43-300. MTL_Interrupt_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	Q0IS	R	0h	Queue 0 Interrupt status This bit indicates that there is an interrupt from Queue 0. To reset this bit, the application must read Queue 0 Interrupt Control and Status register to get the exact cause of the interrupt and clear its source. 0h = Queue 0 Interrupt status not detected : 0x0 1h = Queue 0 Interrupt status detected : 0x1

43.7.3.208 MTL_RxQ_DMA_Map0 Register (Offset = C30h) [Reset = 0h]

MTL_RxQ_DMA_Map0 is shown in Figure 43-247 and described in Table 43-301.

Return to the Summary Table.

The Receive Queue and DMA Channel Mapping 0 register is reserved in EQOS-CORE and EQOS-MTL configurations.

Figure 43-247. MTL_RxQ_DMA_Map0 Register

31	30	29	28	27	26	25	24
RESERVED			RESERVED	RESERVED			RESERVED
R-0h			R-0h	R-0h			R-0h
23	22	21	20	19	18	17	16
RESERVED			RESERVED	RESERVED			RESERVED
R-0h			R-0h	R-0h			R-0h
15	14	13	12	11	10	9	8
RESERVED			Q1DDMACH	RESERVED			Q1MDMACH
R-0h			R/W-0h	R-0h			R/W-0h
7	6	5	4	3	2	1	0
RESERVED			Q0DDMACH	RESERVED			Q0MDMACH
R-0h			R/W-0h	R-0h			R/W-0h

Table 43-301. MTL_RxQ_DMA_Map0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved.
28	RESERVED	R	0h	Reserved.
27-25	RESERVED	R	0h	Reserved.
24	RESERVED	R	0h	Reserved.
23-21	RESERVED	R	0h	Reserved.
20	RESERVED	R	0h	Reserved.
19-17	RESERVED	R	0h	Reserved.
16	RESERVED	R	0h	Reserved.
15-13	RESERVED	R	0h	Reserved.
12	Q1DDMACH	R/W	0h	Queue 1 Enabled for DA-based DMA Channel Selection When set, this bit indicates that the packets received in Queue 1 are routed to a particular DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, or the Ethernet DA address. When reset, this bit indicates that the packets received in Queue 1 are routed to the DMA Channel programmed in the Q1MDMACH field (Bits[10:8]). 0h = Queue 1 disabled for DA-based DMA Channel Selection : 0x0 1h = Queue 1 enabled for DA-based DMA Channel Selection : 0x1
11-9	RESERVED	R	0h	Reserved.

Table 43-301. MTL_RxQ_DMA_Map0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	Q1MDMACH	R/W	0h	Queue 1 Mapped to DMA Channel This field controls the routing of the received packet in Queue 1 to the DMA channel: - 000: DMA Channel 0 - 001: DMA Channel 1 - 010: DMA Channel 2 - 011: DMA Channel 3 - 100: DMA Channel 4 - 101: DMA Channel 5 - 110: DMA Channel 6 - 111: DMA Channel 7 This field is valid when the Q1DDMACH field is reset. The width of this field depends on the number of RX DMA channels and not all the values may be valid in some configurations. For example, if the number of RX DMA channels selected is 2, only 000 and 001 are valid, the other bits are reserved.
7-5	RESERVED	R	0h	Reserved.
4	Q0DDMACH	R/W	0h	Queue 0 Enabled for DA-based DMA Channel Selection When set, this bit indicates that the packets received in Queue 0 are routed to a particular DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, or the Ethernet DA address. When reset, this bit indicates that the packets received in Queue 0 are routed to the DMA Channel programmed in the Q0MDMACH field. 0h = Queue 0 disabled for DA-based DMA Channel Selection : 0x0 1h = Queue 0 enabled for DA-based DMA Channel Selection : 0x1
3-1	RESERVED	R	0h	Reserved.
0	Q0MDMACH	R/W	0h	Queue 0 Mapped to DMA Channel This field controls the routing of the packet received in Queue 0 to the DMA channel: - 000: DMA Channel 0 - 001: DMA Channel 1 - 010: DMA Channel 2 - 011: DMA Channel 3 - 100: DMA Channel 4 - 101: DMA Channel 5 - 110: DMA Channel 6 - 111: DMA Channel 7 This field is valid when the Q0DDMACH field is reset. The width of this field depends on the number of RX DMA channels and not all the values may be valid in some configurations. For example, if the number of RX DMA channels selected is 2, only 000 and 001 are valid, the other bits are reserved.

43.7.3.209 MTL_TxQ0_Operation_Mode Register (Offset = D00h) [Reset = 0h]

MTL_TxQ0_Operation_Mode is shown in Figure 43-248 and described in Table 43-302.

Return to the Summary Table.

The Queue 0 Transmit Operation Mode register establishes the Transmit queue operating modes and commands.

Figure 43-248. MTL_TxQ0_Operation_Mode Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED												TQS			
R-0h												R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									TTC			TXQEN		TSF	FTQ
R-0h									R/W-0h			R/W-0h		R/W-0h	R/W-0h

Table 43-302. MTL_TxQ0_Operation_Mode Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved.
19-16	TQS	R/W	0h	Transmit Queue Size This field indicates the size of the allocated Transmit queues in blocks of 256 bytes. The TQS field is read-write only if the number of Tx Queues more than one, the reset value is 0x0 and indicates size of 256 bytes. When the number of Tx Queues is one, the field is read-only and the configured TX FIFO size in blocks of 256 bytes is reflected in the reset value. The width of this field depends on the Tx memory size selected in your configuration. For example, if the memory size is 2048, the width of this field is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits
15-7	RESERVED	R	0h	Reserved.
6-4	TTC	R/W	0h	Transmit Threshold Control These bits control the threshold level of the MTL Tx Queue. The transmission starts when the packet size within the MTL Tx Queue is larger than the threshold. In addition, full packets with length less than the threshold are also transmitted. These bits are used only when the TSF bit is reset. 0h = 32 : 0x0 1h = 64 : 0x1 2h = 96 : 0x2 3h = 128 : 0x3 4h = 192 : 0x4 5h = 256 : 0x5 6h = 384 : 0x6 7h = 512 : 0x7
3-2	TXQEN	R/W	0h	Transmit Queue Enable This field is used to enable/disable the transmit queue 0. - 2'b00: Not enabled - 2'b01: Reserved - 2'b10: Enabled - 2'b11: Reserved This field is Read Only in Single Queue configurations and Read Write in Multiple Queue configurations. Note: In multiple Tx queues configuration, all the queues are disabled by default. Enable the Tx queue by programming this field. 0h = Not enabled : 0x0 1h = Enable in AV mode(Reserved in non-AV) : 0x1 2h = Enabled : 0x2 3h = Reserved : 0x3

Table 43-302. MTL_TxQ0_Operation_Mode Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	TSF	R/W	0h	Transmit Store and Forward When this bit is set, the transmission starts when a full packet resides in the MTL Tx queue. When this bit is set, the TTC values specified in Bits[6:4] of this register are ignored. This bit should be changed only when the transmission is stopped. 0h = Transmit Store and Forward is disabled : 0x0 1h = Transmit Store and Forward is enabled : 0x1
0	FTQ	R/W	0h	Flush Transmit Queue When this bit is set, the Tx queue controller logic is reset to its default values. Therefore, all the data in the Tx queue is lost or flushed. This bit is internally reset when the flushing operation is complete. Until this bit is reset, you should not write to the MTL_TxQ1_Operation_Mode register. The data which is already accepted by the MAC transmitter is not flushed. It is scheduled for transmission and results in underflow and runt packet transmission. Note: The flush operation is complete only when the Tx queue is empty and the application has accepted the pending Tx Status of all transmitted packets. To complete this flush operation, the PHY Tx clock (clk_tx_i) should be active. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0h = Flush Transmit Queue is disabled : 0x0 1h = Flush Transmit Queue is enabled : 0x1

43.7.3.210 MTL_TxQ0_Underflow Register (Offset = D04h) [Reset = 0h]

MTL_TxQ0_Underflow is shown in Figure 43-249 and described in Table 43-303.

Return to the Summary Table.

The Queue 0 Underflow Counter register contains the counter for packets aborted because of Transmit queue underflow and packets missed because of Receive queue packet flush

Figure 43-249. MTL_TxQ0_Underflow Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				UFCNTOVF	UFFRMCNT		
R-0h				R-0h	R-0h		
7	6	5	4	3	2	1	0
UFFRMCNT							
R-0h							

Table 43-303. MTL_TxQ0_Underflow Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved.
11	UFCNTOVF	R	0h	Overflow Bit for Underflow Packet Counter This bit is set every time the Tx queue Underflow Packet Counter field overflows, that is, it has crossed the maximum count. In such a scenario, the overflow packet counter is reset to all-zeros and this bit indicates that the rollover happened. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = Overflow not detected for Underflow Packet Counter : 0x0 1h = Overflow detected for Underflow Packet Counter : 0x1
10-0	UFFRMCNT	R	0h	Underflow Packet Counter This field indicates the number of packets aborted by the controller because of Tx Queue Underflow. This counter is incremented each time the MAC aborts outgoing packet because of underflow. The counter is cleared when this register is read with mci_be_i[0] at 1'b1. Access restriction applies. Clears on read. Self-set to 1 on internal event.

43.7.3.211 MTL_TxQ0_Debug Register (Offset = D08h) [Reset = 0h]

MTL_TxQ0_Debug is shown in Figure 43-250 and described in Table 43-304.

Return to the Summary Table.

The Queue 0 Transmit Debug register gives the debug status of various blocks related to the Transmit queue.

Figure 43-250. MTL_TxQ0_Debug Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED	STXSTSF			RESERVED	PTXQ		
R-0h	R-0h			R-0h	R-0h		
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		TXSTSFSTS	TXQSTS	TWCSTS	TRCSTS		TXQPAUSED
R-0h		R-0h	R-0h	R-0h	R-0h		R-0h

Table 43-304. MTL_TxQ0_Debug Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved.
22-20	STXSTSF	R	0h	Number of Status Words in Tx Status FIFO of Queue This field indicates the current number of status in the Tx Status FIFO of this queue. When the DTXSTS bit of MTL_Operation_Mode register is set to 1, this field does not reflect the number of status words in Tx Status FIFO.
19	RESERVED	R	0h	Reserved.
18-16	PTXQ	R	0h	Number of Packets in the Transmit Queue This field indicates the current number of packets in the Tx Queue. When the DTXSTS bit of MTL_Operation_Mode register is set to 1, this field does not reflect the number of packets in the Transmit queue.
15-6	RESERVED	R	0h	Reserved.
5	TXSTSFSTS	R	0h	MTL Tx Status FIFO Full Status When high, this bit indicates that the MTL Tx Status FIFO is full. Therefore, the MTL cannot accept any more packets for transmission. 0h = MTL Tx Status FIFO Full status is not detected : 0x0 1h = MTL Tx Status FIFO Full status is detected : 0x1
4	TXQSTS	R	0h	MTL Tx Queue Not Empty Status When this bit is high, it indicates that the MTL Tx Queue is not empty and some data is left for transmission. 0h = MTL Tx Queue Not Empty status is not detected : 0x0 1h = MTL Tx Queue Not Empty status is detected : 0x1
3	TWCSTS	R	0h	MTL Tx Queue Write Controller Status When high, this bit indicates that the MTL Tx Queue Write Controller is active, and it is transferring the data to the Tx Queue. 0h = MTL Tx Queue Write Controller status is not detected : 0x0 1h = MTL Tx Queue Write Controller status is detected : 0x1

Table 43-304. MTL_TxQ0_Debug Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-1	TRCSTS	R	0h	MTL Tx Queue Read Controller Status This field indicates the state of the Tx Queue Read Controller: 0h = Idle state : 0x0 1h = Read state (transferring data to the MAC transmitter) : 0x1 2h = Waiting for pending Tx Status from the MAC transmitter : 0x2 3h = Flushing the Tx queue because of the Packet Abort request from the MAC : 0x3
0	TXQPAUSED	R	0h	Transmit Queue in Pause When this bit is high and the Rx flow control is enabled, it indicates that the Tx Queue is in the Pause condition (in the full-duplex only mode) because of the following: - Reception of the PFC packet for the priorities assigned to the Tx Queue when PFC is enabled - Reception of 802.3x Pause packet when PFC is disabled 0h = Transmit Queue in Pause status is not detected : 0x0 1h = Transmit Queue in Pause status is detected : 0x1

43.7.3.212 MTL_TxQ0_ETS_Status Register (Offset = D14h) [Reset = 0h]

MTL_TxQ0_ETS_Status is shown in Figure 43-251 and described in Table 43-305.

Return to the Summary Table.

The Queue 0 ETS Status register provides the average traffic transmitted in Queue 0.

Figure 43-251. MTL_TxQ0_ETS_Status Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								ABS																							
R-0h								R-0h																							

Table 43-305. MTL_TxQ0_ETS_Status Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved.
23-0	ABS	R	0h	Average Bits per Slot This field contains the average transmitted bits per slot. When the DCB operation is enabled for Queue 0, this field is computed over every 10 million bit times slot (4 ms in 2500 Mbps 10 ms in 1000 Mbps 100 ms in 100 Mbps). The maximum value is 0x989680.

43.7.3.213 MTL_TxQ0_Quantum_Weight Register (Offset = D18h) [Reset = 0h]

MTL_TxQ0_Quantum_Weight is shown in Figure 43-252 and described in Table 43-306.

Return to the Summary Table.

The Queue 0 Quantum or Weights register contains the quantum value for Deficit Weighted Round Robin (DWRR), weights for the Weighted Round Robin (WRR), and Weighted Fair Queuing (WFQ) for Queue 0.

Figure 43-252. MTL_TxQ0_Quantum_Weight Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											ISCQW																				
R-0h											R/W-0h																				

Table 43-306. MTL_TxQ0_Quantum_Weight Register Field Descriptions

Bit	Field	Type	Reset	Description
31-21	RESERVED	R	0h	Reserved.
20-0	ISCQW	R/W	0h	Quantum or Weights When the DCB operation is enabled with DWRR algorithm for Queue 0 traffic, this field contains the quantum value in bytes to be added to credit during every queue scanning cycle. The maximum value is 0x1312D0 bytes. When DCB operation is enabled with WFQ algorithm for Queue 0 traffic, this field contains the weight for this queue. The maximum value is 0x3FFF where weight of 0 indicates 100% bandwidth. Bits[20:14] must be written to zero. The higher the programmed weights lesser the bandwidth allocated for the particular Transmit Queue. This is because the weights are used to compute the packet finish time (weights*packet_size). Lesser the finish time, higher the probability of the packet getting scheduled first and using more bandwidth. When DCB operation or generic queuing operation is enabled with WRR algorithm for Queue 0 traffic, this field contains the weight for this queue. The maximum value is 0x64. Bits [20:7] must be written to zero.

43.7.3.214 MTL_Q0_Interrupt_Control_Status Register (Offset = D2Ch) [Reset = 0h]

MTL_Q0_Interrupt_Control_Status is shown in Figure 43-253 and described in Table 43-307.

Return to the Summary Table.

This register contains the interrupt enable and status bits for the queue 0 interrupts.

Figure 43-253. MTL_Q0_Interrupt_Control_Status Register

31	30	29	28	27	26	25	24
RESERVED							RXOIE
R-0h							R/W-0h
23	22	21	20	19	18	17	16
RESERVED							RXOVFIS
R-0h							R/W-0h
15	14	13	12	11	10	9	8
RESERVED						ABPSIE	TXUIE
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED						ABPSIS	TXUNFIS
R-0h						R/W-0h	R/W-0h

Table 43-307. MTL_Q0_Interrupt_Control_Status Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved.
24	RXOIE	R/W	0h	Receive Queue Overflow Interrupt Enable When this bit is set, the Receive Queue Overflow interrupt is enabled. When this bit is reset, the Receive Queue Overflow interrupt is disabled. 0h = Receive Queue Overflow Interrupt is disabled : 0x0 1h = Receive Queue Overflow Interrupt is enabled : 0x1
23-17	RESERVED	R	0h	Reserved.
16	RXOVFIS	R/W	0h	Receive Queue Overflow Interrupt Status This bit indicates that the Receive Queue had an overflow while receiving the packet. If a partial packet is transferred to the application, the overflow status is set in RDES3[21]. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Receive Queue Overflow Interrupt Status not detected : 0x0 1h = Receive Queue Overflow Interrupt Status detected : 0x1
15-10	RESERVED	R	0h	Reserved.
9	ABPSIE	R/W	0h	Average Bits Per Slot Interrupt Enable When this bit is set, the MAC asserts the sbd_intr_o or mci_intr_o interrupt when the average bits per slot status is updated. When this bit is cleared, the interrupt is not asserted for such an event. 0h = Average Bits Per Slot Interrupt is disabled : 0x0 1h = Average Bits Per Slot Interrupt is enabled : 0x1
8	TXUIE	R/W	0h	Transmit Queue Underflow Interrupt Enable When this bit is set, the Transmit Queue Underflow interrupt is enabled. When this bit is reset, the Transmit Queue Underflow interrupt is disabled. 0h = Transmit Queue Underflow Interrupt Status is disabled : 0x0 1h = Transmit Queue Underflow Interrupt Status is enabled : 0x1
7-2	RESERVED	R	0h	Reserved.

Table 43-307. MTL_Q0_Interrupt_Control_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	ABPSIS	R/W	0h	Average Bits Per Slot Interrupt Status When set, this bit indicates that the MAC has updated the ABS value. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Average Bits Per Slot Interrupt Status not detected : 0x0 1h = Average Bits Per Slot Interrupt Status detected : 0x1
0	TXUNFIS	R/W	0h	Transmit Queue Underflow Interrupt Status This bit indicates that the Transmit Queue had an underflow while transmitting the packet. Transmission is suspended and an Underflow Error TDES3[2] is set. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Transmit Queue Underflow Interrupt Status not detected : 0x0 1h = Transmit Queue Underflow Interrupt Status detected : 0x1

43.7.3.215 MTL_RxQ0_Operation_Mode Register (Offset = D30h) [Reset = 0h]

MTL_RxQ0_Operation_Mode is shown in Figure 43-254 and described in Table 43-308.

Return to the Summary Table.

The Queue 0 Receive Operation Mode register establishes the Receive queue operating modes and command. The RFA and RFD fields are not backward compatible with the RFA and RFD fields of 4.00a release

Figure 43-254. MTL_RxQ0_Operation_Mode Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RQS				RESERVED			RFD
R/W-0h				R-0h			R/W-0h
15	14	13	12	11	10	9	8
RFD		RESERVED			RFA		
R/W-0h		R-0h			R/W-0h		
7	6	5	4	3	2	1	0
EHFC	DIS_TCP_EF	RSF	FEP	FUP	RESERVED	RTC	
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	

Table 43-308. MTL_RxQ0_Operation_Mode Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved.
23-20	RQS	R/W	0h	Receive Queue Size This field indicates the size of the allocated Receive queues in blocks of 256 bytes. The RQS field is read-write only if the number of Rx Queues more than one, the reset value is 0x0 and indicates size of 256 bytes. When the number of Rx Queues is one, the field is read-only and the configured RX FIFO size in blocks of 256 bytes is reflected in the reset value. The width of this field depends on the Rx memory size selected in your configuration. For example, if the memory size is 2048, the width of this field is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits
19-17	RESERVED	R	0h	Reserved.

Table 43-308. MTL_RxQ0_Operation_Mode Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16-14	RFD	R/W	0h	Threshold for Deactivating Flow Control (in half-duplex and full-duplex modes) These bits control the threshold (fill-level of Rx queue) at which the flow control is de-asserted after activation: - 0: Full minus 1 KB, that is, FULL 1 KB - 1: Full minus 1.5 KB, that is, FULL 1.5 KB - 2: Full minus 2 KB, that is, FULL 2 KB - 3: Full minus 2.5 KB, that is, FULL 2.5 KB - ... - 6: Full minus 4 KB, that is, FULL 4 KB - 7: Full minus 4.5 KB, that is, FULL 4.5 KB The de-assertion is effective only after flow control is asserted. Note: The value must be programmed in such a way to make sure that the threshold is a positive number. When the EHFC is set high, these values are applicable only when the Rx queue size determined by the RQS field of this register, is equal to or greater than 4 KB. For a given queue size, the values ranges between 0 and the encoding for FULL minus (QSIZE - 0.5 KB) and all other values are illegal. Here the term FULL and QSIZE refers to the queue size determined by the RQS field of this register. The width of this field depends on RX FIFO size selected during the configuration. Remaining bits are reserved and read only.
13-11	RESERVED	R	0h	Reserved.
10-8	RFA	R/W	0h	Threshold for Activating Flow Control (in half-duplex and full-duplex These bits control the threshold (fill-level of Rx queue) at which the flow control is activated: For more information on encoding for this field, see RFD.
7	EHFC	R/W	0h	Enable Hardware Flow Control When this bit is set, the flow control signal operation, based on the fill-level of Rx queue, is enabled. When reset, the flow control operation is disabled. 0h = Hardware Flow Control is disabled : 0x0 1h = Hardware Flow Control is enabled : 0x1
6	DIS_TCP_EF	R/W	0h	Disable Dropping of TCP/IP Checksum Error Packets When this bit is set, the MAC does not drop the packets which only have the errors detected by the Receive Checksum Offload engine. Such packets have errors only in the encapsulated payload. There are no errors (including FCS error) in the Ethernet packet received by the MAC. When this bit is reset, all error packets are dropped if the FEP bit is reset. 0h = Dropping of TCP/IP Checksum Error Packets is enabled : 0x0 1h = Dropping of TCP/IP Checksum Error Packets is disabled : 0x1
5	RSF	R/W	0h	Receive Queue Store and Forward When this bit is set, the DWC_ether_qos reads a packet from the Rx queue only after the complete packet has been written to it, ignoring the RTC field of this register. When this bit is reset, the Rx queue operates in the Threshold (cut-through) mode, subject to the threshold specified by the RTC field of this register. 0h = Receive Queue Store and Forward is disabled : 0x0 1h = Receive Queue Store and Forward is enabled : 0x1

Table 43-308. MTL_RxQ0_Operation_Mode Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	FEP	R/W	0h	Forward Error Packets When this bit is reset, the Rx queue drops packets with error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, if the start byte (write) pointer of a packet is already transferred to the read controller side (in Threshold mode), the packet is not dropped. When this bit is set, all packets except the runt error packets are forwarded to the application or DMA. If the RSF bit is set and the Rx queue overflows when a partial packet is written, the packet is dropped irrespective of the setting of this bit. However, if the RSF bit is reset and the Rx queue overflows when a partial packet is written, a partial packet may be forwarded to the application or DMA. 0h = Forward Error Packets is disabled : 0x0 1h = Forward Error Packets is enabled : 0x1
3	FUP	R/W	0h	Forward Undersized Good Packets When this bit is set, the Rx queue forwards the undersized good packets (packets with no error and length less than 64 bytes), including pad-bytes and CRC. When this bit is reset, the Rx queue drops all packets of less than 64 bytes, unless a packet is already transferred because of the lower value of Rx Threshold, for example, RTC = 01. 0h = Forward Undersized Good Packets is disabled : 0x0 1h = Forward Undersized Good Packets is enabled : 0x1
2	RESERVED	R	0h	Reserved.
1-0	RTC	R/W	0h	Receive Queue Threshold Control These bits control the threshold level of the MTL Rx queue (in bytes): The received packet is transferred to the application or DMA when the packet size within the MTL Rx queue is larger than the threshold. In addition, full packets with length less than the threshold are automatically transferred. This field is valid only when the RSF bit is zero. This field is ignored when the RSF bit is set to 1. 0h = 64 : 0x0 1h = 32 : 0x1 2h = 96 : 0x2 3h = 128 : 0x3

43.7.3.216 MTL_RxQ0_Missed_Packet_Overflow_Cnt Register (Offset = D34h) [Reset = 0h]

MTL_RxQ0_Missed_Packet_Overflow_Cnt is shown in Figure 43-255 and described in Table 43-309.

Return to the Summary Table.

The Queue 0 Missed Packet and Overflow Counter register contains the counter for packets missed because of Receive queue packet flush and packets discarded because of Receive queue overflow.

Figure 43-255. MTL_RxQ0_Missed_Packet_Overflow_Cnt Register

31	30	29	28	27	26	25	24
RESERVED				MISCNTOVF	MISPKTCNT		
R-0h				R-0h	R-0h		
23	22	21	20	19	18	17	16
MISPKTCNT							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				OVFCNTOVF	OVFPKTCNT		
R-0h				R-0h	R-0h		
7	6	5	4	3	2	1	0
OVFPKTCNT							
R-0h							

Table 43-309. MTL_RxQ0_Missed_Packet_Overflow_Cnt Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved.
27	MISCNTOVF	R	0h	Missed Packet Counter Overflow Bit When set, this bit indicates that the Rx Queue Missed Packet Counter crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = Missed Packet Counter overflow not detected : 0x0 1h = Missed Packet Counter overflow detected : 0x1
26-16	MISPKTCNT	R	0h	Missed Packet Counter This field indicates the number of packets missed by the DWC_ether_qos because the application asserted ari_pkt_flush_i[] for this queue. This counter is reset when this register is read with mci_be_i[0] at 1b1. This counter is incremented by 1 when the DMA discards the packet because of buffer unavailability. Access restriction applies. Clears on read. Self-set to 1 on internal event.
15-12	RESERVED	R	0h	Reserved.
11	OVFCNTOVF	R	0h	Overflow Counter Overflow Bit When set, this bit indicates that the Rx Queue Overflow Packet Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = Overflow Counter overflow not detected : 0x0 1h = Overflow Counter overflow detected : 0x1

Table 43-309. MTL_RxQ0_Missed_Packet_Overflow_Cnt Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10-0	OVFPKTCNT	R	0h	Overflow Packet Counter This field indicates the number of packets discarded by the DWC_ether_qos because of Receive queue overflow. This counter is incremented each time the DWC_ether_qos discards an incoming packet because of overflow. This counter is reset when this register is read with mci_be_i[0] at 1'b1. Access restriction applies. Clears on read. Self-set to 1 on internal event.

43.7.3.217 MTL_RxQ0_Debug Register (Offset = D38h) [Reset = 0h]

MTL_RxQ0_Debug is shown in Figure 43-256 and described in Table 43-310.

Return to the Summary Table.

The Queue 0 Receive Debug register gives the debug status of various blocks related to the Receive queue.

Figure 43-256. MTL_RxQ0_Debug Register

31	30	29	28	27	26	25	24
RESERVED		PRXQ					
R-0h		R-0h					
23	22	21	20	19	18	17	16
PRXQ							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RXQSTS		RESERVED	RRCSTS		RWCSTS
R-0h		R-0h		R-0h	R-0h		R-0h

Table 43-310. MTL_RxQ0_Debug Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved.
29-16	PRXQ	R	0h	Number of Packets in Receive Queue This field indicates the current number of packets in the Rx Queue. The theoretical maximum value for this field is 256KB/16B = 16K Packets, that is, Max_Queue_Size/Min_Packet_Size.
15-6	RESERVED	R	0h	Reserved.
5-4	RXQSTS	R	0h	MTL Rx Queue Fill-Level Status This field gives the status of the fill-level of the Rx Queue: 0h = Rx Queue empty : 0x0 1h = Rx Queue fill-level below flow-control deactivate threshold : 0x1 2h = Rx Queue fill-level above flow-control activate threshold : 0x2 3h = Rx Queue full : 0x3
3	RESERVED	R	0h	Reserved.
2-1	RRCSTS	R	0h	MTL Rx Queue Read Controller State This field gives the state of the Rx queue Read controller: 0h = Idle state : 0x0 1h = Reading packet data : 0x1 2h = Reading packet status (or timestamp) : 0x2 3h = Flushing the packet data and status : 0x3
0	RWCSTS	R	0h	MTL Rx Queue Write Controller Active Status When high, this bit indicates that the MTL Rx queue Write controller is active, and it is transferring a received packet to the Rx Queue. 0h = MTL Rx Queue Write Controller Active Status not detected : 0x0 1h = MTL Rx Queue Write Controller Active Status detected : 0x1

43.7.3.218 MTL_RxQ0_Control Register (Offset = D3Ch) [Reset = 0h]

MTL_RxQ0_Control is shown in Figure 43-257 and described in Table 43-311.

Return to the Summary Table.

The Queue Receive Control register controls the receive arbitration and passing of received packets to the application.

Figure 43-257. MTL_RxQ0_Control Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RXQ_FRM_ARBIT	RXQ_WEGT		
R-0h				R/W-0h	R/W-0h		

Table 43-311. MTL_RxQ0_Control Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved.
3	RXQ_FRM_ARBIT	R/W	0h	Receive Queue Packet Arbitration When this bit is set, the DWC_ether_qos drives the packet data to the ARI interface such that the entire packet data of currently-selected queue is transmitted before switching to other queue. When this bit is reset, the DWC_ether_qos drives the packet data to the ARI interface such that the following amount of data of currently-selected queue is transmitted before switching to other queue: - PBL amount of data (indicated by ari_qN_pbl_i[]) or - Complete data of a packet The status and the timestamp are not a part of the PBL data. Therefore, the DWC_ether_qos drives the complete status (including timestamp status) during first PBL request for the packet (in store-and-forward mode) or the last PBL request for the packet (in Threshold mode). 0h = Receive Queue Packet Arbitration is disabled : 0x0 1h = Receive Queue Packet Arbitration is enabled : 0x1
2-0	RXQ_WEGT	R/W	0h	Receive Queue Weight This field indicates the weight assigned to the Rx Queue 0. The weight is used as the number of continuous PBL or packets requests (depending on the RXQ_FRM_ARBIT) allocated to the queue in one arbitration cycle.

43.7.3.219 MTL_TxQ1_Operation_Mode Register (Offset = D40h) [Reset = 0h]

MTL_TxQ1_Operation_Mode is shown in Figure 43-258 and described in Table 43-312.

Return to the Summary Table.

The Queue 1 Transmit Operation Mode register establishes the Transmit queue operating modes and commands.

Figure 43-258. MTL_TxQ1_Operation_Mode Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED												TQS			
R-0h												R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									TTC			TXQEN		TSF	FTQ
R-0h									R/W-0h			R/W-0h		R/W-0h	R/W-0h

Table 43-312. MTL_TxQ1_Operation_Mode Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved.
19-16	TQS	R/W	0h	Transmit Queue Size This field indicates the size of the allocated Transmit queues in blocks of 256 bytes. The TQS field is read-write only if the number of Tx Queues more than one, the reset value is 0x0 and indicates size of 256 bytes. When the number of Tx Queues is one, the field is read-only and the configured TX FIFO size in blocks of 256 bytes is reflected in the reset value. The width of this field depends on the Tx memory size selected in your configuration. For example, if the memory size is 2048, the width of this field is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits
15-7	RESERVED	R	0h	Reserved.
6-4	TTC	R/W	0h	Transmit Threshold Control These bits control the threshold level of the MTL Tx Queue. The transmission starts when the packet size within the MTL Tx Queue is larger than the threshold. In addition, full packets with length less than the threshold are also transmitted. These bits are used only when the TSF bit is reset. 0h = 32 : 0x0 1h = 64 : 0x1 2h = 96 : 0x2 3h = 128 : 0x3 4h = 192 : 0x4 5h = 256 : 0x5 6h = 384 : 0x6 7h = 512 : 0x7
3-2	TXQEN	R/W	0h	Transmit Queue Enable This field is used to enable/disable the transmit queue 0. - 2'b00: Not enabled - 2'b01: Reserved - 2'b10: Enabled - 2'b11: Reserved Note: In multiple Tx queues configuration, all the queues are disabled by default. Enable the Tx queue by programming this field. 0h = Not enabled : 0x0 1h = Enable in AV mode(Reserved in non-AV) : 0x1 2h = Enabled : 0x2 3h = Reserved : 0x3

Table 43-312. MTL_TxQ1_Operation_Mode Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	TSF	R/W	0h	Transmit Store and Forward When this bit is set, the transmission starts when a full packet resides in the MTL Tx queue. When this bit is set, the TTC values specified in Bits[6:4] of this register are ignored. This bit should be changed only when the transmission is stopped. 0h = Transmit Store and Forward is disabled : 0x0 1h = Transmit Store and Forward is enabled : 0x1
0	FTQ	R/W	0h	Flush Transmit Queue When this bit is set, the Tx queue controller logic is reset to its default values. Therefore, all the data in the Tx queue is lost or flushed. This bit is internally reset when the flushing operation is complete. Until this bit is reset, you should not write to the MTL_TxQ1_Operation_Mode register. The data which is already accepted by the MAC transmitter is not flushed. It is scheduled for transmission and results in underflow and runt packet transmission. Note: The flush operation is complete only when the Tx queue is empty and the application has accepted the pending Tx Status of all transmitted packets. To complete this flush operation, the PHY Tx clock (clk_tx_i) should be active. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0h = Flush Transmit Queue is disabled : 0x0 1h = Flush Transmit Queue is enabled : 0x1

43.7.3.220 MTL_TxQ1_Underflow Register (Offset = D44h) [Reset = 0h]

MTL_TxQ1_Underflow is shown in Figure 43-259 and described in Table 43-313.

Return to the Summary Table.

The Queue 1 Underflow Counter register contains the counter for packets aborted because of Transmit queue underflow and packets missed because of Receive queue packet flush

Figure 43-259. MTL_TxQ1_Underflow Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				UFCNTOVF	UFFRMCNT		
R-0h				R-0h	R-0h		
7	6	5	4	3	2	1	0
UFFRMCNT							
R-0h							

Table 43-313. MTL_TxQ1_Underflow Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved.
11	UFCNTOVF	R	0h	Overflow Bit for Underflow Packet Counter This bit is set every time the Tx queue Underflow Packet Counter field overflows, that is, it has crossed the maximum count. In such a scenario, the overflow packet counter is reset to all-zeros and this bit indicates that the rollover happened. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = Overflow not detected for Underflow Packet Counter : 0x0 1h = Overflow detected for Underflow Packet Counter : 0x1
10-0	UFFRMCNT	R	0h	Underflow Packet Counter This field indicates the number of packets aborted by the controller because of Tx Queue Underflow. This counter is incremented each time the MAC aborts outgoing packet because of underflow. The counter is cleared when this register is read with mci_be_i[0] at 1'b1. Access restriction applies. Clears on read. Self-set to 1 on internal event.

43.7.3.221 MTL_TxQ1_Debug Register (Offset = D48h) [Reset = 0h]

MTL_TxQ1_Debug is shown in Figure 43-260 and described in Table 43-314.

Return to the Summary Table.

The Queue 1 Transmit Debug register gives the debug status of various blocks related to the Transmit queue.

Figure 43-260. MTL_TxQ1_Debug Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED	STXSTSF			RESERVED	PTXQ		
R-0h	R-0h			R-0h	R-0h		
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		TXSTSFSTS	TXQSTS	TWCSTS	TRCSTS		TXQPAUSED
R-0h		R-0h	R-0h	R-0h	R-0h		R-0h

Table 43-314. MTL_TxQ1_Debug Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved.
22-20	STXSTSF	R	0h	Number of Status Words in Tx Status FIFO of Queue This field indicates the current number of status in the Tx Status FIFO of this queue. When the DTXSTS bit of MTL_Operation_Mode register is set to 1, this field does not reflect the number of status words in Tx Status FIFO.
19	RESERVED	R	0h	Reserved.
18-16	PTXQ	R	0h	Number of Packets in the Transmit Queue This field indicates the current number of packets in the Tx Queue. When the DTXSTS bit of MTL_Operation_Mode register is set to 1, this field does not reflect the number of packets in the Transmit queue.
15-6	RESERVED	R	0h	Reserved.
5	TXSTSFSTS	R	0h	MTL Tx Status FIFO Full Status When high, this bit indicates that the MTL Tx Status FIFO is full. Therefore, the MTL cannot accept any more packets for transmission. 0h = MTL Tx Status FIFO Full status is not detected : 0x0 1h = MTL Tx Status FIFO Full status is detected : 0x1
4	TXQSTS	R	0h	MTL Tx Queue Not Empty Status When this bit is high, it indicates that the MTL Tx Queue is not empty and some data is left for transmission. 0h = MTL Tx Queue Not Empty status is not detected : 0x0 1h = MTL Tx Queue Not Empty status is detected : 0x1
3	TWCSTS	R	0h	MTL Tx Queue Write Controller Status When high, this bit indicates that the MTL Tx Queue Write Controller is active, and it is transferring the data to the Tx Queue. 0h = MTL Tx Queue Write Controller status is not detected : 0x0 1h = MTL Tx Queue Write Controller status is detected : 0x1

Table 43-314. MTL_TxQ1_Debug Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-1	TRCSTS	R	0h	MTL Tx Queue Read Controller Status This field indicates the state of the Tx Queue Read Controller: 0h = Idle state : 0x0 1h = Read state (transferring data to the MAC transmitter) : 0x1 2h = Waiting for pending Tx Status from the MAC transmitter : 0x2 3h = Flushing the Tx queue because of the Packet Abort request from the MAC : 0x3
0	TXQPAUSED	R	0h	Transmit Queue in Pause When this bit is high and the Rx flow control is enabled, it indicates that the Tx Queue is in the Pause condition (in the full-duplex only mode) because of the following: - Reception of the PFC packet for the priorities assigned to the Tx Queue when PFC is enabled - Reception of 802.3x Pause packet when PFC is disabled 0h = Transmit Queue in Pause status is not detected : 0x0 1h = Transmit Queue in Pause status is detected : 0x1

43.7.3.222 MTL_TxQ1_ETS_Status Register (Offset = D54h) [Reset = 0h]

MTL_TxQ1_ETS_Status is shown in Figure 43-261 and described in Table 43-315.

Return to the Summary Table.

The Queue 1 ETS Status register provides the average traffic transmitted in Queue 1.

Figure 43-261. MTL_TxQ1_ETS_Status Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								ABS																							
R-0h								R-0h																							

Table 43-315. MTL_TxQ1_ETS_Status Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved.
23-0	ABS	R	0h	Average Bits per Slot This field contains the average transmitted bits per slot. If AV operation is enabled, this field is computed over number of slots, programmed in the SLC field of MTL_TxQ[n]_ETS_CONTROL register. When Enable Slot Interval feature is selected, the maximum value of this field is 0x6_4000 in 100 Mbps, 0x3E_8000 in 1000 Mbps and 9C_4000 in 2500 Mbps mode respectively. Otherwise, the maximum value of this field is 0x30D4 in 100 Mbs, 0x1E848 in 1000 Mbps and 0x4C4B4 in 2500 Mbps respectively. When the DCB operation is enabled for Queue, this field is computed over every 10 million bit times slot (4 ms in 2500 Mbps 10 ms in 1000 Mbps 100 ms in 100 Mbps). The maximum value is 0x989680.

43.7.3.223 MTL_TxQ1_Quantum_Weight Register (Offset = D58h) [Reset = 0h]

MTL_TxQ1_Quantum_Weight is shown in Figure 43-262 and described in Table 43-316.

Return to the Summary Table.

The Queue 1 idleSlopeCredit, Quantum or Weights register provides the average traffic transmitted in Queue 1.

Figure 43-262. MTL_TxQ1_Quantum_Weight Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											ISCQW																				
R-0h											R/W-0h																				

Table 43-316. MTL_TxQ1_Quantum_Weight Register Field Descriptions

Bit	Field	Type	Reset	Description
31-21	RESERVED	R	0h	Reserved.
20-0	ISCQW	R/W	0h	idleSlopeCredit, Quantum or Weights - idleSlopeCredit When AV feature is enabled, this field contains the idleSlopeCredit value required for the credit-based shaper algorithm for Queue 1. This is the rate of change of credit in bits per cycle (40 ns for 100 Mbps 8 ns for 1000 Mbps 3.2 ns for 2500 Mbps) when the credit is increasing. The software should program this field with computed credit in bits per cycle scaled by 1,024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbps mode and 0x1000 in 100 Mbps mode. Bits[20:14] must be written to zero. - Quantum When the DCB operation is enabled with DWRR algorithm for Queue 1 traffic, this field contains the quantum value in bytes to be added to credit during every queue scanning cycle. The maximum value is 0x1312D0 bytes. - Weights When DCB operation is enabled with WFQ algorithm for Queue 1 traffic, this field contains the weight for this queue. The maximum value is 0x3FFF where weight of 0 indicates 100% bandwidth. Bits[20:14] must be written to zero. When DCB operation or generic queuing operation is enabled with WRR algorithm for Queue 1 traffic, this field contains the weight for this queue. The maximum value is 0x64. Bits [20:7] must be written to zero. - Note 1: In multiple Queue configuration this field in respective per queue register must be programmed to some non-zero value when multiple queues are enabled or single queue other than Q0 is enabled. This field need not be programmed when only Q0 is enabled. In general, when WRR algorithm is selected a non-zero value must be programmed on both Receive and Transmit. In Receive, the register is MTL_Operation_Mode register. - Note 2: For WFQ algorithm, higher the programmed weights lesser the bandwidth allocated for that Transmit Queue. The finish time is not a function of particular packet alone but it is as per the formula: (previous_finish_time of particular Transmit Queue + (weights*packet_size)) - Note 3: The weights programmed do not correspond to the number of packets but the fraction of bandwidth or time allocated for particular queue w.r.t. total BW or time.

43.7.3.224 MTL_Q1_Interrupt_Control_Status Register (Offset = D6Ch) [Reset = 0h]

MTL_Q1_Interrupt_Control_Status is shown in Figure 43-263 and described in Table 43-317.

Return to the Summary Table.

This register contains the interrupt enable and status bits for the queue 1 interrupts.

Figure 43-263. MTL_Q1_Interrupt_Control_Status Register

31	30	29	28	27	26	25	24
RESERVED							RXOIE
R-0h							R/W-0h
23	22	21	20	19	18	17	16
RESERVED							RXOVFIS
R-0h							R/W-0h
15	14	13	12	11	10	9	8
RESERVED						ABPSIE	TXUIE
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED						ABPSIS	TXUNFIS
R-0h						R/W-0h	R/W-0h

Table 43-317. MTL_Q1_Interrupt_Control_Status Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved.
24	RXOIE	R/W	0h	Receive Queue Overflow Interrupt Enable When this bit is set, the Receive Queue Overflow interrupt is enabled. When this bit is reset, the Receive Queue Overflow interrupt is disabled. 0h = Receive Queue Overflow Interrupt is disabled : 0x0 1h = Receive Queue Overflow Interrupt is enabled : 0x1
23-17	RESERVED	R	0h	Reserved.
16	RXOVFIS	R/W	0h	Receive Queue Overflow Interrupt Status This bit indicates that the Receive Queue had an overflow while receiving the packet. If a partial packet is transferred to the application, the overflow status is set in RDES3[21]. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Receive Queue Overflow Interrupt Status not detected : 0x0 1h = Receive Queue Overflow Interrupt Status detected : 0x1
15-10	RESERVED	R	0h	Reserved.
9	ABPSIE	R/W	0h	Average Bits Per Slot Interrupt Enable When this bit is set, the MAC asserts the sbd_intr_o or mci_intr_o interrupt when the average bits per slot status is updated. When this bit is cleared, the interrupt is not asserted for such an event. 0h = Average Bits Per Slot Interrupt is disabled : 0x0 1h = Average Bits Per Slot Interrupt is enabled : 0x1
8	TXUIE	R/W	0h	Transmit Queue Underflow Interrupt Enable When this bit is set, the Transmit Queue Underflow interrupt is enabled. When this bit is reset, the Transmit Queue Underflow interrupt is disabled. 0h = Transmit Queue Underflow Interrupt Status is disabled : 0x0 1h = Transmit Queue Underflow Interrupt Status is enabled : 0x1
7-2	RESERVED	R	0h	Reserved.

Table 43-317. MTL_Q1_Interrupt_Control_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	ABPSIS	R/W	0h	Average Bits Per Slot Interrupt Status When set, this bit indicates that the MAC has updated the ABS value. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Average Bits Per Slot Interrupt Status not detected : 0x0 1h = Average Bits Per Slot Interrupt Status detected : 0x1
0	TXUNFIS	R/W	0h	Transmit Queue Underflow Interrupt Status This bit indicates that the Transmit Queue had an underflow while transmitting the packet. Transmission is suspended and an Underflow Error TDES3[2] is set. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Transmit Queue Underflow Interrupt Status not detected : 0x0 1h = Transmit Queue Underflow Interrupt Status detected : 0x1

43.7.3.225 MTL_RxQ1_Operation_Mode Register (Offset = D70h) [Reset = 0h]

MTL_RxQ1_Operation_Mode is shown in Figure 43-264 and described in Table 43-318.

Return to the Summary Table.

The Queue 1 Receive Operation Mode register establishes the Receive queue operating modes and command. The RFA and RFD fields are not backward compatible with the RFA and RFD fields of 4.00a release

Figure 43-264. MTL_RxQ1_Operation_Mode Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RQS				RESERVED			RFD
R/W-0h				R-0h			R/W-0h
15	14	13	12	11	10	9	8
RFD		RESERVED			RFA		
R/W-0h		R-0h			R/W-0h		
7	6	5	4	3	2	1	0
EHFC	DIS_TCP_EF	RSF	FEP	FUP	RESERVED	RTC	
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	

Table 43-318. MTL_RxQ1_Operation_Mode Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved.
23-20	RQS	R/W	0h	Receive Queue Size This field indicates the size of the allocated Receive queues in blocks of 256 bytes. The RQS field is read-write only if the number of Rx Queues more than one, the reset value is 0x0 and indicates size of 256 bytes. When the number of Rx Queues is one, the field is read-only and the configured RX FIFO size in blocks of 256 bytes is reflected in the reset value. The width of this field depends on the Rx memory size selected in your configuration. For example, if the memory size is 2048, the width of this field is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits
19-17	RESERVED	R	0h	Reserved.

Table 43-318. MTL_RxQ1_Operation_Mode Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16-14	RFD	R/W	0h	Threshold for Deactivating Flow Control (in half-duplex and full-duplex modes) These bits control the threshold (fill-level of Rx queue) at which the flow control is de-asserted after activation: - 0: Full minus 1 KB, that is, FULL 1 KB - 1: Full minus 1.5 KB, that is, FULL 1.5 KB - 2: Full minus 2 KB, that is, FULL 2 KB - 3: Full minus 2.5 KB, that is, FULL 2.5 KB - ... - 6: Full minus 4 KB, that is, FULL 4 KB - 7: Full minus 4.5 KB, that is, FULL 4.5 KB The de-assertion is effective only after flow control is asserted. Note: The value must be programmed in such a way to make sure that the threshold is a positive number. When the EHFC is set high, these values are applicable only when the Rx queue size determined by the RQS field of this register, is equal to or greater than 4 KB. For a given queue size, the values ranges between 0 and the encoding for FULL minus (QSIZE - 0.5 KB) and all other values are illegal. Here the term FULL and QSIZE refers to the queue size determined by the RQS field of this register. The width of this field depends on RX FIFO size selected during the configuration. Remaining bits are reserved and read only.
13-11	RESERVED	R	0h	Reserved.
10-8	RFA	R/W	0h	Threshold for Activating Flow Control (in half-duplex and full-duplex These bits control the threshold (fill-level of Rx queue) at which the flow control is activated: For more information on encoding for this field, see RFD.
7	EHFC	R/W	0h	Enable Hardware Flow Control When this bit is set, the flow control signal operation, based on the fill-level of Rx queue, is enabled. When reset, the flow control operation is disabled. 0h = Hardware Flow Control is disabled : 0x0 1h = Hardware Flow Control is enabled : 0x1
6	DIS_TCP_EF	R/W	0h	Disable Dropping of TCP/IP Checksum Error Packets When this bit is set, the MAC does not drop the packets which only have the errors detected by the Receive Checksum Offload engine. Such packets have errors only in the encapsulated payload. There are no errors (including FCS error) in the Ethernet packet received by the MAC. When this bit is reset, all error packets are dropped if the FEP bit is reset. 0h = Dropping of TCP/IP Checksum Error Packets is enabled : 0x0 1h = Dropping of TCP/IP Checksum Error Packets is disabled : 0x1
5	RSF	R/W	0h	Receive Queue Store and Forward When this bit is set, the DWC_ether_qos reads a packet from the Rx queue only after the complete packet has been written to it, ignoring the RTC field of this register. When this bit is reset, the Rx queue operates in the Threshold (cut-through) mode, subject to the threshold specified by the RTC field of this register. 0h = Receive Queue Store and Forward is disabled : 0x0 1h = Receive Queue Store and Forward is enabled : 0x1

Table 43-318. MTL_RxQ1_Operation_Mode Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	FEP	R/W	0h	Forward Error Packets When this bit is reset, the Rx queue drops packets with error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, if the start byte (write) pointer of a packet is already transferred to the read controller side (in Threshold mode), the packet is not dropped. When this bit is set, all packets except the runt error packets are forwarded to the application or DMA. If the RSF bit is set and the Rx queue overflows when a partial packet is written, the packet is dropped irrespective of the setting of this bit. However, if the RSF bit is reset and the Rx queue overflows when a partial packet is written, a partial packet may be forwarded to the application or DMA. 0h = Forward Error Packets is disabled : 0x0 1h = Forward Error Packets is enabled : 0x1
3	FUP	R/W	0h	Forward Undersized Good Packets When this bit is set, the Rx queue forwards the undersized good packets (packets with no error and length less than 64 bytes), including pad-bytes and CRC. When this bit is reset, the Rx queue drops all packets of less than 64 bytes, unless a packet is already transferred because of the lower value of Rx Threshold, for example, RTC = 01. 0h = Forward Undersized Good Packets is disabled : 0x0 1h = Forward Undersized Good Packets is enabled : 0x1
2	RESERVED	R	0h	Reserved.
1-0	RTC	R/W	0h	Receive Queue Threshold Control These bits control the threshold level of the MTL Rx queue (in bytes): The received packet is transferred to the application or DMA when the packet size within the MTL Rx queue is larger than the threshold. In addition, full packets with length less than the threshold are automatically transferred. This field is valid only when the RSF bit is zero. This field is ignored when the RSF bit is set to 1. 0h = 64 : 0x0 1h = 32 : 0x1 2h = 96 : 0x2 3h = 128 : 0x3

43.7.3.226 MTL_RxQ1_Missed_Packet_Overflow_Cnt Register (Offset = D74h) [Reset = 0h]

MTL_RxQ1_Missed_Packet_Overflow_Cnt is shown in Figure 43-265 and described in Table 43-319.

Return to the Summary Table.

The Queue 1 Missed Packet and Overflow Counter register contains the counter for packets missed because of Receive queue packet flush and packets discarded because of Receive queue overflow.

Figure 43-265. MTL_RxQ1_Missed_Packet_Overflow_Cnt Register

31	30	29	28	27	26	25	24
RESERVED				MISCNTOVF	MISPKTCNT		
R-0h				R-0h	R-0h		
23	22	21	20	19	18	17	16
MISPKTCNT							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				OVFCNTOVF	OVFPKTCNT		
R-0h				R-0h	R-0h		
7	6	5	4	3	2	1	0
OVFPKTCNT							
R-0h							

Table 43-319. MTL_RxQ1_Missed_Packet_Overflow_Cnt Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved.
27	MISCNTOVF	R	0h	Missed Packet Counter Overflow Bit When set, this bit indicates that the Rx Queue Missed Packet Counter crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = Missed Packet Counter overflow not detected : 0x0 1h = Missed Packet Counter overflow detected : 0x1
26-16	MISPKTCNT	R	0h	Missed Packet Counter This field indicates the number of packets missed by the DWC_ether_qos because the application asserted ari_pkt_flush_i[] for this queue. This counter is reset when this register is read with mci_be_i[0] at 1b1. This counter is incremented by 1 when the DMA discards the packet because of buffer unavailability. Access restriction applies. Clears on read. Self-set to 1 on internal event.
15-12	RESERVED	R	0h	Reserved.
11	OVFCNTOVF	R	0h	Overflow Counter Overflow Bit When set, this bit indicates that the Rx Queue Overflow Packet Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. 0h = Overflow Counter overflow not detected : 0x0 1h = Overflow Counter overflow detected : 0x1

Table 43-319. MTL_RxQ1_Missed_Packet_Overflow_Cnt Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10-0	OVFPKTCNT	R	0h	Overflow Packet Counter This field indicates the number of packets discarded by the DWC_ether_qos because of Receive queue overflow. This counter is incremented each time the DWC_ether_qos discards an incoming packet because of overflow. This counter is reset when this register is read with mci_be_i[0] at 1'b1. Access restriction applies. Clears on read. Self-set to 1 on internal event.

43.7.3.227 MTL_RxQ1_Debug Register (Offset = D78h) [Reset = 0h]

MTL_RxQ1_Debug is shown in Figure 43-266 and described in Table 43-320.

Return to the Summary Table.

The Queue 1 Receive Debug register gives the debug status of various blocks related to the Receive queue.

Figure 43-266. MTL_RxQ1_Debug Register

31	30	29	28	27	26	25	24
RESERVED		PRXQ					
R-0h		R-0h					
23	22	21	20	19	18	17	16
PRXQ							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RXQSTS		RESERVED	RRCSTS		RWCSTS
R-0h		R-0h		R-0h	R-0h		R-0h

Table 43-320. MTL_RxQ1_Debug Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved.
29-16	PRXQ	R	0h	Number of Packets in Receive Queue This field indicates the current number of packets in the Rx Queue. The theoretical maximum value for this field is 256KB/16B = 16K Packets, that is, Max_Queue_Size/Min_Packet_Size.
15-6	RESERVED	R	0h	Reserved.
5-4	RXQSTS	R	0h	MTL Rx Queue Fill-Level Status This field gives the status of the fill-level of the Rx Queue: 0h = Rx Queue empty : 0x0 1h = Rx Queue fill-level below flow-control deactivate threshold : 0x1 2h = Rx Queue fill-level above flow-control activate threshold : 0x2 3h = Rx Queue full : 0x3
3	RESERVED	R	0h	Reserved.
2-1	RRCSTS	R	0h	MTL Rx Queue Read Controller State This field gives the state of the Rx queue Read controller: 0h = Idle state : 0x0 1h = Reading packet data : 0x1 2h = Reading packet status (or timestamp) : 0x2 3h = Flushing the packet data and status : 0x3
0	RWCSTS	R	0h	MTL Rx Queue Write Controller Active Status When high, this bit indicates that the MTL Rx queue Write controller is active, and it is transferring a received packet to the Rx Queue. 0h = MTL Rx Queue Write Controller Active Status not detected : 0x0 1h = MTL Rx Queue Write Controller Active Status detected : 0x1

43.7.3.228 MTL_RxQ1_Control Register (Offset = D7Ch) [Reset = 0h]

MTL_RxQ1_Control is shown in Figure 43-267 and described in Table 43-321.

Return to the Summary Table.

The Queue Receive Control register controls the receive arbitration and passing of received packets to the application.

Figure 43-267. MTL_RxQ1_Control Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RXQ_FRM_ARBIT	RXQ_WEGT		
R-0h				R/W-0h	R/W-0h		

Table 43-321. MTL_RxQ1_Control Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved.
3	RXQ_FRM_ARBIT	R/W	0h	Receive Queue Packet Arbitration When this bit is set, the DWC_ether_qos drives the packet data to the ARI interface such that the entire packet data of currently-selected queue is transmitted before switching to other queue. When this bit is reset, the DWC_ether_qos drives the packet data to the ARI interface such that the following amount of data of currently-selected queue is transmitted before switching to other queue: - PBL amount of data (indicated by ari_qN_pbl_i[]) or - Complete data of a packet The status and the timestamp are not a part of the PBL data. Therefore, the DWC_ether_qos drives the complete status (including timestamp status) during first PBL request for the packet (in store-and-forward mode) or the last PBL request for the packet (in Threshold mode). 0h = Receive Queue Packet Arbitration is disabled : 0x0 1h = Receive Queue Packet Arbitration is enabled : 0x1
2-0	RXQ_WEGT	R/W	0h	Receive Queue Weight This field indicates the weight assigned to the Rx Queue 0. The weight is used as the number of continuous PBL or packets requests (depending on the RXQ_FRM_ARBIT) allocated to the queue in one arbitration cycle.

43.7.3.229 DMA_Mode Register (Offset = 1000h) [Reset = 0h]

DMA_Mode is shown in Figure 43-268 and described in Table 43-322.

Return to the Summary Table.

The Bus Mode register establishes the bus operating modes for the DMA.

Figure 43-268. DMA_Mode Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						INTM	
R-0h						R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	PR			TXPR	RESERVED		RESERVED
R-0h	R/W-0h			R/W-0h	R-0h		R-0h
7	6	5	4	3	2	1	0
RESERVED			TAA			DA	SWR
R-0h			R/W-0h			R/W-0h	R/W-0h

Table 43-322. DMA_Mode Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved.
17-16	INTM	R/W	0h	Interrupt Mode This field defines the interrupt mode of DWC_ether_qos. The behavior of the following outputs changes depending on the following settings: - sbd_perch_tx_intr_o[] (Transmit Per Channel Interrupt) - sbd_perch_rx_intr_o[] (Receive Per Channel Interrupt) - sbd_intr_o (Common Interrupt) It also changes the behavior of the RI/TI bits in the DMA_CH0_Status. - 00: sbd_perch_* are pulse signals for each completion events. sbd_intr_o is also asserted and cleared only when software clears the corresponding RI/TI status bits - 01: sbd_perch_* are level signals asserted on corresponding event and de-asserted when the software clears the corresponding RI/TI status bits. The sbd_intr_o is not asserted for these packet transfer completion events. - 10: sbd_perch_* are level signals asserted on corresponding event and de-asserted when the software clears the corresponding RI/TI status bits. However, the signal is asserted again if the same event occurred again before it was cleared. The sbd_intr_o is not asserted for these packet transfer completion events. - 11: Reserved For more details please refer Table "DWC_ether_qos Transfer Complete Interrupt Behavior". 0h = See above description : 0x0 1h = See above description : 0x1 2h = See above description : 0x2 3h = Reserved : 0x3
15	RESERVED	R	0h	Reserved.

Table 43-322. DMA_Mode Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14-12	PR	R/W	0h	Priority Ratio These bits control the priority ratio in weighted round-robin arbitration between the Rx DMA and Tx DMA. These bits are valid only when the DA bit is reset. The priority ratio is Rx:Tx or Tx:Rx depending on whether the TXPR bit is reset or set. 0h = The priority ratio is 1:1 : 0x0 1h = The priority ratio is 2:1 : 0x1 2h = The priority ratio is 3:1 : 0x2 3h = The priority ratio is 4:1 : 0x3 4h = The priority ratio is 5:1 : 0x4 5h = The priority ratio is 6:1 : 0x5 6h = The priority ratio is 7:1 : 0x6 7h = The priority ratio is 8:1 : 0x7
11	TXPR	R/W	0h	Transmit Priority When set, this bit indicates that the Tx DMA has higher priority than the Rx DMA during arbitration for the system-side bus. 0h = Transmit Priority is disabled : 0x0 1h = Transmit Priority is enabled : 0x1
10-9	RESERVED	R	0h	Reserved.
8	RESERVED	R	0h	Reserved.
7-5	RESERVED	R	0h	Reserved.
4-2	TAA	R/W	0h	Transmit Arbitration Algorithm This field is used to select the arbitration algorithm for the Transmit side when multiple Tx DMAs are selected. 0h = Fixed priority(Channel 0 has the lowest priority and the last channel has the highest priority) : 0x0 1h = Weighted Strict Priority (WSP) : 0x1 2h = Weighted Round-Robin (WRR) : 0x2 3h = Reserved (for 3'b011 to 3'b111) : 0x3
1	DA	R/W	0h	DMA Tx or Rx Arbitration Scheme This bit specifies the arbitration scheme between the Transmit and Receive paths of all channels: - 0: Weighted Round-Robin with Rx:Tx or Tx:Rx The priority between the paths is according to the priority specified in Bits[14:12] and the priority weight is specified in the TXPR bit. - 1: Fixed Priority The Tx path has priority over the Rx path when the TXPR bit is set. Otherwise, the Rx path has priority over the Tx path. 0h = Weighted Round-Robin with Rx:Tx or Tx:Rx : 0x0 1h = Fixed Priority : 0x1
0	SWR	R/W	0h	Software Reset When this bit is set, the MAC and the DMA controller reset the logic and all internal registers of the DMA, MTL, and MAC. This bit is automatically cleared after the reset operation is complete in all DWC_ether_qos clock domains. Before reprogramming any DWC_ether_qos register, a value of zero should be read in this bit. This bit must be read at least 4 CSR clock cycles after it is written to 1. Note: The reset operation is complete only when all resets in all active clock domains are de-asserted. Therefore, it is essential that all PHY inputs clocks (applicable for the selected PHY interface) are present for software reset completion. The time to complete the software reset operation depends on the frequency of the slowest active clock. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. 0h = Software Reset is disabled : 0x0 1h = Software Reset is enabled : 0x1

43.7.3.230 DMA_SysBus_Mode Register (Offset = 1004h) [Reset = 0h]

DMA_SysBus_Mode is shown in Figure 43-269 and described in Table 43-323.

Return to the Summary Table.

The System Bus mode register controls the behavior of the AHB or AXI master. It mainly controls burst splitting and number of outstanding requests.

Figure 43-269. DMA_SysBus_Mode Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED				RESERVED	
R-0h	R-0h	R-0h				R-0h	
23	22	21	20	19	18	17	16
RESERVED						RESERVED	
R-0h						R-0h	
15	14	13	12	11	10	9	8
RB	MB	RESERVED	AAL	RESERVED	RESERVED	RESERVED	
R/W-0h	R/W-0h	R-0h	R/W-0h	R-0h	R-0h	R-0h	
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	FB
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R/W-0h

Table 43-323. DMA_SysBus_Mode Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved.
30	RESERVED	R	0h	Reserved.
29-26	RESERVED	R	0h	Reserved.
25-24	RESERVED	R	0h	Reserved.
23-18	RESERVED	R	0h	Reserved.
17-16	RESERVED	R	0h	Reserved.
15	RB	R/W	0h	Rebuild INCRx Burst When this bit is set high and the AHB master gets SPLIT, RETRY, or Early Burst Termination (EBT) response, the AHB master interface rebuilds the pending beats of any initiated burst transfer with INCRx and SINGLE transfers. By default, the AHB master interface rebuilds pending beats of an EBT with an unspecified (INCR) burst. 0h = Rebuild INCRx Burst is disabled : 0x0 1h = Rebuild INCRx Burst is enabled : 0x1
14	MB	R/W	0h	Mixed Burst When this bit is set high and the FB bit is low, the AHB master performs undefined bursts transfers (INCR) for burst length of 16 or more. For burst length of 16 or less, the AHB master performs fixed burst transfers (INCRx and SINGLE). 0h = Mixed Burst is disabled : 0x0 1h = Mixed Burst is enabled : 0x1
13	RESERVED	R	0h	Reserved.
12	AAL	R/W	0h	Address-Aligned Beats When this bit is set to 1, the EQOS-AXI or EQOS-AHB master performs address-aligned burst transfers on Read and Write channels. 0h = Address-Aligned Beats is disabled : 0x0 1h = Address-Aligned Beats is enabled : 0x1
11	RESERVED	R	0h	Reserved.
10	RESERVED	R	0h	Reserved.

Table 43-323. DMA_SysBus_Mode Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9-8	RESERVED	R	0h	Reserved.
7	RESERVED	R	0h	Reserved.
6	RESERVED	R	0h	Reserved.
5	RESERVED	R	0h	Reserved.
4	RESERVED	R	0h	Reserved.
3	RESERVED	R	0h	Reserved.
2	RESERVED	R	0h	Reserved.
1	RESERVED	R	0h	Reserved.
0	FB	R/W	0h	Fixed Burst Length When this bit is set to 1, the AHB master initiates burst transfers of specified length (INCRx or SINGLE). When this bit is set to 0, the AHB master initiates transfers of unspecified length (INCR) or SINGLE transfers. 0h = Fixed Burst Length is disabled : 0x0 1h = Fixed Burst Length is enabled : 0x1

43.7.3.231 DMA_Interrupt_Status Register (Offset = 1008h) [Reset = 0h]

DMA_Interrupt_Status is shown in Figure 43-270 and described in Table 43-324.

Return to the Summary Table.

The application reads this Interrupt Status register during interrupt service routine or polling to determine the interrupt status of DMA channels, MTL queues, and the MAC.

Figure 43-270. DMA_Interrupt_Status Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						MACIS	MTLIS
R-0h						R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	DC1IS	DC0IS
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 43-324. DMA_Interrupt_Status Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved.
17	MACIS	R	0h	MAC Interrupt Status This bit indicates an interrupt event in the MAC. To reset this bit to 1'b0, the software must read the corresponding register in the MAC to get the exact cause of the interrupt and clear its source. 0h = MAC Interrupt Status not detected : 0x0 1h = MAC Interrupt Status detected : 0x1
16	MTLIS	R	0h	MTL Interrupt Status This bit indicates an interrupt event in the MTL. To reset this bit to 1'b0, the software must read the corresponding register in the MTL to get the exact cause of the interrupt and clear its source. 0h = MTL Interrupt Status not detected : 0x0 1h = MTL Interrupt Status detected : 0x1
15-8	RESERVED	R	0h	Reserved.
7	RESERVED	R	0h	Reserved.
6	RESERVED	R	0h	Reserved.
5	RESERVED	R	0h	Reserved.
4	RESERVED	R	0h	Reserved.
3	RESERVED	R	0h	Reserved.
2	RESERVED	R	0h	Reserved.
1	DC1IS	R	0h	DMA Channel 1 Interrupt Status This bit indicates an interrupt event in DMA Channel 1. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 1 to get the exact cause of the interrupt and clear its source. 0h = DMA Channel 1 Interrupt Status not detected : 0x0 1h = DMA Channel 1 Interrupt Status detected : 0x1

Table 43-324. DMA_Interrupt_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	DC0IS	R	0h	DMA Channel 0 Interrupt Status This bit indicates an interrupt event in DMA Channel 0. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 0 to get the exact cause of the interrupt and clear its source. 0h = DMA Channel 0 Interrupt Status not detected : 0x0 1h = DMA Channel 0 Interrupt Status detected : 0x1

43.7.3.232 DMA_Debug_Status0 Register (Offset = 100Ch) [Reset = 0h]

DMA_Debug_Status0 is shown in Figure 43-271 and described in Table 43-325.

Return to the Summary Table.

The Debug Status 0 register gives the Receive and Transmit process status for DMA Channel 0-Channel 2 for debugging purpose.

Figure 43-271. DMA_Debug_Status0 Register

31	30	29	28	27	26	25	24
RESERVED				RESERVED			
R-0h				R-0h			
23	22	21	20	19	18	17	16
TPS1				RPS1			
R-0h				R-0h			
15	14	13	12	11	10	9	8
TPS0				RPS0			
R-0h				R-0h			
7	6	5	4	3	2	1	0
RESERVED						RESERVED	AXWHSTS
R-0h						R-0h	R-0h

Table 43-325. DMA_Debug_Status0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved.
27-24	RESERVED	R	0h	Reserved.
23-20	TPS1	R	0h	DMA Channel 1 Transmit Process State This field indicates the Tx DMA FSM state for Channel 1. The MSB of this field always returns 0. This field does not generate an interrupt. 0h = Stopped (Reset or Stop Transmit Command issued) : 0x0 1h = Running (Fetching Tx Transfer Descriptor) : 0x1 2h = Running (Waiting for status) : 0x2 3h = Running (Reading Data from system memory buffer and queuing it to the Tx buffer (Tx FIFO)) : 0x3 4h = Timestamp write state : 0x4 5h = Reserved for future use : 0x5 6h = Suspended (Tx Descriptor Unavailable or Tx Buffer Underflow) : 0x6 7h = Running (Closing Tx Descriptor) : 0x7
19-16	RPS1	R	0h	DMA Channel 1 Receive Process State This field indicates the Rx DMA FSM state for Channel 1. The MSB of this field always returns 0. This field does not generate an interrupt. 0h = Stopped (Reset or Stop Receive Command issued) : 0x0 1h = Running (Fetching Rx Transfer Descriptor) : 0x1 2h = Reserved for future use : 0x2 3h = Running (Waiting for Rx packet) : 0x3 4h = Suspended (Rx Descriptor Unavailable) : 0x4 5h = Running (Closing the Rx Descriptor) : 0x5 6h = Timestamp write state : 0x6 7h = Running (Transferring the received packet data from the Rx buffer to the system memory) : 0x7

Table 43-325. DMA_Debug_Status0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-12	TPS0	R	0h	DMA Channel 0 Transmit Process State This field indicates the Tx DMA FSM state for Channel 0. The MSB of this field always returns 0. This field does not generate an interrupt. 0h = Stopped (Reset or Stop Transmit Command issued) : 0x0 1h = Running (Fetching Tx Transfer Descriptor) : 0x1 2h = Running (Waiting for status) : 0x2 3h = Running (Reading Data from system memory buffer and queuing it to the Tx buffer (Tx FIFO)) : 0x3 4h = Timestamp write state : 0x4 5h = Reserved for future use : 0x5 6h = Suspended (Tx Descriptor Unavailable or Tx Buffer Underflow) : 0x6 7h = Running (Closing Tx Descriptor) : 0x7
11-8	RPS0	R	0h	DMA Channel 0 Receive Process State This field indicates the Rx DMA FSM state for Channel 0. The MSB of this field always returns 0. This field does not generate an interrupt. 0h = Stopped (Reset or Stop Receive Command issued) : 0x0 1h = Running (Fetching Rx Transfer Descriptor) : 0x1 2h = Reserved for future use : 0x2 3h = Running (Waiting for Rx packet) : 0x3 4h = Suspended (Rx Descriptor Unavailable) : 0x4 5h = Running (Closing the Rx Descriptor) : 0x5 6h = Timestamp write state : 0x6 7h = Running (Transferring the received packet data from the Rx buffer to the system memory) : 0x7
7-2	RESERVED	R	0h	Reserved.
1	RESERVED	R	0h	Reserved.
0	AXWHSTS	R	0h	AHB Master Status When high, this bit indicates that the AHB master FSMs are in the non-idle state. 0h = AXI Master Write Channel or AHB Master Status not detected : 0x0 1h = AXI Master Write Channel or AHB Master Status detected : 0x1

43.7.3.233 DMA_CH0_Control Register (Offset = 1100h) [Reset = 0h]

DMA_CH0_Control is shown in Figure 43-272 and described in Table 43-326.

Return to the Summary Table.

The DMA Channeli Control register specifies the MSS value for segmentation, length to skip between two descriptors, and also the features such as header splitting and 8xPBL mode.

Figure 43-272. DMA_CH0_Control Register

31	30	29	28	27	26	25	24
RESERVED							SPH
R-0h							R/W-0h
23	22	21	20	19	18	17	16
RESERVED			DSL			RESERVED	PBLx8
R-0h			R/W-0h			R-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED		MSS					
R-0h		R/W-0h					
7	6	5	4	3	2	1	0
MSS							
R/W-0h							

Table 43-326. DMA_CH0_Control Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved.
24	SPH	R/W	0h	Split Headers When this bit is set, the DMA splits the header and payload in the Receive path. The DMA writes the header to the Buffer Address1 of RDES0. The DMA writes the payload to the buffer to which the Buffer Address2 is pointing. The software must ensure that the header fits into the Receive buffers. If the header length exceeds the receive buffer size, the DMA does not split the header and payload. This bit is available only if Enable Split Header Structure option is selected. 0h = Split Headers feature is disabled : 0x0 1h = Split Headers feature is enabled : 0x1
23-21	RESERVED	R	0h	Reserved.
20-18	DSL	R/W	0h	Descriptor Skip Length This bit specifies the Word, Dword, or Lword number (depending on the 32-bit, 64-bit, or 128-bit bus) to skip between two unchained descriptors. The address skipping starts from the end of the current descriptor to the start of the next descriptor. When the DSL value is equal to zero, the DMA takes the descriptor table as contiguous.
17	RESERVED	R	0h	Reserved.
16	PBLx8	R/W	0h	8xPBL mode When this bit is set, the PBL value programmed in Bits[21:16] in DMA_CH0_Tx_Control and Bits[21:16] in DMA_CH0_Rx_Control is multiplied by eight times. Therefore, the DMA transfers the data in 8, 16, 32, 64, 128, and 256 beats depending on the PBL value. 0h = 8xPBL mode is disabled : 0x0 1h = 8xPBL mode is enabled : 0x1
15-14	RESERVED	R	0h	Reserved.

Table 43-326. DMA_CH0_Control Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13-0	MSS	R/W	0h	Maximum Segment Size This field specifies the maximum segment size that should be used while segmenting the packet. This field is valid only if the TSE bit of DMA_CH0_Tx_Control register is set. The value programmed in this field must be more than the configured Datawidth in bytes. It is recommended to use a MSS value of 64 bytes or more.

43.7.3.234 DMA_CH0_Tx_Control Register (Offset = 1104h) [Reset = 0h]

DMA_CH0_Tx_Control is shown in Figure 43-273 and described in Table 43-327.

Return to the Summary Table.

The DMA Channeli Transmit Control register controls the Tx features such as PBL, TCP segmentation, and Tx Channel weights.

Figure 43-273. DMA_CH0_Tx_Control Register

31	30	29	28	27	26	25	24
RESERVED			RESERVED	RESERVED			
R-0h			R-0h	R-0h			
23	22	21	20	19	18	17	16
RESERVED	ETIC	TxPBL					
R-0h	R/W-0h	R/W-0h					
15	14	13	12	11	10	9	8
RESERVED	RESERVED		TSE	RESERVED			
R-0h	R-0h		R/W-0h	R-0h			
7	6	5	4	3	2	1	0
RESERVED			OSF	TCW			ST
R-0h			R/W-0h	R/W-0h			R/W-0h

Table 43-327. DMA_CH0_Tx_Control Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved.
28	RESERVED	R	0h	Reserved.
27-24	RESERVED	R	0h	Reserved.
23	RESERVED	R	0h	Reserved.
22	ETIC	R/W	0h	Early Transmit Interrupt Control When this bit is set, Early Transmit Interrupt (ETI) status is set after completion of transfer of data from buffers of a transmit descriptor in which IOC bit (TDES2[31]) is set. When this bit is reset, ETI is set only after a complete packet is transferred to the MTL TX FIFO memory. 0h = Early Transmit Interrupt is disabled : 0x0 1h = Early Transmit Interrupt is enabled : 0x1
21-16	TxPBL	R/W	0h	Transmit Programmable Burst Length These bits indicate the maximum number of beats to be transferred in one DMA block data transfer. The DMA always attempts max burst as specified in PBL each time it starts a burst transfer on the application bus. You can program PBL with any of the following values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. To transfer more than 32 beats, perform the following steps: 1. Set the 8xPBL mode in DMA_CH0_Control register. 2. Set the TxPBL. Note: The maximum value of TxPBL must be less than or equal to half the Tx Queue size (TQS field of MTL_TxQ[n]_Operation_Mode register) in terms of beats. This is required so that the Tx Queue has space to store at least another Tx PBL worth of data while the MTL Tx Queue Controller is transferring data to MAC. For example, in 64-bit data width configurations the total locations in Tx Queue of size 512 bytes is 64, TxPBL and 8xPBL needs to be programmed to less than or equal to 32.
15	RESERVED	R	0h	Reserved.
14-13	RESERVED	R	0h	Reserved.

Table 43-327. DMA_CH0_Tx_Control Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	TSE	R/W	0h	TCP Segmentation Enabled When this bit is set, the DMA performs the TCP segmentation or UDP Segmentation/Fragmentation for packets in this channel. The TCP segmentation or UDP packet's segmentation/Fragmentation is done only for those packets for which the TSE bit (TDES0[19]) is set in the Tx Normal descriptor.When this bit is set, the TxPBL value must be greater than 4. 0h = TCP Segmentation is disabled : 0x0 1h = TCP Segmentation is enabled : 0x1
11-5	RESERVED	R	0h	Reserved.
4	OSF	R/W	0h	Operate on Second Packet When this bit is set, it instructs the DMA to process the second packet of the Transmit data even before the status for the first packet is obtained. 0h = Operate on Second Packet disabled : 0x0 1h = Operate on Second Packet enabled : 0x1
3-1	TCW	R/W	0h	Transmit Channel Weight This field indicates the weight assigned to the corresponding Transmit channel. When reset is complete, this field is set to 0 for all channels by default, resulting in equal weights to all channels.
0	ST	R/W	0h	Start or Stop Transmission Command When this bit is set, transmission is placed in the Running state. The DMA checks the Transmit list at the current position for a packet to be transmitted. The DMA tries to acquire descriptor from either of the following positions: - The current position in the list This is the base address of the Transmit list set by the DMA_CH0_TxDesc_List_Address register. - The position at which the transmission was previously stopped If the DMA does not own the current descriptor, the transmission enters the Suspended state and the TBU bit of the DMA_CH0_Status register is set. The Start Transmission command is effective only when the transmission is stopped. If the command is issued before setting the DMA_CH0_TxDesc_List_Address register, the DMA behavior is unpredictable. When this bit is reset, the transmission process is placed in the Stopped state after completing the transmission of the current packet. The Next Descriptor position in the Transmit list is saved, and it becomes the current position when the transmission is restarted. To change the list address, you need to program DMA_CH0_TxDesc_List_Address register with a new value when this bit is reset. The new value is considered when this bit is set again. The stop transmission command is effective only when the transmission of the current packet is complete or the transmission is in the Suspended state. 0h = Stop Transmission Command : 0x0 1h = Start Transmission Command : 0x1

43.7.3.235 DMA_CH0_Rx_Control Register (Offset = 1108h) [Reset = 0h]

DMA_CH0_Rx_Control is shown in Figure 43-274 and described in Table 43-328.

Return to the Summary Table.

The DMA Channeli Receive Control register controls the Rx features such as PBL, buffer size, and extended status.

Figure 43-274. DMA_CH0_Rx_Control Register

31	30	29	28	27	26	25	24
RPF	RESERVED			RESERVED			
R/W-0h	R-0h			R-0h			
23	22	21	20	19	18	17	16
RESERVED	ERIC	RxPBL					
R-0h	R/W-0h	R/W-0h					
15	14	13	12	11	10	9	8
RESERVED	RBSZ_13_y						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
RBSZ_13_y					RBSZ_x_0		SR
R/W-0h					R-0h		R/W-0h

Table 43-328. DMA_CH0_Rx_Control Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RPF	R/W	0h	Rx Packet Flush. When this bit is set to 1, then DWC_ether_qos automatically flushes the packet from the Rx Queues destined to this DMA Rx Channel, when it is stopped. When this bit remains set and the DMA is re-started by the software driver, the packets residing in the Rx Queues that were received when this RxDMA was stopped, get flushed out. The packets that are received by the MAC after the RxDMA is re-started are routed to the RxDMA. The flushing happens on the Read side of the Rx Queue. When this bit is set to 0, the DWC_ether_qos not flush the packet in the Rx Queue destined to this RxDMA Channel when it is STOP state. This may in turn cause head-of-line blocking in the corresponding RxQueue. 0h = Rx Packet Flush is disabled : 0x0 1h = Rx Packet Flush is enabled : 0x1
30-28	RESERVED	R	0h	Reserved.
27-24	RESERVED	R	0h	Reserved.
23	RESERVED	R	0h	Reserved.
22	ERIC	R/W	0h	Early Receive Interrupt Control When this bit is set, Early Receive Interrupt (ERI) status is set after completion of every burst transfer of data from the Rx DMA to the buffer. When this bit is reset, ERI is set only after a complete buffer is filled up by the RxDMA. 0h = Early Receive Interrupt is disabled : 0x0 1h = Early Receive Interrupt is enabled : 0x1

Table 43-328. DMA_CH0_Rx_Control Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-16	RxPBL	R/W	0h	Receive Programmable Burst Length These bits indicate the maximum number of beats to be transferred in one DMA block data transfer. The DMA always attempts max burst as specified in PBL each time it starts a burst transfer on the application bus. You can program PBL with any of the following values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. To transfer more than 32 beats, perform the following steps: 1. Set the 8xPBL mode in the DMA_CH0_Control register. 2. Set the RxPBL. Note: The maximum value of RxPBL must be less than or equal to half the Rx Queue size (RQS field of MTL_RxQ[n]_Operation_Mode register) in terms of beats. This is required so that the Rx Queue has space to store at least another Rx PBL worth of data while the Rx DMA is transferring a block of data. For example, in 64-bit data width configurations the total locations in Rx Queue of size 512 bytes is 64, so RxPBL and 8xPBL needs to be programmed to less than or equal to 32.
15	RESERVED	R	0h	Reserved.
14-3	RBSZ_13_y	R/W	0h	Receive Buffer size High RBSZ[13:0] is split into two fields higher RBSZ_13_y and lower RBSZ_x_0. The RBSZ[13:0] field indicates the size of the Rx buffers specified in bytes. The maximum buffer size is limited to 16K bytes. The buffer size is applicable to payload buffers when split headers are enabled. Note: The buffer size must be a multiple of 4, 8, or 16 depending on the data bus widths (32-bit, 64-bit, or 128-bit respectively). This is required even if the value of buffer address pointer is not aligned to data bus width. Hence the lower RBSZ_x_0 bits are read-only and the value is considered as all-zero. Thus the RBSZ_13_y indicates the buffer size in terms of locations (with the width same as bus-width).
2-1	RBSZ_x_0	R	0h	Receive Buffer size Low RBSZ[13:0] is split into two fields RBSZ_13_y and RBSZ_x_0. The RBSZ_x_0 is the lower field whose width is based on data bus width of the configuration. This field is of width 2, 3, or 4 bits for 32-bit, 64-bit, or 128-bit data bus width respectively. This field is read-only (RO).
0	SR	R/W	0h	Start or Stop Receive When this bit is set, the DMA tries to acquire the descriptor from the Receive list and processes the incoming packets. The DMA tries to acquire descriptor from either of the following positions: - The current position in the list This is the address set by the DMA_CH0_RxDesc_List_Address register. - The position at which the Rx process was previously stopped If the DMA does not own the current descriptor, the reception is suspended and the RBU bit of the DMA_CH0_Status register is set. The Start Receive command is effective only when the reception is stopped. If the command is issued before setting the DMA_CH0_RxDesc_List_Address register, the DMA behavior is unpredictable. When this bit is reset, the Rx DMA operation is stopped after the transfer of the current packet. The next descriptor position in the Receive list is saved, and it becomes the current position after the Rx process is restarted. The Stop Receive command is effective only when the Rx process is in the Running (waiting for Rx packet) or Suspended state. 0h = Stop Receive : 0x0 1h = Start Receive : 0x1

43.7.3.236 DMA_CH0_TxDesc_List_Address Register (Offset = 1114h) [Reset = 0h]

DMA_CH0_TxDesc_List_Address is shown in Figure 43-275 and described in Table 43-329.

Return to the Summary Table.

The Channeli Tx Descriptor List Address register points the DMA to the start of Transmit descriptor list. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LSB to low.
You can write to this register only when the Tx DMA has stopped, that is, the ST bit is set to zero in DMA_CH0_Tx_Control register. When stopped, this register can be written with a new descriptor list address. When you set the ST bit to 1, the DMA takes the newly-programmed descriptor base address. If this register is not changed when the ST bit is set to 0, the DMA takes the descriptor address where it was stopped earlier.

Figure 43-275. DMA_CH0_TxDesc_List_Address Register

31	30	29	28	27	26	25	24
TDESLA							
R/W-0h							
23	22	21	20	19	18	17	16
TDESLA							
R/W-0h							
15	14	13	12	11	10	9	8
TDESLA							
R/W-0h							
7	6	5	4	3	2	1	0
TDESLA						RESERVED	
R/W-0h						R-0h	

Table 43-329. DMA_CH0_TxDesc_List_Address Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	TDESLA	R/W	0h	Start of Transmit List This field contains the base address of the first descriptor in the Transmit descriptor list. The DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally takes these bits as all-zero. Therefore, these LSB bits are read-only (RO). The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration
1-0	RESERVED	R	0h	

43.7.3.237 DMA_CH0_RxDesc_List_Address Register (Offset = 111Ch) [Reset = 0h]

DMA_CH0_RxDesc_List_Address is shown in Figure 43-276 and described in Table 43-330.

Return to the Summary Table.

The Channeli Rx Descriptor List Address register points the DMA to the start of Receive descriptor list.
This register points to the start of the Receive Descriptor List. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LS bits low. Writing to this register is permitted only when reception is stopped. When stopped, this register must be written to before the receive Start command is given. You can write to this register only when Rx DMA has stopped, that is, SR bit is set to zero in DMA_CH0_Rx_Control register. When stopped, this register can be written with a new descriptor list address.
When you set the SR bit to 1, the DMA takes the newly programmed descriptor base address.

Figure 43-276. DMA_CH0_RxDesc_List_Address Register

31	30	29	28	27	26	25	24
RDESLA							
R/W-0h							
23	22	21	20	19	18	17	16
RDESLA							
R/W-0h							
15	14	13	12	11	10	9	8
RDESLA							
R/W-0h							
7	6	5	4	3	2	1	0
RDESLA						RESERVED	
R/W-0h						R-0h	

Table 43-330. DMA_CH0_RxDesc_List_Address Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RDESLA	R/W	0h	Start of Receive List This field contains the base address of the first descriptor in the Rx Descriptor list. The DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally takes these bits as all-zero. Therefore, these LSB bits are read-only (RO). The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration
1-0	RESERVED	R	0h	

43.7.3.238 DMA_CH0_TxDesc_Tail_Pointer Register (Offset = 1120h) [Reset = 0h]

DMA_CH0_TxDesc_Tail_Pointer is shown in Figure 43-277 and described in Table 43-331.

Return to the Summary Table.

The Channeli Tx Descriptor Tail Pointer register points to an offset from the base and indicates the location of the last valid descriptor.

Figure 43-277. DMA_CH0_TxDesc_Tail_Pointer Register

31	30	29	28	27	26	25	24
TDTP							
R/W-0h							
23	22	21	20	19	18	17	16
TDTP							
R/W-0h							
15	14	13	12	11	10	9	8
TDTP							
R/W-0h							
7	6	5	4	3	2	1	0
TDTP						RESERVED	
R/W-0h						R-0h	

Table 43-331. DMA_CH0_TxDesc_Tail_Pointer Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	TDTP	R/W	0h	Transmit Descriptor Tail Pointer This field contains the tail pointer for the Tx descriptor ring. The software writes the tail pointer to add more descriptors to the Tx channel. The hardware tries to transmit all packets referenced by the descriptors between the head and the tail pointer registers. The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration
1-0	RESERVED	R	0h	

43.7.3.239 DMA_CH0_RxDesc_Tail_Pointer Register (Offset = 1128h) [Reset = 0h]

DMA_CH0_RxDesc_Tail_Pointer is shown in Figure 43-278 and described in Table 43-332.

Return to the Summary Table.

The Channeli Rx Descriptor Tail Pointer Points to an offset from the base and indicates the location of the last valid descriptor.

Figure 43-278. DMA_CH0_RxDesc_Tail_Pointer Register

31	30	29	28	27	26	25	24
RDTP							
R/W-0h							
23	22	21	20	19	18	17	16
RDTP							
R/W-0h							
15	14	13	12	11	10	9	8
RDTP							
R/W-0h							
7	6	5	4	3	2	1	0
RDTP						RESERVED	
R/W-0h						R-0h	

Table 43-332. DMA_CH0_RxDesc_Tail_Pointer Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RDTP	R/W	0h	Receive Descriptor Tail Pointer This field contains the tail pointer for the Rx descriptor ring. The software writes the tail pointer to add more descriptors to the Rx channel. The hardware tries to write all received packets to the descriptors referenced between the head and the tail pointer registers. The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration
1-0	RESERVED	R	0h	

43.7.3.240 DMA_CH0_TxDesc_Ring_Length Register (Offset = 112Ch) [Reset = 0h]

DMA_CH0_TxDesc_Ring_Length is shown in Figure 43-279 and described in Table 43-333.

Return to the Summary Table.

The Tx Descriptor Ring Length register contains the length of the Transmit descriptor ring.

Figure 43-279. DMA_CH0_TxDesc_Ring_Length Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						TDRL									
R-0h																						R/W-0h									

Table 43-333. DMA_CH0_TxDesc_Ring_Length Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved.
9-0	TDRL	R/W	0h	Transmit Descriptor Ring Length This field sets the maximum number of Tx descriptors in the circular descriptor ring. The maximum number of descriptors is limited to 1K descriptors. Synopsys recommends a minimum ring descriptor length of 4. For example, You can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. If you want to have 10 descriptors, program it to a value of 0x9.

43.7.3.241 DMA_CH0_RxDesc_Ring_Length Register (Offset = 1130h) [Reset = 0h]

DMA_CH0_RxDesc_Ring_Length is shown in Figure 43-280 and described in Table 43-334.

Return to the Summary Table.

The Channeli Rx Descriptor Ring Length register contains the length of the Receive descriptor circular ring.

Figure 43-280. DMA_CH0_RxDesc_Ring_Length Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						RDRL									
R-0h																						R/W-0h									

Table 43-334. DMA_CH0_RxDesc_Ring_Length Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved.
9-0	RDRL	R/W	0h	Receive Descriptor Ring Length This register sets the maximum number of Rx descriptors in the circular descriptor ring. The maximum number of descriptors is limited to 1K descriptors. For example, You can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. If you want to have 10 descriptors, program it to a value of 0x9.

43.7.3.242 DMA_CH0_Interrupt_Enable Register (Offset = 1134h) [Reset = 0h]

DMA_CH0_Interrupt_Enable is shown in Figure 43-281 and described in Table 43-335.

Return to the Summary Table.

The Channeli Interrupt Enable register enables the interrupts reported by the Status register.

Figure 43-281. DMA_CH0_Interrupt_Enable Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
NIE	AIE	CDEE	FBEE	ERIE	ETIE	RWTE	RSE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RBUE	RIE	RESERVED			TBUE	TXSE	TIE
R/W-0h	R/W-0h	R-0h			R/W-0h	R/W-0h	R/W-0h

Table 43-335. DMA_CH0_Interrupt_Enable Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved.
15	NIE	R/W	0h	Normal Interrupt Summary Enable When this bit is set, the normal interrupt summary is enabled. This bit enables the following interrupts in the DMA_CH0_Status register: - Bit 0: Transmit Interrupt - Bit 2: Transmit Buffer Unavailable - Bit 6: Receive Interrupt - Bit 11: Early Receive Interrupt When this bit is reset, the normal interrupt summary is disabled. 0h = Normal Interrupt Summary is disabled : 0x0 1h = Normal Interrupt Summary is enabled : 0x1
14	AIE	R/W	0h	Abnormal Interrupt Summary Enable When this bit is set, the abnormal interrupt summary is enabled. This bit enables the following interrupts in the DMA_CH0_Status register: - Bit 1: Transmit Process Stopped - Bit 7: Rx Buffer Unavailable - Bit 8: Receive Process Stopped - Bit 9: Receive Watchdog Timeout - Bit 10: Early Transmit Interrupt - Bit 12: Fatal Bus Error - Bit 13: Context Descriptor Error When this bit is reset, the abnormal interrupt summary is disabled. 0h = Abnormal Interrupt Summary is disabled : 0x0 1h = Abnormal Interrupt Summary is enabled : 0x1
13	CDEE	R/W	0h	Context Descriptor Error Enable When this bit is set along with the AIE bit, the Descriptor error interrupt is enabled. When this bit is reset, the Descriptor error interrupt is disabled. 0h = Context Descriptor Error is disabled : 0x0 1h = Context Descriptor Error is enabled : 0x1

Table 43-335. DMA_CH0_Interrupt_Enable Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	FBEE	R/W	0h	Fatal Bus Error Enable When this bit is set along with the AIE bit, the Fatal Bus error interrupt is enabled. When this bit is reset, the Fatal Bus Error error interrupt is disabled. 0h = Fatal Bus Error is disabled : 0x0 1h = Fatal Bus Error is enabled : 0x1
11	ERIE	R/W	0h	Early Receive Interrupt Enable When this bit is set along with the NIE bit, the Early Receive interrupt is enabled. When this bit is reset, the Early Receive interrupt is disabled. 0h = Early Receive Interrupt is disabled : 0x0 1h = Early Receive Interrupt is enabled : 0x1
10	ETIE	R/W	0h	Early Transmit Interrupt Enable When this bit is set along with the AIE bit, the Early Transmit interrupt is enabled. When this bit is reset, the Early Transmit interrupt is disabled. 0h = Early Transmit Interrupt is disabled : 0x0 1h = Early Transmit Interrupt is enabled : 0x1
9	RWTE	R/W	0h	Receive Watchdog Timeout Enable When this bit is set along with the AIE bit, the Receive Watchdog Timeout interrupt is enabled. When this bit is reset, the Receive Watchdog Timeout interrupt is disabled. 0h = Receive Watchdog Timeout is disabled : 0x0 1h = Receive Watchdog Timeout is enabled : 0x1
8	RSE	R/W	0h	Receive Stopped Enable When this bit is set along with the AIE bit, the Receive Stopped Interrupt is enabled. When this bit is reset, the Receive Stopped interrupt is disabled. 0h = Receive Stopped is disabled : 0x0 1h = Receive Stopped is enabled : 0x1
7	RBUE	R/W	0h	Receive Buffer Unavailable Enable When this bit is set along with the AIE bit, the Receive Buffer Unavailable interrupt is enabled. When this bit is reset, the Receive Buffer Unavailable interrupt is disabled. 0h = Receive Buffer Unavailable is disabled : 0x0 1h = Receive Buffer Unavailable is enabled : 0x1
6	RIE	R/W	0h	Receive Interrupt Enable When this bit is set along with the NIE bit, the Receive Interrupt is enabled. When this bit is reset, the Receive Interrupt is disabled. 0h = Receive Interrupt is disabled : 0x0 1h = Receive Interrupt is enabled : 0x1
5-3	RESERVED	R	0h	Reserved.
2	TBUE	R/W	0h	Transmit Buffer Unavailable Enable When this bit is set along with the NIE bit, the Transmit Buffer Unavailable interrupt is enabled. When this bit is reset, the Transmit Buffer Unavailable interrupt is disabled. 0h = Transmit Buffer Unavailable is disabled : 0x0 1h = Transmit Buffer Unavailable is enabled : 0x1
1	TXSE	R/W	0h	Transmit Stopped Enable When this bit is set along with the AIE bit, the Transmission Stopped interrupt is enabled. When this bit is reset, the Transmission Stopped interrupt is disabled. 0h = Transmit Stopped is disabled : 0x0 1h = Transmit Stopped is enabled : 0x1
0	TIE	R/W	0h	Transmit Interrupt Enable When this bit is set along with the NIE bit, the Transmit Interrupt is enabled. When this bit is reset, the Transmit Interrupt is disabled. 0h = Transmit Interrupt is disabled : 0x0 1h = Transmit Interrupt is enabled : 0x1

43.7.3.243 DMA_CH0_Rx_Interrupt_Watchdog_Timer Register (Offset = 1138h) [Reset = 0h]

DMA_CH0_Rx_Interrupt_Watchdog_Timer is shown in Figure 43-282 and described in Table 43-336.

Return to the Summary Table.

The Receive Interrupt Watchdog Timer register indicates the watchdog timeout for Receive Interrupt (RI) from the DMA. When this register is written with a non-zero value, it enables the watchdog timer for the RI bit of the DMA_CHi_Status register.

Figure 43-282. DMA_CH0_Rx_Interrupt_Watchdog_Timer Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED														RWTU	
R-0h														R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RWT							
R-0h								R/W-0h							

Table 43-336. DMA_CH0_Rx_Interrupt_Watchdog_Timer Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved.
17-16	RWTU	R/W	0h	Receive Interrupt Watchdog Timer Count Units This fields indicates the number of system clock cycles corresponding to one unit in RWT field. - 2'b00: 256 - 2'b01: 512 - 2'b10: 1024 - 2'b11: 2048 For example, when RWT=2 and RWTU=1, the watchdog timer is set for 2*512=1024 system clock cycles.
15-8	RESERVED	R	0h	Reserved.
7-0	RWT	R/W	0h	Receive Interrupt Watchdog Timer Count This field indicates the number of system clock cycles, multiplied by factor indicated in RWTU field, for which the watchdog timer is set. The watchdog timer is triggered with the programmed value after the Rx DMA completes the transfer of a packet for which the RI bit is not set in the DMA_CH0_Status register, because of the setting of Interrupt Enable bit in the corresponding descriptor RDES3[30]. When the watchdog timer runs out, the RI bit is set and the timer is stopped. The watchdog timer is reset when the RI bit is set high because of automatic setting of RI as per the Interrupt Enable bit RDES3[30] of any received packet.

43.7.3.244 DMA_CH0_Current_App_TxDesc Register (Offset = 1144h) [Reset = 0h]

DMA_CH0_Current_App_TxDesc is shown in Figure 43-283 and described in Table 43-337.

Return to the Summary Table.

The Channeli Current Application Transmit Descriptor register points to the current Transmit descriptor read by the DMA.

Figure 43-283. DMA_CH0_Current_App_TxDesc Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CURTDESAPTR																															
R-0h																															

Table 43-337. DMA_CH0_Current_App_TxDesc Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CURTDESAPTR	R	0h	Application Transmit Descriptor Address Pointer The DMA updates this pointer during Tx operation. This pointer is cleared on reset.

43.7.3.245 DMA_CH0_Current_App_RxDesc Register (Offset = 114Ch) [Reset = 0h]

DMA_CH0_Current_App_RxDesc is shown in Figure 43-284 and described in Table 43-338.

Return to the Summary Table.

The Channeli Current Application Receive Descriptor register points to the current Receive descriptor read by the DMA.

Figure 43-284. DMA_CH0_Current_App_RxDesc Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CURRDESAPTR																															
R-0h																															

Table 43-338. DMA_CH0_Current_App_RxDesc Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CURRDESAPTR	R	0h	Application Receive Descriptor Address Pointer The DMA updates this pointer during Rx operation. This pointer is cleared on reset.

43.7.3.246 DMA_CH0_Current_App_TxBuffer Register (Offset = 1154h) [Reset = 0h]

DMA_CH0_Current_App_TxBuffer is shown in Figure 43-285 and described in Table 43-339.

Return to the Summary Table.

The Channeli Current Application Transmit Buffer Address register points to the current Tx buffer address read by the DMA.

Figure 43-285. DMA_CH0_Current_App_TxBuffer Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CURTBUFAPTR																															
R-0h																															

Table 43-339. DMA_CH0_Current_App_TxBuffer Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CURTBUFAPTR	R	0h	Application Transmit Buffer Address Pointer The DMA updates this pointer during Tx operation. This pointer is cleared on reset.

43.7.3.247 DMA_CH0_Current_App_RxBuffer Register (Offset = 115Ch) [Reset = 0h]

DMA_CH0_Current_App_RxBuffer is shown in Figure 43-286 and described in Table 43-340.

Return to the Summary Table.

The Channel 0 Current Application Receive Buffer Address register points to the current Rx buffer address read by the DMA.

Figure 43-286. DMA_CH0_Current_App_RxBuffer Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CURRBUFAPTR																															
R-0h																															

Table 43-340. DMA_CH0_Current_App_RxBuffer Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CURRBUFAPTR	R	0h	Application Receive Buffer Address Pointer The DMA updates this pointer during Rx operation. This pointer is cleared on reset.

43.7.3.248 DMA_CH0_Status Register (Offset = 1160h) [Reset = 0h]

DMA_CH0_Status is shown in Figure 43-287 and described in Table 43-341.

Return to the Summary Table.

The software driver (application) reads the Status register during interrupt service routine or polling to determine the status of the DMA.
Note: The number of DMA_CH[n]_Status register in the configuration is the higher of number of Rx DMA Channels and Tx DMA Channels.

Figure 43-287. DMA_CH0_Status Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED										REB			TEB		
R-0h										R-0h			R-0h		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NIS	AIS	CDE	FBE	ERI	ETI	RWT	RPS	RBU	RI	RESERVED			TBU	TPS	TI
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h			R/W-0h	R/W-0h	R/W-0h

Table 43-341. DMA_CH0_Status Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reserved.
21-19	REB	R	0h	Rx DMA Error Bits This field indicates the type of error that caused a Bus Error. For example, error response on the AHB or AXI interface. Bit 21 - 1'b1: Error during data transfer by Rx DMA - 1'b0: No Error during data transfer by Rx DMA Bit 20 - 1'b1: Error during descriptor access - 1'b0: Error during data buffer access Bit 19 - 1'b1: Error during read transfer - 1'b0: Error during write transfer This field is valid only when the FBE bit is set. This field does not generate an interrupt.
18-16	TEB	R	0h	Tx DMA Error Bits This field indicates the type of error that caused a Bus Error. For example, error response on the AHB or AXI interface. Bit 18 - 1'b1: Error during data transfer by Tx DMA - 1'b0: No Error during data transfer by Tx DMA Bit 17 - 1'b1: Error during descriptor access - 1'b0: Error during data buffer access Bit 16 - 1'b1: Error during read transfer - 1'b0: Error during write transfer This field is valid only when the FBE bit is set. This field does not generate an interrupt.

Table 43-341. DMA_CH0_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15	NIS	R/W	0h	Normal Interrupt Summary Normal Interrupt Summary bit value is the logical OR of the following bits when the corresponding interrupt bits are enabled in the DMA_CH0_Interrupt_Enable register: - Bit 0: Transmit Interrupt - Bit 2: Transmit Buffer Unavailable - Bit 6: Receive Interrupt - Bit 11: Early Receive Interrupt Only unmasked bits (interrupts for which interrupt enable is set in DMA_CH0_Interrupt_Enable register) affect the Normal Interrupt Summary bit. This is a sticky bit. You must clear this bit (by writing 1 to this bit) each time a corresponding bit which causes NIS to be set is cleared. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Normal Interrupt Summary status not detected : 0x0 1h = Normal Interrupt Summary status detected : 0x1
14	AIS	R/W	0h	Abnormal Interrupt Summary Abnormal Interrupt Summary bit value is the logical OR of the following when the corresponding interrupt bits are enabled in the DMA_CH0_Interrupt_Enable register: - Bit 1: Transmit Process Stopped - Bit 7: Receive Buffer Unavailable - Bit 8: Receive Process Stopped - Bit 10: Early Transmit Interrupt - Bit 12: Fatal Bus Error - Bit 13: Context Descriptor Error Only unmasked bits affect the Abnormal Interrupt Summary bit. This is a sticky bit. You must clear this bit (by writing 1 to this bit) each time a corresponding bit, which causes AIS to be set, is cleared. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Abnormal Interrupt Summary status not detected : 0x0 1h = Abnormal Interrupt Summary status detected : 0x1
13	CDE	R/W	0h	Context Descriptor Error This bit indicates that the DMA Tx/Rx engine received a descriptor error, which indicates invalid context in the middle of packet flow (intermediate descriptor) or all one's descriptor in Tx case and on Rx side it indicates DMA has read a descriptor with either of the buffer address as ones which is considered to be invalid. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Context Descriptor Error status not detected : 0x0 1h = Context Descriptor Error status detected : 0x1
12	FBE	R/W	0h	Fatal Bus Error This bit indicates that a bus error occurred (as described in the EB field). When this bit is set, the corresponding DMA channel engine disables all bus accesses. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Fatal Bus Error status not detected : 0x0 1h = Fatal Bus Error status detected : 0x1
11	ERI	R/W	0h	Early Receive Interrupt This bit when set indicates that the RxDMA has completed the transfer of packet data to the memory. When ERIC=0, this bit is set only after the Rx DMA has filled up a complete receive buffer with packet data. When ERIC=1, this bit is set after every burst transfer of data from the Rx DMA to the buffer. The setting of RI bit automatically clears this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Early Receive Interrupt status not detected : 0x0 1h = Early Receive Interrupt status detected : 0x1

Table 43-341. DMA_CH0_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	ETI	R/W	0h	Early Transmit Interrupt This bit when set indicates that the TxDMA has completed the transfer of packet data to the MTL TXFIFO memory. When ETIC=0, this bit is set only after the Tx DMA has transferred a complete packet to MTL. When ETIC=1, this bit is set after completion of (partial) packet data transfer from buffers in the Transmit descriptor in which IOC=1. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Early Transmit Interrupt status not detected : 0x0 1h = Early Transmit Interrupt status detected : 0x1
9	RWT	R/W	0h	Receive Watchdog Timeout This bit is asserted when a packet with length greater than 2,048 bytes (10,240 bytes when Jumbo Packet mode is enabled) is received. 0h = Receive Watchdog Timeout status not detected : 0x0 1h = Receive Watchdog Timeout status detected : 0x1
8	RPS	R/W	0h	Receive Process Stopped This bit is asserted when the Rx process enters the Stopped state. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Receive Process Stopped status not detected : 0x0 1h = Receive Process Stopped status detected : 0x1
7	RBU	R/W	0h	Receive Buffer Unavailable This bit indicates that the application owns the next descriptor in the Receive list, and the DMA cannot acquire it. The Rx process is suspended. To resume processing Rx descriptors, the application should change the ownership of the descriptor and issue a Receive Poll Demand command. If this command is not issued, the Rx process resumes when the next recognized incoming packet is received. In ring mode, the application should advance the Receive Descriptor Tail Pointer register of a channel. This bit is set only when the DMA owns the previous Rx descriptor. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Receive Buffer Unavailable status not detected : 0x0 1h = Receive Buffer Unavailable status detected : 0x1
6	RI	R/W	0h	Receive Interrupt This bit indicates that the packet reception is complete. When packet reception is complete, Bit 31 of RDES3 is reset in the last descriptor, and the specific packet status information is updated in the descriptor. The reception remains in the Running state. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Receive Interrupt status not detected : 0x0 1h = Receive Interrupt status detected : 0x1
5-3	RESERVED	R	0h	Reserved.

Table 43-341. DMA_CH0_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	TBU	R/W	0h	Transmit Buffer Unavailable This bit indicates that the application owns the next descriptor in the Transmit list, and the DMA cannot acquire it. Transmission is suspended. The TPS0 field of the DMA_Debug_Status0 register explains the Transmit Process state transitions. To resume processing the Transmit descriptors, the application should do the following: 1. Change the ownership of the descriptor by setting Bit 31 of TDES3. 2. Issue a Transmit Poll Demand command. For ring mode, the application should advance the Transmit Descriptor Tail Pointer register of a channel. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Transmit Buffer Unavailable status not detected : 0x0 1h = Transmit Buffer Unavailable status detected : 0x1
1	TPS	R/W	0h	Transmit Process Stopped This bit is set when the transmission is stopped. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Transmit Process Stopped status not detected : 0x0 1h = Transmit Process Stopped status detected : 0x1
0	TI	R/W	0h	Transmit Interrupt This bit indicates that the packet transmission is complete. When transmission is complete, Bit 31 of TDES3 is reset in the last descriptor, and the specific packet status information is updated in the descriptor. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Transmit Interrupt status not detected : 0x0 1h = Transmit Interrupt status detected : 0x1

43.7.3.249 DMA_CH0_Miss_Frame_Cnt Register (Offset = 1164h) [Reset = 0h]

DMA_CH0_Miss_Frame_Cnt is shown in Figure 43-288 and described in Table 43-342.

Return to the Summary Table.

This register has the number of packet counter that got dropped by the DMA either due to Bus Error or due to programming RPF field in DMA_CH${i}_Rx_Control register.

Figure 43-288. DMA_CH0_Miss_Frame_Cnt Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
MFCO	RESERVED				MFC		
R-0h	R-0h				R-0h		
7	6	5	4	3	2	1	0
MFC							
R-0h							

Table 43-342. DMA_CH0_Miss_Frame_Cnt Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved.
15	MFCO	R	0h	Overflow status of the MFC Counter When this bit is set then the MFC counter does not get incremented further. The bit gets cleared when this register is read. Access restriction applies. Clears on read. Self-set to 1 on internal event.
14-11	RESERVED	R	0h	Reserved.
10-0	MFC	R	0h	Dropped Packet Counters This counter indicates the number of packet counters that are dropped by the DMA either because of bus error or because of programing RPF field in DMA_CH${i}_Rx_Control register. The counter gets cleared when this register is read. Access restriction applies. Clears on read. Self-set to 1 on internal event.

43.7.3.250 DMA_CH0_RX_ERI_Cnt Register (Offset = 116Ch) [Reset = 0h]

DMA_CH0_RX_ERI_Cnt is shown in Figure 43-289 and described in Table 43-343.

Return to the Summary Table.

Figure 43-289. DMA_CH0_RX_ERI_Cnt Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				ECNT											
R-0h																				R-0h											

Table 43-343. DMA_CH0_RX_ERI_Cnt Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	
11-0	ECNT	R	0h	ERI Counter When ERIC bit of DMA_CH(#i)_RX_Control register is set, this counter increments for burst transfer completed by the Rx DMA from the start of packet transfer. This counter will get reset at the start of new packet.

43.7.3.251 DMA_CH1_Control Register (Offset = 1180h) [Reset = 0h]

DMA_CH1_Control is shown in Figure 43-290 and described in Table 43-344.

Return to the Summary Table.

The DMA Channeli Control register specifies the MSS value for segmentation, length to skip between two descriptors, and also the features such as header splitting and 8xPBL mode.

Figure 43-290. DMA_CH1_Control Register

31	30	29	28	27	26	25	24
RESERVED							SPH
R-0h							R/W-0h
23	22	21	20	19	18	17	16
RESERVED			DSL			RESERVED	PBLx8
R-0h			R/W-0h			R-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED		MSS					
R-0h		R/W-0h					
7	6	5	4	3	2	1	0
MSS							
R/W-0h							

Table 43-344. DMA_CH1_Control Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved.
24	SPH	R/W	0h	Split Headers When this bit is set, the DMA splits the header and payload in the Receive path. The DMA writes the header to the Buffer Address1 of RDES0. The DMA writes the payload to the buffer to which the Buffer Address2 is pointing. The software must ensure that the header fits into the Receive buffers. If the header length exceeds the receive buffer size, the DMA does not split the header and payload. This bit is available only if Enable Split Header Structure option is selected. 0h = Split Headers feature is disabled : 0x0 1h = Split Headers feature is enabled : 0x1
23-21	RESERVED	R	0h	Reserved.
20-18	DSL	R/W	0h	Descriptor Skip Length This bit specifies the Word, Dword, or Lword number (depending on the 32-bit, 64-bit, or 128-bit bus) to skip between two unchained descriptors. The address skipping starts from the end of the current descriptor to the start of the next descriptor. When the DSL value is equal to zero, the DMA takes the descriptor table as contiguous.
17	RESERVED	R	0h	Reserved.
16	PBLx8	R/W	0h	8xPBL mode When this bit is set, the PBL value programmed in Bits[21:16] in DMA_CH0_Tx_Control and Bits[21:16] in DMA_CH0_Rx_Control is multiplied by eight times. Therefore, the DMA transfers the data in 8, 16, 32, 64, 128, and 256 beats depending on the PBL value. 0h = 8xPBL mode is disabled : 0x0 1h = 8xPBL mode is enabled : 0x1
15-14	RESERVED	R	0h	Reserved.

Table 43-344. DMA_CH1_Control Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13-0	MSS	R/W	0h	Maximum Segment Size This field specifies the maximum segment size that should be used while segmenting the packet. This field is valid only if the TSE bit of DMA_CH0_Tx_Control register is set. The value programmed in this field must be more than the configured Datawidth in bytes. It is recommended to use a MSS value of 64 bytes or more.

43.7.3.252 DMA_CH1_Tx_Control Register (Offset = 1184h) [Reset = 0h]

DMA_CH1_Tx_Control is shown in Figure 43-291 and described in Table 43-345.

Return to the Summary Table.

The DMA Channeli Transmit Control register controls the Tx features such as PBL, TCP segmentation, and Tx Channel weights.

Figure 43-291. DMA_CH1_Tx_Control Register

31	30	29	28	27	26	25	24
RESERVED			RESERVED	RESERVED			
R-0h			R-0h	R-0h			
23	22	21	20	19	18	17	16
RESERVED	ETIC	TxPBL					
R-0h	R/W-0h	R/W-0h					
15	14	13	12	11	10	9	8
RESERVED	RESERVED		TSE	RESERVED			
R-0h	R-0h		R/W-0h	R-0h			
7	6	5	4	3	2	1	0
RESERVED			OSF	TCW			ST
R-0h			R/W-0h	R/W-0h			R/W-0h

Table 43-345. DMA_CH1_Tx_Control Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved.
28	RESERVED	R	0h	Reserved.
27-24	RESERVED	R	0h	Reserved.
23	RESERVED	R	0h	Reserved.
22	ETIC	R/W	0h	Early Transmit Interrupt Control When this bit is set, Early Transmit Interrupt (ETI) status is set after completion of transfer of data from buffers of a transmit descriptor in which IOC bit (TDES2[31]) is set. When this bit is reset, ETI is set only after a complete packet is transferred to the MTL TX FIFO memory. 0h = Early Transmit Interrupt is disabled : 0x0 1h = Early Transmit Interrupt is enabled : 0x1
21-16	TxPBL	R/W	0h	Transmit Programmable Burst Length These bits indicate the maximum number of beats to be transferred in one DMA block data transfer. The DMA always attempts max burst as specified in PBL each time it starts a burst transfer on the application bus. You can program PBL with any of the following values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. To transfer more than 32 beats, perform the following steps: 1. Set the 8xPBL mode in DMA_CH0_Control register. 2. Set the TxPBL. Note: The maximum value of TxPBL must be less than or equal to half the Tx Queue size (TQS field of MTL_TxQ[n]_Operation_Mode register) in terms of beats. This is required so that the Tx Queue has space to store at least another Tx PBL worth of data while the MTL Tx Queue Controller is transferring data to MAC. For example, in 64-bit data width configurations the total locations in Tx Queue of size 512 bytes is 64, TxPBL and 8xPBL needs to be programmed to less than or equal to 32.
15	RESERVED	R	0h	Reserved.
14-13	RESERVED	R	0h	Reserved.

Table 43-345. DMA_CH1_Tx_Control Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	TSE	R/W	0h	TCP Segmentation Enabled When this bit is set, the DMA performs the TCP segmentation or UDP Segmentation/Fragmentation for packets in this channel. The TCP segmentation or UDP packet's segmentation/Fragmentation is done only for those packets for which the TSE bit (TDES0[19]) is set in the Tx Normal descriptor.When this bit is set, the TxPBL value must be greater than 4. 0h = TCP Segmentation is disabled : 0x0 1h = TCP Segmentation is enabled : 0x1
11-5	RESERVED	R	0h	Reserved.
4	OSF	R/W	0h	Operate on Second Packet When this bit is set, it instructs the DMA to process the second packet of the Transmit data even before the status for the first packet is obtained. 0h = Operate on Second Packet disabled : 0x0 1h = Operate on Second Packet enabled : 0x1
3-1	TCW	R/W	0h	Transmit Channel Weight This field indicates the weight assigned to the corresponding Transmit channel. When reset is complete, this field is set to 0 for all channels by default, resulting in equal weights to all channels.
0	ST	R/W	0h	Start or Stop Transmission Command When this bit is set, transmission is placed in the Running state. The DMA checks the Transmit list at the current position for a packet to be transmitted. The DMA tries to acquire descriptor from either of the following positions: - The current position in the list This is the base address of the Transmit list set by the DMA_CH0_TxDesc_List_Address register. - The position at which the transmission was previously stopped If the DMA does not own the current descriptor, the transmission enters the Suspended state and the TBU bit of the DMA_CH0_Status register is set. The Start Transmission command is effective only when the transmission is stopped. If the command is issued before setting the DMA_CH0_TxDesc_List_Address register, the DMA behavior is unpredictable. When this bit is reset, the transmission process is placed in the Stopped state after completing the transmission of the current packet. The Next Descriptor position in the Transmit list is saved, and it becomes the current position when the transmission is restarted. To change the list address, you need to program DMA_CH0_TxDesc_List_Address register with a new value when this bit is reset. The new value is considered when this bit is set again. The stop transmission command is effective only when the transmission of the current packet is complete or the transmission is in the Suspended state. 0h = Stop Transmission Command : 0x0 1h = Start Transmission Command : 0x1

43.7.3.253 DMA_CH1_Rx_Control Register (Offset = 1188h) [Reset = 0h]

DMA_CH1_Rx_Control is shown in Figure 43-292 and described in Table 43-346.

Return to the Summary Table.

The DMA Channeli Receive Control register controls the Rx features such as PBL, buffer size, and extended status.

Figure 43-292. DMA_CH1_Rx_Control Register

31	30	29	28	27	26	25	24
RPF	RESERVED			RESERVED			
R/W-0h	R-0h			R-0h			
23	22	21	20	19	18	17	16
RESERVED	ERIC	RxPBL					
R-0h	R/W-0h	R/W-0h					
15	14	13	12	11	10	9	8
RESERVED	RBSZ_13_y						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
RBSZ_13_y					RBSZ_x_0		SR
R/W-0h					R-0h		R/W-0h

Table 43-346. DMA_CH1_Rx_Control Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RPF	R/W	0h	Rx Packet Flush. When this bit is set to 1, then DWC_ether_qos automatically flushes the packet from the Rx Queues destined to this DMA Rx Channel, when it is stopped. When this bit remains set and the DMA is re-started by the software driver, the packets residing in the Rx Queues that were received when this RxDMA was stopped, get flushed out. The packets that are received by the MAC after the RxDMA is re-started are routed to the RxDMA. The flushing happens on the Read side of the Rx Queue. When this bit is set to 0, the DWC_ether_qos not flush the packet in the Rx Queue destined to this RxDMA Channel when it is STOP state. This may in turn cause head-of-line blocking in the corresponding RxQueue. 0h = Rx Packet Flush is disabled : 0x0 1h = Rx Packet Flush is enabled : 0x1
30-28	RESERVED	R	0h	Reserved.
27-24	RESERVED	R	0h	Reserved.
23	RESERVED	R	0h	Reserved.
22	ERIC	R/W	0h	Early Receive Interrupt Control When this bit is set, Early Receive Interrupt (ERI) status is set after completion of every burst transfer of data from the Rx DMA to the buffer. When this bit is reset, ERI is set only after a complete buffer is filled up by the RxDMA. 0h = Early Receive Interrupt is disabled : 0x0 1h = Early Receive Interrupt is enabled : 0x1

Table 43-346. DMA_CH1_Rx_Control Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-16	RxPBL	R/W	0h	Receive Programmable Burst Length These bits indicate the maximum number of beats to be transferred in one DMA block data transfer. The DMA always attempts max burst as specified in PBL each time it starts a burst transfer on the application bus. You can program PBL with any of the following values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. To transfer more than 32 beats, perform the following steps: 1. Set the 8xPBL mode in the DMA_CH0_Control register. 2. Set the RxPBL. Note: The maximum value of RxPBL must be less than or equal to half the Rx Queue size (RQS field of MTL_RxQ[n]_Operation_Mode register) in terms of beats. This is required so that the Rx Queue has space to store at least another Rx PBL worth of data while the Rx DMA is transferring a block of data. For example, in 64-bit data width configurations the total locations in Rx Queue of size 512 bytes is 64, so RxPBL and 8xPBL needs to be programmed to less than or equal to 32.
15	RESERVED	R	0h	Reserved.
14-3	RBSZ_13_y	R/W	0h	Receive Buffer size High RBSZ[13:0] is split into two fields higher RBSZ_13_y and lower RBSZ_x_0. The RBSZ[13:0] field indicates the size of the Rx buffers specified in bytes. The maximum buffer size is limited to 16K bytes. The buffer size is applicable to payload buffers when split headers are enabled. Note: The buffer size must be a multiple of 4, 8, or 16 depending on the data bus widths (32-bit, 64-bit, or 128-bit respectively). This is required even if the value of buffer address pointer is not aligned to data bus width. Hence the lower RBSZ_x_0 bits are read-only and the value is considered as all-zero. Thus the RBSZ_13_y indicates the buffer size in terms of locations (with the width same as bus-width).
2-1	RBSZ_x_0	R	0h	Receive Buffer size Low RBSZ[13:0] is split into two fields RBSZ_13_y and RBSZ_x_0. The RBSZ_x_0 is the lower field whose width is based on data bus width of the configuration. This field is of width 2, 3, or 4 bits for 32-bit, 64-bit, or 128-bit data bus width respectively. This field is read-only (RO).
0	SR	R/W	0h	Start or Stop Receive When this bit is set, the DMA tries to acquire the descriptor from the Receive list and processes the incoming packets. The DMA tries to acquire descriptor from either of the following positions: - The current position in the list This is the address set by the DMA_CH0_RxDesc_List_Address register. - The position at which the Rx process was previously stopped If the DMA does not own the current descriptor, the reception is suspended and the RBU bit of the DMA_CH0_Status register is set. The Start Receive command is effective only when the reception is stopped. If the command is issued before setting the DMA_CH0_RxDesc_List_Address register, the DMA behavior is unpredictable. When this bit is reset, the Rx DMA operation is stopped after the transfer of the current packet. The next descriptor position in the Receive list is saved, and it becomes the current position after the Rx process is restarted. The Stop Receive command is effective only when the Rx process is in the Running (waiting for Rx packet) or Suspended state. 0h = Stop Receive : 0x0 1h = Start Receive : 0x1

43.7.3.254 DMA_CH1_TxDesc_List_Address Register (Offset = 1194h) [Reset = 0h]

DMA_CH1_TxDesc_List_Address is shown in Figure 43-293 and described in Table 43-347.

Return to the Summary Table.

The Channeli Tx Descriptor List Address register points the DMA to the start of Transmit descriptor list. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LSB to low.
You can write to this register only when the Tx DMA has stopped, that is, the ST bit is set to zero in DMA_CH0_Tx_Control register. When stopped, this register can be written with a new descriptor list address. When you set the ST bit to 1, the DMA takes the newly-programmed descriptor base address. If this register is not changed when the ST bit is set to 0, the DMA takes the descriptor address where it was stopped earlier.

Figure 43-293. DMA_CH1_TxDesc_List_Address Register

31	30	29	28	27	26	25	24
TDESLA							
R/W-0h							
23	22	21	20	19	18	17	16
TDESLA							
R/W-0h							
15	14	13	12	11	10	9	8
TDESLA							
R/W-0h							
7	6	5	4	3	2	1	0
TDESLA						RESERVED	
R/W-0h						R-0h	

Table 43-347. DMA_CH1_TxDesc_List_Address Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	TDESLA	R/W	0h	Start of Transmit List This field contains the base address of the first descriptor in the Transmit descriptor list. The DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally takes these bits as all-zero. Therefore, these LSB bits are read-only (RO). The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration
1-0	RESERVED	R	0h	

43.7.3.255 DMA_CH1_RxDesc_List_Address Register (Offset = 119Ch) [Reset = 0h]

DMA_CH1_RxDesc_List_Address is shown in Figure 43-294 and described in Table 43-348.

Return to the Summary Table.

The Channeli Rx Descriptor List Address register points the DMA to the start of Receive descriptor list.
This register points to the start of the Receive Descriptor List. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LS bits low. Writing to this register is permitted only when reception is stopped. When stopped, this register must be written to before the receive Start command is given. You can write to this register only when Rx DMA has stopped, that is, SR bit is set to zero in DMA_CH0_Rx_Control register. When stopped, this register can be written with a new descriptor list address.
When you set the SR bit to 1, the DMA takes the newly programmed descriptor base address.

Figure 43-294. DMA_CH1_RxDesc_List_Address Register

31	30	29	28	27	26	25	24
RDESLA							
R/W-0h							
23	22	21	20	19	18	17	16
RDESLA							
R/W-0h							
15	14	13	12	11	10	9	8
RDESLA							
R/W-0h							
7	6	5	4	3	2	1	0
RDESLA						RESERVED	
R/W-0h						R-0h	

Table 43-348. DMA_CH1_RxDesc_List_Address Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RDESLA	R/W	0h	Start of Receive List This field contains the base address of the first descriptor in the Rx Descriptor list. The DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally takes these bits as all-zero. Therefore, these LSB bits are read-only (RO). The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration
1-0	RESERVED	R	0h	

43.7.3.256 DMA_CH1_TxDesc_Tail_Pointer Register (Offset = 11A0h) [Reset = 0h]

DMA_CH1_TxDesc_Tail_Pointer is shown in Figure 43-295 and described in Table 43-349.

Return to the Summary Table.

The Channeli Tx Descriptor Tail Pointer register points to an offset from the base and indicates the location of the last valid descriptor.

Figure 43-295. DMA_CH1_TxDesc_Tail_Pointer Register

31	30	29	28	27	26	25	24
TDTP							
R/W-0h							
23	22	21	20	19	18	17	16
TDTP							
R/W-0h							
15	14	13	12	11	10	9	8
TDTP							
R/W-0h							
7	6	5	4	3	2	1	0
TDTP						RESERVED	
R/W-0h						R-0h	

Table 43-349. DMA_CH1_TxDesc_Tail_Pointer Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	TDTP	R/W	0h	Transmit Descriptor Tail Pointer This field contains the tail pointer for the Tx descriptor ring. The software writes the tail pointer to add more descriptors to the Tx channel. The hardware tries to transmit all packets referenced by the descriptors between the head and the tail pointer registers. The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration
1-0	RESERVED	R	0h	

43.7.3.257 DMA_CH1_RxDesc_Tail_Pointer Register (Offset = 11A8h) [Reset = 0h]

DMA_CH1_RxDesc_Tail_Pointer is shown in Figure 43-296 and described in Table 43-350.

Return to the Summary Table.

The Channeli Rx Descriptor Tail Pointer Points to an offset from the base and indicates the location of the last valid descriptor.

Figure 43-296. DMA_CH1_RxDesc_Tail_Pointer Register

31	30	29	28	27	26	25	24
RDTP							
R/W-0h							
23	22	21	20	19	18	17	16
RDTP							
R/W-0h							
15	14	13	12	11	10	9	8
RDTP							
R/W-0h							
7	6	5	4	3	2	1	0
RDTP						RESERVED	
R/W-0h						R-0h	

Table 43-350. DMA_CH1_RxDesc_Tail_Pointer Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RDTP	R/W	0h	Receive Descriptor Tail Pointer This field contains the tail pointer for the Rx descriptor ring. The software writes the tail pointer to add more descriptors to the Rx channel. The hardware tries to write all received packets to the descriptors referenced between the head and the tail pointer registers. The width of this field depends on the configuration: - 31:2 for 32-bit configuration - 31:3 for 64-bit configuration - 31:4 for 128-bit configuration
1-0	RESERVED	R	0h	

43.7.3.258 DMA_CH1_TxDesc_Ring_Length Register (Offset = 11ACh) [Reset = 0h]

DMA_CH1_TxDesc_Ring_Length is shown in Figure 43-297 and described in Table 43-351.

Return to the Summary Table.

The Tx Descriptor Ring Length register contains the length of the Transmit descriptor ring.

Figure 43-297. DMA_CH1_TxDesc_Ring_Length Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						TDRL									
R-0h																						R/W-0h									

Table 43-351. DMA_CH1_TxDesc_Ring_Length Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved.
9-0	TDRL	R/W	0h	Transmit Descriptor Ring Length This field sets the maximum number of Tx descriptors in the circular descriptor ring. The maximum number of descriptors is limited to 1K descriptors. Synopsys recommends a minimum ring descriptor length of 4. For example, You can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. If you want to have 10 descriptors, program it to a value of 0x9.

43.7.3.259 DMA_CH1_RxDesc_Ring_Length Register (Offset = 11B0h) [Reset = 0h]

DMA_CH1_RxDesc_Ring_Length is shown in Figure 43-298 and described in Table 43-352.

Return to the Summary Table.

The Channeli Rx Descriptor Ring Length register contains the length of the Receive descriptor circular ring.

Figure 43-298. DMA_CH1_RxDesc_Ring_Length Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						RDRL									
R-0h																						R/W-0h									

Table 43-352. DMA_CH1_RxDesc_Ring_Length Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved.
9-0	RDRL	R/W	0h	Receive Descriptor Ring Length This register sets the maximum number of Rx descriptors in the circular descriptor ring. The maximum number of descriptors is limited to 1K descriptors. For example, You can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. If you want to have 10 descriptors, program it to a value of 0x9.

43.7.3.260 DMA_CH1_Interrupt_Enable Register (Offset = 11B4h) [Reset = 0h]

DMA_CH1_Interrupt_Enable is shown in Figure 43-299 and described in Table 43-353.

Return to the Summary Table.

The Channeli Interrupt Enable register enables the interrupts reported by the Status register.

Figure 43-299. DMA_CH1_Interrupt_Enable Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
NIE	AIE	CDEE	FBEE	ERIE	ETIE	RWTE	RSE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RBUE	RIE	RESERVED			TBUE	TXSE	TIE
R/W-0h	R/W-0h	R-0h			R/W-0h	R/W-0h	R/W-0h

Table 43-353. DMA_CH1_Interrupt_Enable Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved.
15	NIE	R/W	0h	Normal Interrupt Summary Enable When this bit is set, the normal interrupt summary is enabled. This bit enables the following interrupts in the DMA_CH0_Status register: - Bit 0: Transmit Interrupt - Bit 2: Transmit Buffer Unavailable - Bit 6: Receive Interrupt - Bit 11: Early Receive Interrupt When this bit is reset, the normal interrupt summary is disabled. 0h = Normal Interrupt Summary is disabled : 0x0 1h = Normal Interrupt Summary is enabled : 0x1
14	AIE	R/W	0h	Abnormal Interrupt Summary Enable When this bit is set, the abnormal interrupt summary is enabled. This bit enables the following interrupts in the DMA_CH0_Status register: - Bit 1: Transmit Process Stopped - Bit 7: Rx Buffer Unavailable - Bit 8: Receive Process Stopped - Bit 9: Receive Watchdog Timeout - Bit 10: Early Transmit Interrupt - Bit 12: Fatal Bus Error - Bit 13: Context Descriptor Error When this bit is reset, the abnormal interrupt summary is disabled. 0h = Abnormal Interrupt Summary is disabled : 0x0 1h = Abnormal Interrupt Summary is enabled : 0x1
13	CDEE	R/W	0h	Context Descriptor Error Enable When this bit is set along with the AIE bit, the Descriptor error interrupt is enabled. When this bit is reset, the Descriptor error interrupt is disabled. 0h = Context Descriptor Error is disabled : 0x0 1h = Context Descriptor Error is enabled : 0x1

Table 43-353. DMA_CH1_Interrupt_Enable Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	FBEE	R/W	0h	Fatal Bus Error Enable When this bit is set along with the AIE bit, the Fatal Bus error interrupt is enabled. When this bit is reset, the Fatal Bus Error error interrupt is disabled. 0h = Fatal Bus Error is disabled : 0x0 1h = Fatal Bus Error is enabled : 0x1
11	ERIE	R/W	0h	Early Receive Interrupt Enable When this bit is set along with the NIE bit, the Early Receive interrupt is enabled. When this bit is reset, the Early Receive interrupt is disabled. 0h = Early Receive Interrupt is disabled : 0x0 1h = Early Receive Interrupt is enabled : 0x1
10	ETIE	R/W	0h	Early Transmit Interrupt Enable When this bit is set along with the AIE bit, the Early Transmit interrupt is enabled. When this bit is reset, the Early Transmit interrupt is disabled. 0h = Early Transmit Interrupt is disabled : 0x0 1h = Early Transmit Interrupt is enabled : 0x1
9	RWTE	R/W	0h	Receive Watchdog Timeout Enable When this bit is set along with the AIE bit, the Receive Watchdog Timeout interrupt is enabled. When this bit is reset, the Receive Watchdog Timeout interrupt is disabled. 0h = Receive Watchdog Timeout is disabled : 0x0 1h = Receive Watchdog Timeout is enabled : 0x1
8	RSE	R/W	0h	Receive Stopped Enable When this bit is set along with the AIE bit, the Receive Stopped Interrupt is enabled. When this bit is reset, the Receive Stopped interrupt is disabled. 0h = Receive Stopped is disabled : 0x0 1h = Receive Stopped is enabled : 0x1
7	RBUE	R/W	0h	Receive Buffer Unavailable Enable When this bit is set along with the AIE bit, the Receive Buffer Unavailable interrupt is enabled. When this bit is reset, the Receive Buffer Unavailable interrupt is disabled. 0h = Receive Buffer Unavailable is disabled : 0x0 1h = Receive Buffer Unavailable is enabled : 0x1
6	RIE	R/W	0h	Receive Interrupt Enable When this bit is set along with the NIE bit, the Receive Interrupt is enabled. When this bit is reset, the Receive Interrupt is disabled. 0h = Receive Interrupt is disabled : 0x0 1h = Receive Interrupt is enabled : 0x1
5-3	RESERVED	R	0h	Reserved.
2	TBUE	R/W	0h	Transmit Buffer Unavailable Enable When this bit is set along with the NIE bit, the Transmit Buffer Unavailable interrupt is enabled. When this bit is reset, the Transmit Buffer Unavailable interrupt is disabled. 0h = Transmit Buffer Unavailable is disabled : 0x0 1h = Transmit Buffer Unavailable is enabled : 0x1
1	TXSE	R/W	0h	Transmit Stopped Enable When this bit is set along with the AIE bit, the Transmission Stopped interrupt is enabled. When this bit is reset, the Transmission Stopped interrupt is disabled. 0h = Transmit Stopped is disabled : 0x0 1h = Transmit Stopped is enabled : 0x1
0	TIE	R/W	0h	Transmit Interrupt Enable When this bit is set along with the NIE bit, the Transmit Interrupt is enabled. When this bit is reset, the Transmit Interrupt is disabled. 0h = Transmit Interrupt is disabled : 0x0 1h = Transmit Interrupt is enabled : 0x1

43.7.3.261 DMA_CH1_Rx_Interrupt_Watchdog_Timer Register (Offset = 11B8h) [Reset = 0h]

DMA_CH1_Rx_Interrupt_Watchdog_Timer is shown in Figure 43-300 and described in Table 43-354.

Return to the Summary Table.

The Receive Interrupt Watchdog Timer register indicates the watchdog timeout for Receive Interrupt (RI) from the DMA. When this register is written with a non-zero value, it enables the watchdog timer for the RI bit of the DMA_CHi_Status register.

Figure 43-300. DMA_CH1_Rx_Interrupt_Watchdog_Timer Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED														RWTU	
R-0h														R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RWT							
R-0h								R/W-0h							

Table 43-354. DMA_CH1_Rx_Interrupt_Watchdog_Timer Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved.
17-16	RWTU	R/W	0h	Receive Interrupt Watchdog Timer Count Units This fields indicates the number of system clock cycles corresponding to one unit in RWT field. - 2'b00: 256 - 2'b01: 512 - 2'b10: 1024 - 2'b11: 2048 For example, when RWT=2 and RWTU=1, the watchdog timer is set for 2*512=1024 system clock cycles.
15-8	RESERVED	R	0h	Reserved.
7-0	RWT	R/W	0h	Receive Interrupt Watchdog Timer Count This field indicates the number of system clock cycles, multiplied by factor indicated in RWTU field, for which the watchdog timer is set. The watchdog timer is triggered with the programmed value after the Rx DMA completes the transfer of a packet for which the RI bit is not set in the DMA_CH0_Status register, because of the setting of Interrupt Enable bit in the corresponding descriptor RDES3[30]. When the watchdog timer runs out, the RI bit is set and the timer is stopped. The watchdog timer is reset when the RI bit is set high because of automatic setting of RI as per the Interrupt Enable bit RDES3[30] of any received packet.

43.7.3.262 DMA_CH1_Current_App_TxDesc Register (Offset = 11C4h) [Reset = 0h]

DMA_CH1_Current_App_TxDesc is shown in Figure 43-301 and described in Table 43-355.

Return to the Summary Table.

The Channeli Current Application Transmit Descriptor register points to the current Transmit descriptor read by the DMA.

Figure 43-301. DMA_CH1_Current_App_TxDesc Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CURTDESAPTR																															
R-0h																															

Table 43-355. DMA_CH1_Current_App_TxDesc Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CURTDESAPTR	R	0h	Application Transmit Descriptor Address Pointer The DMA updates this pointer during Tx operation. This pointer is cleared on reset.

43.7.3.263 DMA_CH1_Current_App_RxDesc Register (Offset = 11CCh) [Reset = 0h]

DMA_CH1_Current_App_RxDesc is shown in Figure 43-302 and described in Table 43-356.

Return to the Summary Table.

The Channeli Current Application Receive Descriptor register points to the current Receive descriptor read by the DMA.

Figure 43-302. DMA_CH1_Current_App_RxDesc Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CURRDESAPTR																															
R-0h																															

Table 43-356. DMA_CH1_Current_App_RxDesc Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CURRDESAPTR	R	0h	Application Receive Descriptor Address Pointer The DMA updates this pointer during Rx operation. This pointer is cleared on reset.

43.7.3.264 DMA_CH1_Current_App_TxBuffer Register (Offset = 11D4h) [Reset = 0h]

DMA_CH1_Current_App_TxBuffer is shown in Figure 43-303 and described in Table 43-357.

Return to the Summary Table.

The Channeli Current Application Transmit Buffer Address register points to the current Tx buffer address read by the DMA.

Figure 43-303. DMA_CH1_Current_App_TxBuffer Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CURTBUFAPTR																															
R-0h																															

Table 43-357. DMA_CH1_Current_App_TxBuffer Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CURTBUFAPTR	R	0h	Application Transmit Buffer Address Pointer The DMA updates this pointer during Tx operation. This pointer is cleared on reset.

43.7.3.265 DMA_CH1_Current_App_RxBuffer Register (Offset = 11DCh) [Reset = 0h]

DMA_CH1_Current_App_RxBuffer is shown in Figure 43-304 and described in Table 43-358.

Return to the Summary Table.

The Channel 0 Current Application Receive Buffer Address register points to the current Rx buffer address read by the DMA.

Figure 43-304. DMA_CH1_Current_App_RxBuffer Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CURRBUFAPTR																															
R-0h																															

Table 43-358. DMA_CH1_Current_App_RxBuffer Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CURRBUFAPTR	R	0h	Application Receive Buffer Address Pointer The DMA updates this pointer during Rx operation. This pointer is cleared on reset.

43.7.3.266 DMA_CH1_Status Register (Offset = 11E0h) [Reset = 0h]

DMA_CH1_Status is shown in Figure 43-305 and described in Table 43-359.

Return to the Summary Table.

The software driver (application) reads the Status register during interrupt service routine or polling to determine the status of the DMA.
Note: The number of DMA_CH[n]_Status register in the configuration is the higher of number of Rx DMA Channels and Tx DMA Channels.

Figure 43-305. DMA_CH1_Status Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED										REB			TEB		
R-0h										R-0h			R-0h		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NIS	AIS	CDE	FBE	ERI	ETI	RWT	RPS	RBU	RI	RESERVED			TBU	TPS	TI
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h			R/W-0h	R/W-0h	R/W-0h

Table 43-359. DMA_CH1_Status Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reserved.
21-19	REB	R	0h	Rx DMA Error Bits This field indicates the type of error that caused a Bus Error. For example, error response on the AHB or AXI interface. Bit 21 - 1'b1: Error during data transfer by Rx DMA - 1'b0: No Error during data transfer by Rx DMA Bit 20 - 1'b1: Error during descriptor access - 1'b0: Error during data buffer access Bit 19 - 1'b1: Error during read transfer - 1'b0: Error during write transfer This field is valid only when the FBE bit is set. This field does not generate an interrupt.
18-16	TEB	R	0h	Tx DMA Error Bits This field indicates the type of error that caused a Bus Error. For example, error response on the AHB or AXI interface. Bit 18 - 1'b1: Error during data transfer by Tx DMA - 1'b0: No Error during data transfer by Tx DMA Bit 17 - 1'b1: Error during descriptor access - 1'b0: Error during data buffer access Bit 16 - 1'b1: Error during read transfer - 1'b0: Error during write transfer This field is valid only when the FBE bit is set. This field does not generate an interrupt.

Table 43-359. DMA_CH1_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15	NIS	R/W	0h	Normal Interrupt Summary Normal Interrupt Summary bit value is the logical OR of the following bits when the corresponding interrupt bits are enabled in the DMA_CH0_Interrupt_Enable register: - Bit 0: Transmit Interrupt - Bit 2: Transmit Buffer Unavailable - Bit 6: Receive Interrupt - Bit 11: Early Receive Interrupt Only unmasked bits (interrupts for which interrupt enable is set in DMA_CH0_Interrupt_Enable register) affect the Normal Interrupt Summary bit. This is a sticky bit. You must clear this bit (by writing 1 to this bit) each time a corresponding bit which causes NIS to be set is cleared. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Normal Interrupt Summary status not detected : 0x0 1h = Normal Interrupt Summary status detected : 0x1
14	AIS	R/W	0h	Abnormal Interrupt Summary Abnormal Interrupt Summary bit value is the logical OR of the following when the corresponding interrupt bits are enabled in the DMA_CH0_Interrupt_Enable register: - Bit 1: Transmit Process Stopped - Bit 7: Receive Buffer Unavailable - Bit 8: Receive Process Stopped - Bit 10: Early Transmit Interrupt - Bit 12: Fatal Bus Error - Bit 13: Context Descriptor Error Only unmasked bits affect the Abnormal Interrupt Summary bit. This is a sticky bit. You must clear this bit (by writing 1 to this bit) each time a corresponding bit, which causes AIS to be set, is cleared. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Abnormal Interrupt Summary status not detected : 0x0 1h = Abnormal Interrupt Summary status detected : 0x1
13	CDE	R/W	0h	Context Descriptor Error This bit indicates that the DMA Tx/Rx engine received a descriptor error, which indicates invalid context in the middle of packet flow (intermediate descriptor) or all one's descriptor in Tx case and on Rx side it indicates DMA has read a descriptor with either of the buffer address as ones which is considered to be invalid. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Context Descriptor Error status not detected : 0x0 1h = Context Descriptor Error status detected : 0x1
12	FBE	R/W	0h	Fatal Bus Error This bit indicates that a bus error occurred (as described in the EB field). When this bit is set, the corresponding DMA channel engine disables all bus accesses. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Fatal Bus Error status not detected : 0x0 1h = Fatal Bus Error status detected : 0x1
11	ERI	R/W	0h	Early Receive Interrupt This bit when set indicates that the RxDMA has completed the transfer of packet data to the memory. When ERIC=0, this bit is set only after the Rx DMA has filled up a complete receive buffer with packet data. When ERIC=1, this bit is set after every burst transfer of data from the Rx DMA to the buffer. The setting of RI bit automatically clears this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Early Receive Interrupt status not detected : 0x0 1h = Early Receive Interrupt status detected : 0x1

Table 43-359. DMA_CH1_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	ETI	R/W	0h	Early Transmit Interrupt This bit when set indicates that the TxDMA has completed the transfer of packet data to the MTL TXFIFO memory. When ETIC=0, this bit is set only after the Tx DMA has transferred a complete packet to MTL. When ETIC=1, this bit is set after completion of (partial) packet data transfer from buffers in the Transmit descriptor in which IOC=1. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Early Transmit Interrupt status not detected : 0x0 1h = Early Transmit Interrupt status detected : 0x1
9	RWT	R/W	0h	Receive Watchdog Timeout This bit is asserted when a packet with length greater than 2,048 bytes (10,240 bytes when Jumbo Packet mode is enabled) is received. 0h = Receive Watchdog Timeout status not detected : 0x0 1h = Receive Watchdog Timeout status detected : 0x1
8	RPS	R/W	0h	Receive Process Stopped This bit is asserted when the Rx process enters the Stopped state. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Receive Process Stopped status not detected : 0x0 1h = Receive Process Stopped status detected : 0x1
7	RBU	R/W	0h	Receive Buffer Unavailable This bit indicates that the application owns the next descriptor in the Receive list, and the DMA cannot acquire it. The Rx process is suspended. To resume processing Rx descriptors, the application should change the ownership of the descriptor and issue a Receive Poll Demand command. If this command is not issued, the Rx process resumes when the next recognized incoming packet is received. In ring mode, the application should advance the Receive Descriptor Tail Pointer register of a channel. This bit is set only when the DMA owns the previous Rx descriptor. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Receive Buffer Unavailable status not detected : 0x0 1h = Receive Buffer Unavailable status detected : 0x1
6	RI	R/W	0h	Receive Interrupt This bit indicates that the packet reception is complete. When packet reception is complete, Bit 31 of RDES3 is reset in the last descriptor, and the specific packet status information is updated in the descriptor. The reception remains in the Running state. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Receive Interrupt status not detected : 0x0 1h = Receive Interrupt status detected : 0x1
5-3	RESERVED	R	0h	Reserved.

Table 43-359. DMA_CH1_Status Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	TBU	R/W	0h	Transmit Buffer Unavailable This bit indicates that the application owns the next descriptor in the Transmit list, and the DMA cannot acquire it. Transmission is suspended. The TPS0 field of the DMA_Debug_Status0 register explains the Transmit Process state transitions. To resume processing the Transmit descriptors, the application should do the following: 1. Change the ownership of the descriptor by setting Bit 31 of TDES3. 2. Issue a Transmit Poll Demand command. For ring mode, the application should advance the Transmit Descriptor Tail Pointer register of a channel. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Transmit Buffer Unavailable status not detected : 0x0 1h = Transmit Buffer Unavailable status detected : 0x1
1	TPS	R/W	0h	Transmit Process Stopped This bit is set when the transmission is stopped. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Transmit Process Stopped status not detected : 0x0 1h = Transmit Process Stopped status detected : 0x1
0	TI	R/W	0h	Transmit Interrupt This bit indicates that the packet transmission is complete. When transmission is complete, Bit 31 of TDES3 is reset in the last descriptor, and the specific packet status information is updated in the descriptor. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. 0h = Transmit Interrupt status not detected : 0x0 1h = Transmit Interrupt status detected : 0x1

43.7.3.267 DMA_CH1_Miss_Frame_Cnt Register (Offset = 11E4h) [Reset = 0h]

DMA_CH1_Miss_Frame_Cnt is shown in Figure 43-306 and described in Table 43-360.

Return to the Summary Table.

This register has the number of packet counter that got dropped by the DMA either due to Bus Error or due to programming RPF field in DMA_CH${i}_Rx_Control register.

Figure 43-306. DMA_CH1_Miss_Frame_Cnt Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
MFCO	RESERVED				MFC		
R-0h	R-0h				R-0h		
7	6	5	4	3	2	1	0
MFC							
R-0h							

Table 43-360. DMA_CH1_Miss_Frame_Cnt Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved.
15	MFCO	R	0h	Overflow status of the MFC Counter When this bit is set then the MFC counter does not get incremented further. The bit gets cleared when this register is read. Access restriction applies. Clears on read. Self-set to 1 on internal event.
14-11	RESERVED	R	0h	Reserved.
10-0	MFC	R	0h	Dropped Packet Counters This counter indicates the number of packet counters that are dropped by the DMA either because of bus error or because of programing RPF field in DMA_CH${i}_Rx_Control register. The counter gets cleared when this register is read. Access restriction applies. Clears on read. Self-set to 1 on internal event.

43.7.3.268 DMA_CH1_RX_ERI_Cnt Register (Offset = 11ECh) [Reset = 0h]

DMA_CH1_RX_ERI_Cnt is shown in Figure 43-307 and described in Table 43-361.

Return to the Summary Table.

Figure 43-307. DMA_CH1_RX_ERI_Cnt Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																				ECNT											
R-0h																				R-0h											

Table 43-361. DMA_CH1_RX_ERI_Cnt Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	
11-0	ECNT	R	0h	ERI Counter When ERIC bit of DMA_CH(#i)_RX_Control register is set, this counter increments for burst transfer completed by the Rx DMA from the start of packet transfer. This counter will get reset at the start of new packet.

Chapter 44

Generic Cyclic Redundancy Check (GCRC)

The Generic Cyclic Redundancy Check (GCRC) is a module that computes the cyclic redundancy check value of a specified block of data.

44.1 Generic CRC Overview

The Generic CRC (GCRC) is a designated Connectivity Manager module for computing the CRC value on a configurable block of memory. It accomplishes this by fetching the specified block of memory and using the integrated CRC engine. The calculated CRC value can be compared against a golden CRC value in software to indicate a pass or fail. In essence, the GCRC can help identify memory faults and corruption in the Conectivity Manager's accessible raw data.

44.1.1 GCRC Features

The Generic CRC (GCRC) module has the following features:

- Support programmable polynomials of any order between 1 and 32
- Calculate a CRC on byte (8-bit), half word (16-bit), and word (32-bit) data blocks
- Define the endianness and datatype of the source data
- Reverse the bit order
- Select which data bits participate in the CRC computation
- Single Cycle CRC calculation with fixed polynomial

44.1.2 GCRC Block Diagram

Figure 44-1 shows the block diagram of the GCRC module.

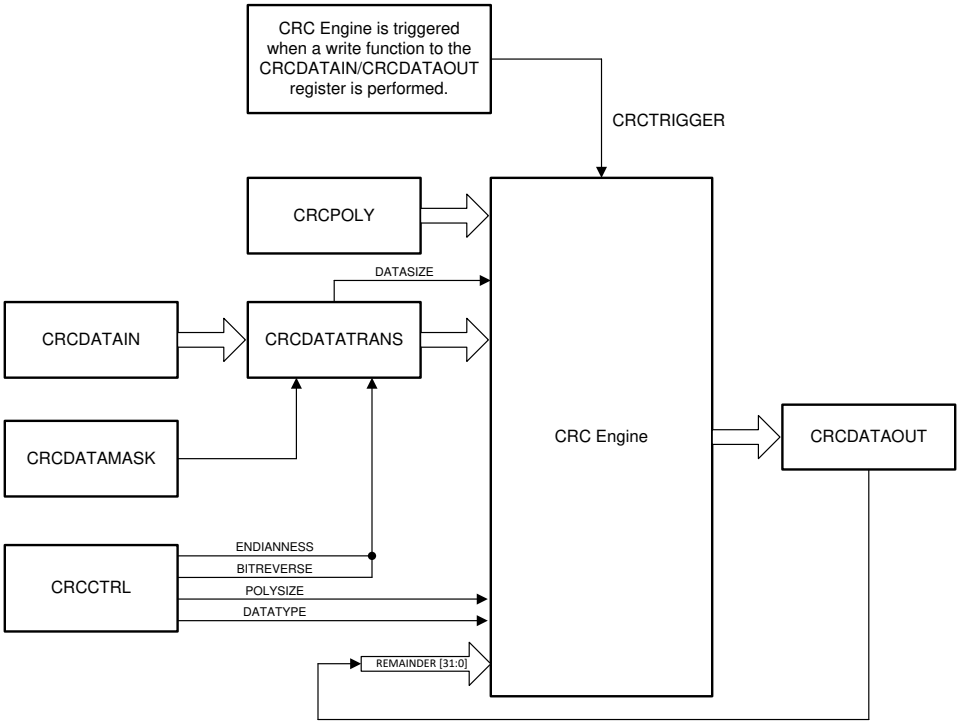


Figure 44-1. GCRC Block Diagram

The CRC is computed as:

$$M.x^n = G(x).Q(x) + R(x) \quad (24)$$

Where:

M : Is the data string that CRC is to be computed in the GF(2^n) polynomial representation

G(x) : Is the generator polynomial in the GF(2^n) field

R(x) : Is the CRC value

Q(x) : Is the quotient.

n : Is the order of the generator polynomial

44.2 GCRC Functional Description

44.2.1 GCRC Polynomials

The GCRC module allows the user to set a custom polynomial value to apply during the CRC calculation. This value is represented in hexadecimal format. The CRCCTRL.POLYSIZE should be set based on the size of the polynomial used in the CRCPOLY register.

44.2.2 Fixed Polynomial

If the polynomial value is not important, a fixed polynomial data path is provided. This fixed data path computes the CRC of a given data set in a single cycle. This mode requires the register setup in Table 44-1.

Table 44-1. Fixed Polynomial Data Path Settings

Register Field	Value
CRCPOLY.POLY	0x04c11db7
CRCCTRL.POLYSIZE	0x20
CRCCTRL.ENDIANNESS	0x0
CRCCTRL.BITREVERSE	0x0
CRCCTRL.DATATYPE	0x2
CRCDATAMASK.DATAMASK	0x0

44.2.3 GCRC Data Input

A write to the CRCDATAIN register will initiate the CRC computation sequence.

44.2.4 GCRC Execution Sequence Flow

Figure 44-2 explains the typical sequence flow for executing a cyclic redundancy check.

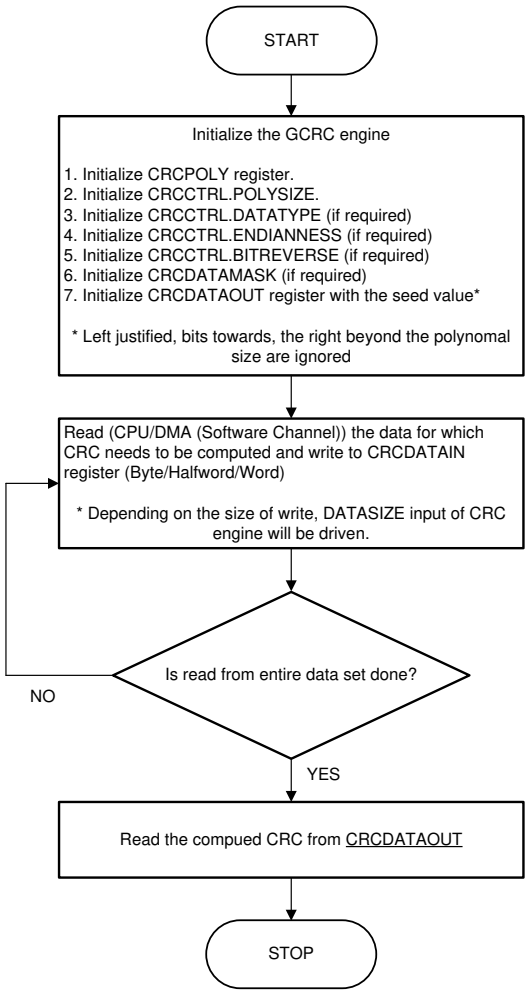


Figure 44-2. CRC Sequence Flow

Note

The Seed Value (CRCDATAOUT) should only be written after all the other configuration registers have been initialized to ensure proper functioning of the GCRC module.

44.2.5 GCRC Transformations

44.2.5.1 Endianness Transformation

The GCRC module is capable of transforming the data that is written to the DATAIN register. This data is stored in the CRCDATATRANS register and can be used for debugging purposes. The module is capable of applying three different transformations to the data provided by the user. The first is the endianness transformation. Table 44-2 shows how setting this byte in the control register influences the data that will be used for the CRC.

Table 44-2. Endianness Table

DATATYPE	Endianness	DATASIZE	DATAIN	DATAOUT	Valid Bits for CRC Calculation (BITS_VALID)
8	Little	8	0x------ab	0x------ab	8
8	Little	16	0x----abcd	0x----abcd	16
8	Little	32	0xabcdefgh	0xabcdefgh	32
8	Big	16	0x----abcd 0xabcd---	0xcdab0000	16
8	Big	32	0xabcdefgh	0xghefcdab	32
16	Little	16	0x----abcd	0x----abcd	16
16	Little	32	0xabcdefgh	0xabcdefgh	32
16	Big	32	0xabcdefgh	0xefghabcd	32
32	Little	32	0xabcdefgh	0xabcdefgh	32
32	Big	32	0xabcdefgh	0xabcdefgh	32
All Other Combinations		8	0x------ab 0x----ab-- 0x--ab---- 0xab------	0xab000000	8
		16	0x----abcd 0xabcd---	0xabcd0000	16
		32	0xabcdefgh	0xabcdefgh	32

44.2.5.2 Mask Transformation

The second transformation that the module is capable of applying is the data mask transformation. Table 44-3 shows how to utilize the data mask byte to control which data is used for the CRC calculation.

Table 44-3. Data Mask Table

BITS_VALID_IN	DATAIN	DATAMASK Example	DATAOUT	BITS_VALID_OUT
8	0bABCDEFGH_00000000_00000000_00000000	0x2	0bCDEFGH00_00000000_00000000_00000000	6
16	0bABCDEFGH_IJKLMNOP_00000000_00000000	0x9	0bJKLMNOP0_00000000_00000000_00000000	7
32	0bABCDEFGH_IJKLMNOP_QRSTUVWX_Yzabcdef	0x11	0bRSTUVWXY_zabcdef0_00000000_00000000	15

44.2.5.3 Bit Reversal Transformation

The last transformation the GCRC module is capable of applying to the input data is bit reversal. Table 44-4 shows the bit reversal modification being applied to the input data.

Table 44-4. Bit Reverse Table

BITS_VALID_IN	Example DATAIN	DATAOUT	BIT REVERSE
6	0bABDEFH00_00000000_00000000_00000000	0bHFEDBA00_00000000_00000000_00000000	1
12	0bABDEFHIJ_KNOP0000_00000000_00000000	0bPONKJIHF_EDBA0000_00000000_00000000	1
23	0bABDEFHIJ_KNOPQRST_XYzcdef0_00000000	0bfedczYXT_SRQPONKJ_IHFEDBA0_00000000	1
-	DATAIN	DATAOUT=DATAIN	0

44.3 Software

44.3.1 GCRC Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/gcrc

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

44.3.1.1 GCRC example - CM

FILE: gcrc_ex1_basic.c

This example showcases how to use GCRC to compute CRC for a 8-bit array. This demonstrates 2 methods for computing the CRC cm_common_config_c28x example needs to be run on C28x1 before running this example on the CM core

External Connections

- None

Watch Variables

- *crcResult1* - CRC value computed using Method 1
- *crcResult2* - CRC value computed using Method 2
- *crcGolden* - Golden CRC value

44.4 GCRC Registers

This section describes the Generic Cyclic Redunancy Check Registers.

44.4.1 GCRC Base Addresses

Table 44-5. GCRC Base Address Table (CM)

DriverLib Name	Base Address	uDMA Access	Ethernet DMA Access
GCRC_BASE	0x4004_0000	YES	-

44.4.2 GCRC_REGS Registers

Table 44-6 lists the memory-mapped registers for the GCRC_REGS registers. All register offset addresses not listed in Table 44-6 should be considered as reserved locations and the register contents should not be modified.

Table 44-6. GCRC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CRCCTRL	CRC Control Register		Go
4h	CRCPOLY	CRC Polynomial Register		Go
8h	CRCDATAMASK	CRC Data Mask Register		Go
Ch	CRCDATAIN	CRC Data Input Register		Go
10h	CRCDATAOUT	CRC Data Output Register		Go
14h	CRCDATATRANS	CRC Transformed Data Register		Go

Complex bit access types are encoded to fit into small table cells. Table 44-7 shows the codes that are used for access types in this section.

Table 44-7. GCRC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

44.4.2.1 CRCCTRL Register (Offset = 0h) [Reset = 220h]

CRCCTRL is shown in Figure 44-3 and described in Table 44-8.

Return to the Summary Table.

This is the Control Register for GCRC module.

Figure 44-3. CRCCTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						DATATYPE	
R-0h						R/W-2h	
7	6	5	4	3	2	1	0
BITREVERSE	ENDIANNESS	POLYSIZE					
R/W-0h	R/W-0h	R/W-20h					

Table 44-8. CRCCTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9-8	DATATYPE	R/W	2h	Defines the datatype of the element of the data array, on which CRC is to be computed. 00 : Byte data type 01 : 16 bit data type 10 : 32 bit data type 11 : Reserved Note: This field works in conjunction with ENDIANNESS field. Reset type: CM.RESETn
7	BITREVERSE	R/W	0h	0: DATAIN bus to the CRC engine is sent as is. 1: DATAIN bus to the CRC engine is bit reversed. Reset type: CM.RESETn
6	ENDIANNESS	R/W	0h	0: Little endian. Endianness applies to word and half word writes. 1: Big endian. Endianness applies to word and half word writes. Note: This field works in conjunction with DATATYPE field. Reset type: CM.RESETn
5-0	POLYSIZE	R/W	20h	POLYSIZE: The value in this field determines the order of the polynomial. For example a value of 0x20 implies an order of 32,a value of 0x18 implies an order of 24 and a value of 0x10 implies an order of 16 and so on. Note: A value of 0x0 is not valid and values more than 0x20 will be treated as 0x20. Reset type: CM.RESETn

44.4.2.2 CRCPOLY Register (Offset = 4h) [Reset = 04C11DB7h]

CRCPOLY is shown in Figure 44-4 and described in Table 44-9.

Return to the Summary Table.

This register is used to store the Polynomial to used for the CRC calculation.

Figure 44-4. CRCPOLY Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POLY																															
R/W-04C11DB7h																															

Table 44-9. CRCPOLY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	POLY	R/W	04C11DB7h	Polynomial value. For example, if the 1.x^3 + 0.x^2 + 1.x^1 + 1.x^0 (CRC3), taking the coeffecients and forming a binary string would result in 1011. The value to be programmed in this field is 11 (0x3), as MSB is assumed to be 1. Note: CRCCTRL.POLYSIZE should be programmed to 0x3 to indicate it is a CRC3 polynomial. Note: A value of zero for the POLY field in invalid and the behavior is not deterministic. Reset type: CM.RESETn

44.4.2.3 CRCDATAMASK Register (Offset = 8h) [Reset = 0h]

CRCDATAMASK is shown in Figure 44-5 and described in Table 44-10.

Return to the Summary Table.

This register is used to apply a data mask to the input data and decide which bits are used in CRC calculation.

Figure 44-5. CRCDATAMASK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											DATAMASK				
R-0h											R/W-0h				

Table 44-10. CRCDATAMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4-0	DATAMASK	R/W	0h	Number of bits to be masked. For example, DATAMASK equals 0x3, 3 most significant bits of data will be masked. Note: For byte data type (DATATYPE = 0x0) DATAMASK < 8, For 16 bit data type (DATATYPE = 0x1) DATAMASK < 16, For 32 bit data type (DATATYPE = 0x2) DATAMASK < 32, Reset type: CM.RESETn

44.4.2.4 CRCDATAIN Register (Offset = Ch) [Reset = 0h]

CRCDATAIN is shown in Figure 44-6 and described in Table 44-11.

Return to the Summary Table.

This register contains the raw input data (before transformations) that will be used in the CRC calculation.

Figure 44-6. CRCDATAIN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATAIN																															
R/W-0h																															

Table 44-11. CRCDATAIN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATAIN	R/W	0h	DATAIN field: This register holds the last value on which CRC was computed. Reset type: CM.RESETn

44.4.2.5 CRCDATAOUT Register (Offset = 10h) [Reset = 0h]

CRCDATAOUT is shown in Figure 44-7 and described in Table 44-12.

Return to the Summary Table.

This register contains the calculated CRC value. It is also used to set an initial seed value for the CRC calculation.

Figure 44-7. CRCDATAOUT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 44-12. CRCDATAOUT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	CRCDATAOUT value: This register stores the computed CRC, It can be initialized to the required seed value before the CRC computation begins, in which case it will act as a seed register. When this register is written, the value written into this register is considered as data (with a seed value of 0) and the CRC for this value is updated back to CRCDATAOUT register (This is required for Type-2 CRC implementation). Note: Seed value needs to be written only after all the other configuration registers are initialized for proper functioning of CRC module. Reset type: CM.RESETn

44.4.2.6 CRCDATATRANS Register (Offset = 14h) [Reset = 0h]

CRCDATATRANS is shown in Figure 44-8 and described in Table 44-13.

Return to the Summary Table.

This register contains the actual data that will be used for the CRC calculations. This is the value of CRCDATAIN after the bit mask, bit reverse, and data endianess transformations.

Figure 44-8. CRCDATATRANS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATAIN																															
R-0h																															

Table 44-13. CRCDATATRANS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATAIN	R	0h	DATAIN field: This register holds the the data value as seen by the core CRC engine after endianess, bit reversal and masking transformation. Can be used for debug. Reset type: CM.RESETn

Chapter 45

Modular Controller Area Network (MCAN)

This chapter describes the Modular Controller Area Network (MCAN). MCAN supports both classic CAN and CAN FD protocols.

45.1 MCAN Introduction

The Controller Area Network (CAN) is a serial communications protocol that efficiently supports distributed real-time control with a high level of reliability. CAN has high immunity to electrical interference and the ability to detect various type of errors. In CAN, many short messages are broadcast to the entire network, which provides data consistency in every node of the system.

The MCAN module supports both classic CAN and CAN FD (CAN with flexible data-rate) protocols. The CAN FD feature allows higher throughput and increased payload per data frame. Classic CAN and CAN FD devices can coexist on the same network without any conflict provided that partial network transceivers, which can detect and ignore CAN FD without generating bus errors, are used by the classic CAN devices. The MCAN module is compliant to ISO 11898-1:2015.

Note

The availability of the CAN FD feature is dependent on the device's part number. Refer to the device data sheet for more information.

Figure 45-1 shows an overview of the MCAN module.

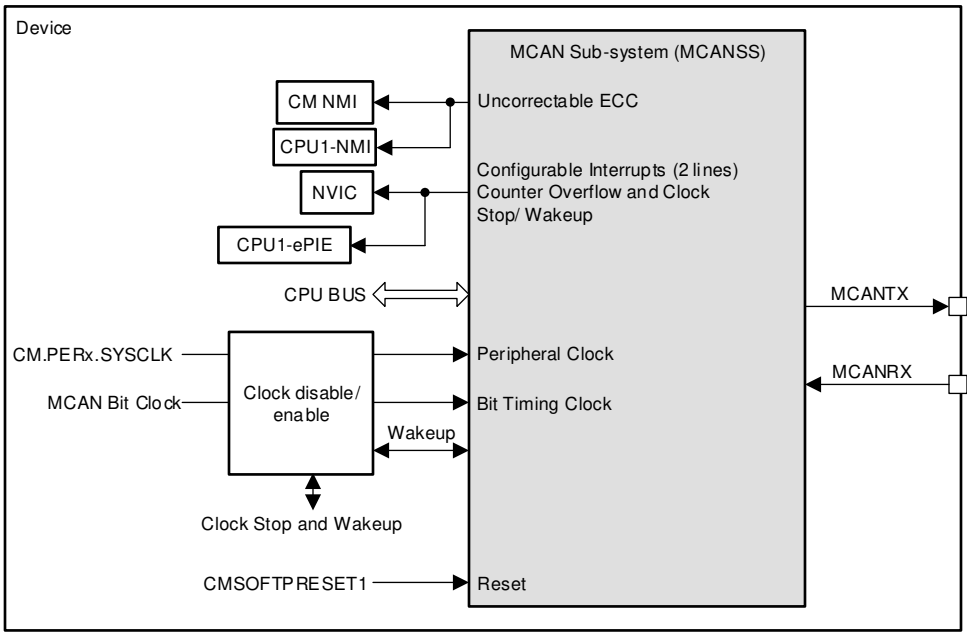


Figure 45-1. MCAN Module Overview

45.1.1 MCAN Related Collateral

Foundational Materials

- C2000 Academy - MCAN
- CAN and CAN FD Overview (Video)
- CAN and CAN FD Protocol (Video)

Getting Started Materials

- *Getting Started with the MCAN (CAN FD) Module Application Report*

45.1.2 MCAN Features

The MCAN module implements the following features:

- Conforms with CAN Protocol 2.0 A, B and ISO 11898-1:2015
- Full CAN FD support (up to 64 data bytes)
- AUTOSAR and SAE J1939 support
- Up to 32 dedicated transmit buffers
- Configurable transmit FIFO, up to 32 elements
- Configurable transmit queue, up to 32 elements
- Configurable transmit Event FIFO, up to 32 elements
- Up to 64 dedicated receive buffers
- Two configurable receive FIFOs, up to 64 elements each
- Up to 128 filter elements
- Loop-back mode for self-test
- Maskable interrupt (two configurable interrupt lines, correctable ECC, counter overflow, and clock stop or wakeup)
- Non-maskable interrupt (uncorrectable ECC)
- Two clock domains (CAN clock and host clock)
- ECC check for Message RAM
- Clock stop and wakeup support
- Timestamp counter
- 1-Mbps nominal bit rate, 5-Mbps data bit rate

Non-supported features:

- Host bus firewall
- Clock calibration
- Debug over CAN

45.2 MCAN Environment

The CAN network physical layer consists of a two-wire differential bus, usually twisted pair, and provides a high level of interference immunity. An external CAN transceiver IC is needed to access the bus.

Figure 45-2 shows typical MCAN wiring. Table 45-1 describes the external signals of the MCAN module.

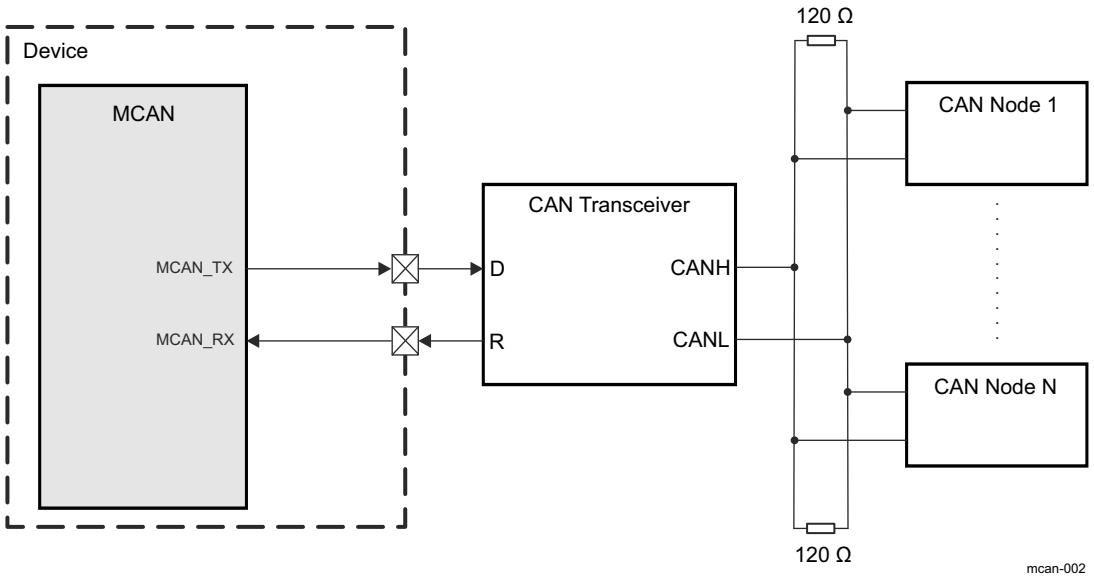


Figure 45-2. MCAN Typical Bus Wiring

Table 45-1. MCAN I/O Description

Module Signal	I/O	Description	Value at Reset
MCAN_RX	Input	Serial data input from external CAN transceiver.	HiZ
MCAN_TX	Output	Serial data output to external CAN transceiver.	HiZ

Note

See the *Terminal Configurations and Functions* section in the device data sheet and the *General-Purpose Input/Output (GPIO)* chapter to configure this peripheral to be connected to the device pins.

45.3 CAN Network Basics

The network basics are:

- The CAN bus is a 2-wire differential bus using non-return-to-zero (NRZ) encoding and has two states:
  - Recessive state (logical 1)
  - Dominant state (logical 0)
- When the bus is idle, any node can initiate a transmission to any other node (or nodes). When two or more nodes (ECUs) attempt to transmit at the same time, a non-destructive arbitration technique makes sure messages are sent in order of priority and no messages are lost.
- The message transmission is multicast. Data messages transmitted are identifier-based, not address-based.
- The content of the message is labeled by the identifier that is unique throughout the network (for example: RPM, temperature, position, pressure, and so forth).
- All nodes on the network receive the message and each performs an acceptance test on the identifier. If the message is relevant, the message is processed; otherwise, the message is ignored.
- The unique identifier also determines the priority of the message (the lower the numerical value of the identifier, the higher the priority).
- Data is transmitted and received using message frames, consisting of the following basic fields:
  - Arbitration field
  - Control field
  - Data field (up to 8 bytes for classical CAN and up to 64 bytes for CAN FD)
  - CRC field
  - ACK field

For more information, see *ISO 11898-1:2015: CAN data link layer and physical signaling*.

45.4 MCAN Integration

Figure 45-3 shows the integration of the MCAN module in the device.

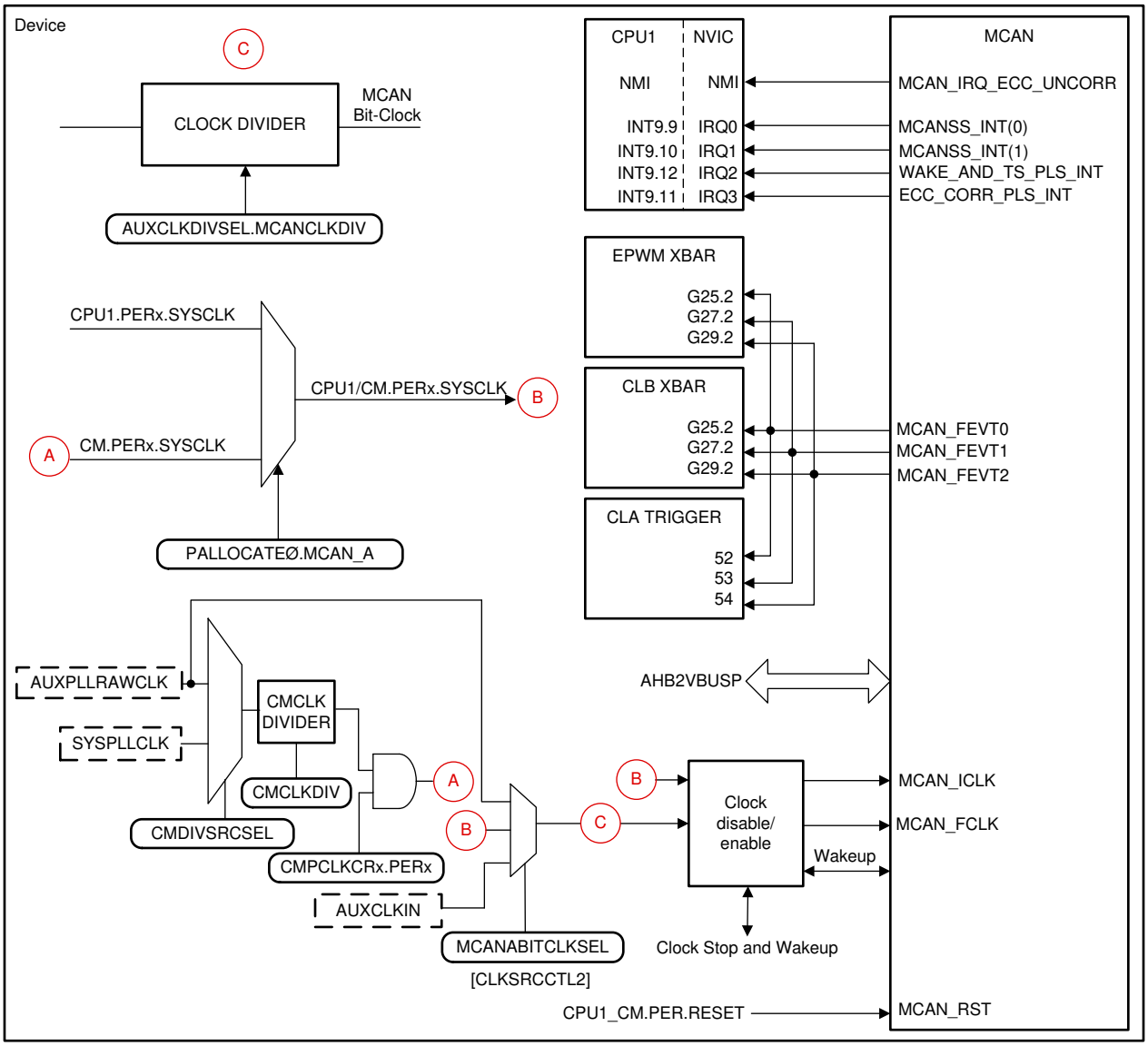


Figure 45-3. MCAN Integration

Table 45-2 and Table 45-3 summarize the integration of the MCAN module in the device.

Table 45-2. MCAN Clocks and Resets

Destination Signal Name	Source Signal Name	Description
Clocks		
MCAN_ICLK	CPU1/CM.PERx.SYSCLK	Interface clock for the MCAN module
MCAN_FCLK	MCANxBIT Clock	Bit timing clock for MCAN
Resets		
MCAN_RST	CPU1_CM.PER.RESET	Asynchronous reset signal to the MCAN module

Table 45-3. MCAN Hardware Requests

Interrupt Requests				
		Default Mapping		
Source Signal Name	NVIC Input	PIE	CM	Description
MCANSS_INT0	IRQ0	INT9.9	Configurable	MCAN interrupt 0
MCANSS_INT1	IRQ1	INT9.10	Configurable	MCAN interrupt 1
WAKE_AND_TS_PLS_INT	IRQ2	INT9.12	Configurable	MCAN timestamp and wakeup interrupt
ECC_CORR_PLS_INT	IRQ3	INT9.11	Configurable	MCAN ECC interrupt
MCAN_IRQ_ECC_UNCORR	NMI	NMI	-2 Priority	MCAN ECC uncorrectable interrupt
Filter Event Connections				
Source Signal Name	**Trigger Input**		**Default Mapping**	**Description**
MCAN_FEVT0	EPWM XBAR CLB XBAR CLA Trigger		G25.2 G25.2 52	MCAN RX Filter Event 1
MCAN_FEVT1	EPWM XBAR CLB XBAR CLA Trigger		G27.2 G27.2 53	MCAN RX Filter Event 2
MCAN_FEVT2	EPWM XBAR CLB XBAR CLA Trigger		G29.2 G29.2 54	MCAN RX Filter Event 3

Note

For more information about the CLA_triggers, see the *Control Law Accelerator (CLA)* chapter.

For more information about the CLB XBAR, see the *Configurable Logic Block (CLB)* chapter.

For more information about the ePWM XBAR module, see the *Enhanced Pulse Width Modulator (ePWM)* chapter.

45.5 MCAN Functional Description

The MCAN module performs CAN protocol communication according to ISO 11898-1:2015. The data bit rate can be programmed to values up to 5 Mbps. Additional transceiver hardware is required for the connection to the physical layer (CAN bus).

For communication on a CAN network, individual message frames can be configured. The message frames and identifier masks are stored in the Message RAM.

All functions concerning the handling of messages are implemented in the Message Handler.

The register set of the MCAN module can be accessed directly via the module interface. These registers are used to control and configure the CAN core and the Message Handler, and to access the Message RAM.

Figure 45-4 shows the MCAN module block diagram, followed by the description of the MCAN module blocks.

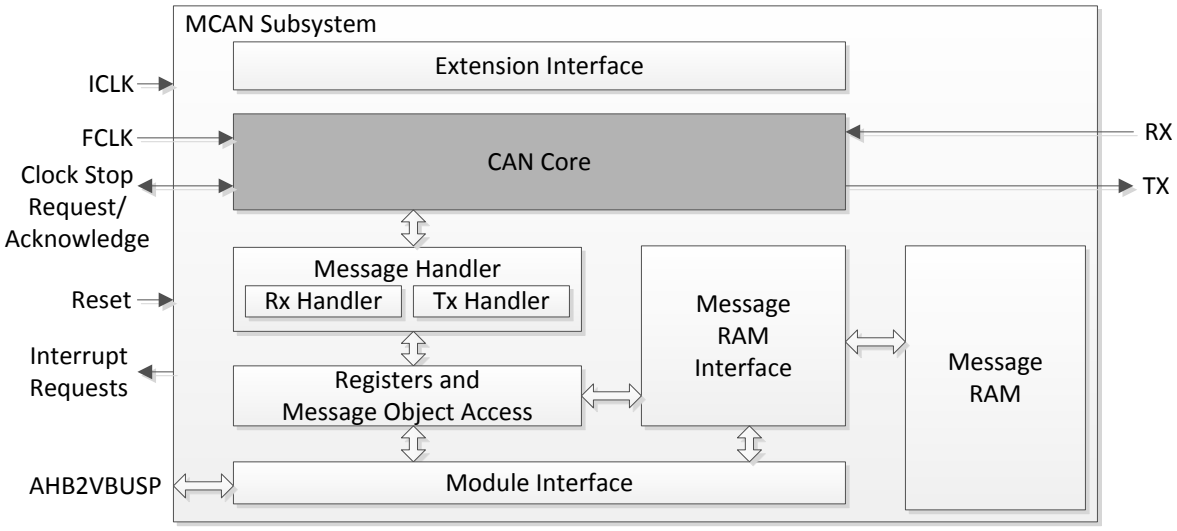


Figure 45-4. MCAN Block Diagram

- **CAN Core**: The CAN core consists of the CAN protocol controller and the Rx/Tx shift register. The CAN handles all ISO 11898-1:2015 protocol functions and supports 11-bit and 29-bit identifiers.
- **Message Handler**: the Message Handler (Rx Handler and Tx Handler) is a state machine that controls the data transfer between the single-ported Message RAM and the CAN core's Rx/Tx shift register. The Message Handler also handles the acceptance filtering and Interrupt generation as programmed in the control registers.
- **Message RAM**: the main purpose of the Message RAM is to store Rx/Tx messages, Tx Event elements, and Message ID Filter elements (for more information, see Section 45.5.16).
- **Message RAM Interface**: enables a connection between the Message RAM and the other blocks in the MCAN module.
- **Registers and Message Object Access**: Data consistency is provided by indirect accesses to the message objects. During normal operation, all software and DMA accesses to the Message RAM are done through interface registers. The interface registers have the same word-length as the Message RAM.
- **Module Interface**: The MCAN module registers are accessed by the user's software through a 32-bit peripheral bus interface.
- **Clocking**: Two clocks are provided to the MCAN module: the peripheral synchronous clock (interface clock - MCAN_ICLK) and the peripheral asynchronous clock (functional clock - MCAN_FCLK).
- **Extension Interface**: All selected internal status and control signals are routed to this interface (except for the indication signals of configuration change enable bit (MCAN_CCCR.CCE) and Interrupt Register bits (MCAN_IR).

45.5.1 Module Clocking Requirements

Two clocks are provided to the MCAN module:

- Host Clock : peripheral synchronous clock (MCAN_ICLK) as the general module clock source, and
- CAN Clock: peripheral asynchronous clock (MCAN_FCLK) provided to the CAN core for generating the CAN bit timing.

Within the MCAN module, there is a synchronization mechanism implemented to make sure there is safe data transfer between the two clock domains. There is synchronization between the signals from the Host clock domain to the CAN clock domain and conversely, and between the reset signal (MCAN_RST) to the Host clock domain and to the CAN clock domain.

Note

MCAN_ICLK must always be higher or equal to MCAN_FCLK, to achieve a stable functionality of the MCAN module: $f_{ICLK} \geq f_{FCLK}$

The CAN-FD supports higher speeds of operation and as such has more stringent timing requirements than the classic CAN. For performance, TI recommends using the lowest N-divider value that maintains a working PLL REF_CLK for the system. Lower N-divider values increase the loop bandwidth of the PLL, which in turn improves timing margins for CAN-FD.

45.5.2 Interrupt Requests

The MCAN module generates interrupt requests and is configured using the Host CPU. The Suspend mode prevents the interrupt requests from propagating to the Host CPU. The MCAN core has two interrupt lines and 30 internal interrupt sources. Each source can be configured to drive one of the two interrupt lines. The interrupts are level high interrupts. The MCAN core provides two interrupt requests (MCANSS_INT0 and MCANSS_INT1).

For more information, see the following registers:

- Interrupt Register (MCAN_IR)
- Interrupt Enable (MCAN_IE)
- Interrupt Line Select (MCAN_ILS)
- Interrupt Line Enable (MCAN_ILE)

The MCAN module supports External Timestamp Counter. The External Timestamp Counter produces an interrupt when the count rolls over (see Section 45.5.10.1).

For more information, see the following registers:

- Interrupt Clear Shadow Register (MCANSS_ICS)
- Interrupt Raw Status Register (MCANSS_IRS)
- Interrupt Enable Clear Shadow Register (MCANSS_IECS)
- Interrupt Enable Register (MCANSS_IE)
- Interrupt Enable Status Register (MCANSS_IES)
- End Of Interrupt Register (MCANSS_EOI)
- External Timestamp Prescaler Register (MCANSS_EXT_TS_PRESCALER)
- External Timestamp Unserviced Interrupts Counter Register (MCANSS_EXT_TS_UNSERVICED_INTR_CNTR)

To clear IRQ_INT0, IRQ_INT1, and TS_WAKE interrupts, write to the EOI bit field for the corresponding interrupt number that is described in the MCANSS_EOI register. When the MCAN is used by the CPU, in addition to clearing the interrupt sources, clear the interrupt by writing 1 to PIEACK register in the bit position for group 9 (refer to the *PIE Channel Mapping* section of the *System Control and Interrupts* chapter) for successive MCAN interrupts to be recognized.

The MCAN module is capable of issuing an ECC interrupt. After clearing the ECC interrupt source, the application software must also write a 1 to the EOI registers (MCANERR_SEC_EOI.EOI_WR/ MCANERR_DED_EOI.EOI_WR). For more information, see Section 45.5.12.2.

45.5.3 Operating Modes

The operating modes are discussed in the following sections.

45.5.3.1 Software Initialization

A software initialization begins when the MCAN_CCCR.INIT bit is set to 1. This is done either by software or by a hardware reset, when an uncorrected bit error is detected in the Message RAM, or by going to a Bus_Off state. While the MCAN_CCCR.INIT bit is set, the message transfer is stopped and the status of the output TX pin is recessive (high). The counters of the Error Management Logic (EML) are unchanged. Setting the MCAN_CCCR.INIT bit does not change any configuration register. Resetting the MCAN_CCCR.INIT bit finishes the software initialization. After waiting for the occurrence of a sequence of 11 consecutive recessive bits (indication for Bus_Idle state) the message transfer starts.

Access to the MCAN configuration registers is only enabled when both MCAN_CCCR.INIT and MCAN_CCCR.CCE bits are set (write protection).

The MCAN_CCCR.CCE bit can only be set/reset while the MCAN_CCCR.INIT = 1. The MCAN_CCCR.CCE bit is automatically reset when the MCAN_CCCR.INIT bit is reset.

The following registers are reset when the MCAN_CCCR.CCE bit is set:

- MCAN_HPMS - High Priority Message Status
- MCAN_RXF0S - Rx FIFO 0 Status
- MCAN_RFX1S - Rx FIFO 1 Status
- MCAN_TXFQS - Tx FIFO/Queue Status
- MCAN_TXBRP - Tx Buffer Request Pending
- MCAN_TXBTO - Tx Buffer Transmission Occurred
- MCAN_TXBCF - Tx Buffer Cancellation Finished
- MCAN_TXEFS - Tx Event FIFO Status

The Timeout Counter value MCAN_TOCV.TOC field is preset to the value configured by the MCAN_TOCC.TOP field when the MCAN_CCCR.CCE bit is set.

In addition, the Tx Handler and Rx Handler are held in idle state while MCAN_CCCR.CCE = 1.

The following registers are only writable while MCAN_CCCR.CCE = 0

- MCAN_TXBAR - Tx Buffer Add Request
- MCAN_TXBCR - Tx Buffer Cancellation Request

MCAN_CCCR.TEST and MCAN_CCCR.MON bits can only be set by the Host CPU while MCAN_CCCR.INIT = 1 and MCAN_CCCR.CCE = 1. Both bits are reset at any time. The MCAN_CCCR.DAR bit can only be set/reset while MCAN_CCCR.INIT = 1 and MCAN_CCCR.CCE = 1.

Table 45-4 shows the steps to configure the MCAN module.

Table 45-4. Steps to Configure MCAN Module

Step	Operation	Description	Pseudo Code
1	Initialize MCAN_CCCR	Set MCAN_CCCR.INIT bit and check that the bit has been set	INIT = 1; If INIT ≠ 1, wait until set
2	Unlock protected registers	Set MCAN_CCCR.CCE bit	CCE = 1;
3	Configure CAN mode	Set MCAN_CCCR.FDOE bit to CAN FD	FDOE = 1 for CAN FD FDOE = 0 for Classic CAN
4	Configure Bit Rate Switching	Set MCAN_CCCR.BRSE bit	BRSE = 1 for bit rate switching BRSE = 0 for no bit rate switching
5	Set nominal bit timing(1)	Set MCAN_NBTP register	
6	Lock protected registers	Clear MCAN_CCCR.CCE bit	CCE = 0;

Table 45-4. Steps to Configure MCAN Module (continued)

Step	Operation	Description	Pseudo Code
7	Return MCAN module to normal operation	Clear MCAN_CCCR.INIT bit and check that the bit has been cleared	INIT = 0; If INIT ≠ 0, wait until cleared

(1) See the MCAN_NBTP register on how to program CAN bit timing in the *MCAN_REGS Registers* section.

45.5.3.2 Normal Operation

Once the MCAN module is initialized and the MCAN_CCCR.INIT bit is reset to zero, the MCAN module synchronizes itself to the CAN bus and is ready for communication. After passing the acceptance filtering, received messages including Message Identifier (ID) and Data Length Code (DLC) are stored into a dedicated Rx Buffer or into Rx FIFO 0/Rx FIFO 1.

For messages to be transmitted, dedicated Tx buffers, and a Tx FIFO or a Tx queue can be initialized or updated.

Note

Automated transmission upon reception of remote frames is not supported.

45.5.3.3 CAN FD Operation

The CAN FD standard allows extended frames to be sent, up to 64 data bytes in a single frame at a higher bit rate for the data phase of a frame, up to 5 Mbps. The CAN FD standard introduces the ability to switch from one bit rate to another. Extended Data Length (EDL), as shown in Figure 45-5 and described in Table 45-5, sets a data length of up to 8 or 64 data bytes. Bit Rate Switching (BRS) indicates whether two bit rates (the data phase is transmitted at a different bit rate compared to the arbitration phase) are enabled.

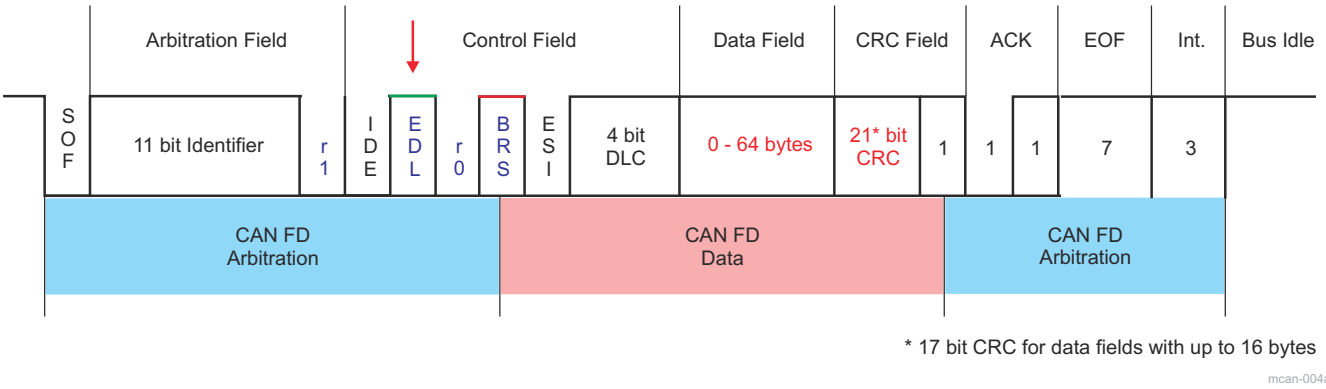


Figure 45-5. CAN FD Frame

Table 45-5. CAN FD Frame Description

Bit	Description
SOF	Start of Frame
IDE	Identifier extension (for 29 bit extended ID)
FDF	Flexible Data Format
BRS	Bit Rate Switching
ESI	Error Status Indicator
DLC	Data Length Code
CRC	Cyclic Redundancy check

There are two variants of CAN FD frame transmission:

- CAN FD frame transmission without bit rate switching
- CAN FD frame transmission where control field, data field, and CRC field are transmitted with a higher bit rate than the beginning and the end of the frame

In the CAN frames, FDF = recessive (logical 1) signifies a CAN FD frame, FDF = dominant (logical 0) signifies a Classic CAN frame. In a CAN FD frame, the two bits following FDF - res and BRS, decide whether the bit rate inside of this CAN FD frame is switched. A CAN FD bit rate switch is signified by res = dominant and BRS = recessive. Note that the coding of res = recessive is reserved for future expansion of the protocol. If the MCAN module receives a frame with FDF = recessive and res = recessive, the MCAN signals a Protocol Exception Event by setting the MCAN_PSR.PXE bit. When Protocol Exception Handling is enabled (MCAN_CCCR.PXHD = 0), this causes the operation state to change from Receiver (MCAN_PSR.ACT = 10) to Integrating (MCAN_PSR.ACT = 00) at the next sample point. In case Protocol Exception Handling is disabled (MCAN_CCCR.PXHD = 1), the MCAN treats a recessive bit as an error and responds with an error frame.

CAN FD operation is enabled by programming the MCAN_CCCR.FDOE bit. If MCAN_CCCR.FDOE = 1, transmission and reception of CAN FD frames is enabled. Transmission and reception of Classic CAN frames is always possible. Whether a CAN FD frame or a Classic CAN frame is transmitted can be configured using the FDF bit in the respective Tx Buffer element.

With MCAN_CCCR.FDOE = 0, received frames are interpreted as Classic CAN frames, which leads to the transmission of an error frame when receiving a CAN FD frame. When CAN FD operation is disabled, no CAN FD frames are transmitted even if the FDF bit of a Tx Buffer element is set. The MCAN_CCCR.FDOE and MCAN_CCCR.BRSE bits can only be changed while the MCAN_CCCR.INIT and MCAN_CCCR.CCE bits are both set. With MCAN_CCCR.FDOE = 0, the setting of bits FDF and BRS is ignored and frames are transmitted in Classic CAN format.

With MCAN_CCCR.FDOE = 1 and MCAN_CCCR.BRSE = 0, only FDF bit of a Tx Buffer element is evaluated. With MCAN_CCCR.FDOE = 1 and MCAN_CCCR.BRSE = 1, transmission of CAN FD frames with bit rate switching is enabled. All Tx Buffer elements with bits FDF and BRS set are transmitted in CAN FD format with bit rate switching.

A mode change during CAN operation is only recommended under the following conditions:

- The failure rate in the CAN FD data phase is significantly higher than in the CAN FD arbitration phase. In this case, disable the CAN FD bit rate switching option for transmissions.
- During system startup, all nodes are transmitting Classic CAN messages until verified that the nodes are able to communicate in CAN FD format. If this is true, all nodes switch to CAN FD operation.
- Wakeup messages in CAN Partial Networking have to be transmitted in Classic CAN format.
- End-of-line programming in case not all nodes are CAN FD capable. Non CAN FD nodes are held in Silent mode until programming has completed. Then all nodes switch back to Classic CAN communication.

The coding of the DLC in the CAN FD format differs from the standard CAN format. The DLC codes 0 to 8 have the same coding as in standard CAN (0 to 8 data bytes), the codes 9 to 15, which in standard CAN all code a data field of 8 bytes, are coded according to Table 45-6.

Table 45-6. DLC Coding in CAN FD

DLC	9	10	11	12	13	14	15
Number of Data Bytes	12	16	20	24	32	48	64

For CAN FD frames, the bit timing is switched inside the frame after the BRS (Bit Rate Switch) bit in case this bit is recessive. In the CAN FD arbitration phase, before the BRS bit, the nominal CAN bit timing (see Figure 45-6) is used as configured by the Nominal Bit Timing and Prescaler Register (MCAN_NBTP). In the following CAN FD data phase, the data phase bit timing is used as configured by the Data Bit Timing and Prescaler Register (MCAN_DBTP). The bit timing is switched back from the data phase timing at the CRC delimiter or when an error is detected, whichever occurs first.

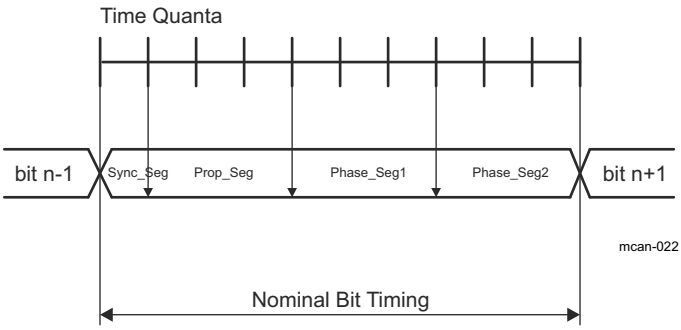


Figure 45-6. CAN Bit Timing

The maximum configurable data phase bit timing depends on the CAN clock frequency (MCAN_FCLK). Example: with MCAN_FCLK = 20 MHz and the shortest configurable bit time of 4 t_q (time quanta), the bit rate in the data phase is 5 Mbit/s.

In both data frame formats, CAN FD and CAN FD with bit rate switching, the value of the Error Status Indicator (ESI) bit depends on the transmitter error state (see MCAN_PSR.RESI bit) monitored at the start of the transmission. If the transmitter has an error passive flag, the ESI bit is transmitted recessive; else, the ESI bit is transmitted dominant.

45.5.4 Transmitter Delay Compensation

45.5.4.1 Description

When only one CAN FD node is transmitting and all other nodes are receivers, the length of the bus line has no impact. When transmitting using the TX pin, the MCAN module receives the transmitted data from the CAN transceiver using the RX pin. The received data is delayed. If the transmitter delay is greater than TSEG1 (time segment before sample point), a bit error is detected.

The MCAN module provides a delay compensation mechanism to compensate for the transmitter delay. The compensation mechanism enables transmission with higher bit rates during the CAN FD data phase independent of the delay of a specific CAN transceiver. Without transmitter delay compensation the bit rate in the data phase is limited by the transmitter delay.

The mechanism enables configurations where the data bit time is shorter than the transmitter delay (it is described in detail in ISO 11898-1:2015). The transmitter delay compensation is enabled by setting the MCAN_DBTP.TDC bit to 1.

The delayed transmit data is compared against the received data at the Secondary Sample Point (SSP) to check for bit errors during the data phase of transmitting nodes. If a bit error is detected, the transmitter reacts on this bit error at the next following regular sample point. During the arbitration phase, the delay compensation is always disabled.

The received bit is compared against the transmitted bit at the SSP. The SSP position is defined as the sum of the measured delay from the MCAN's transmit output TX pin through the transceiver to the receive input RX pin plus the transmitter delay compensation offset configured by the MCAN_TDCR.TDCO field (see Figure 45-7). The transmitter delay compensation offset is used to adjust the position of the SSP inside the received bit (example: half of the bit time in the data phase). The position of the SSP is rounded down to the next integer number of mtq.

The actual transmitter delay compensation value can be checked by reading the MCAN_PSR.TDCV field. This field is cleared when the MCAN_CCCR_INIT bit is set and is updated at each transmission of CAN FD frame while the MCAN_DBTP.TDC bit is set.

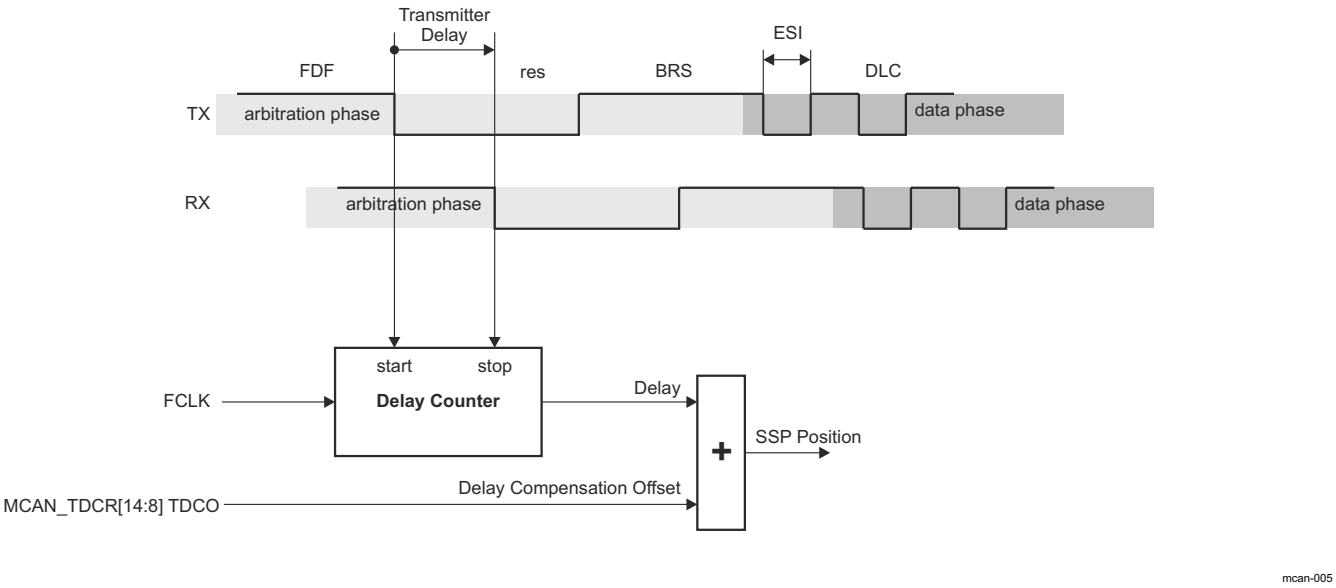


Figure 45-7. Transmitter Delay Measurement

45.5.4.2 Transmitter Delay Compensation Measurement

When transmitter delay compensation is enabled (by programming MCAN_DBTP.TDC = 1), the measurement is started within each transmitted CAN FD frame at the falling edge of FDF bit to bit r0. The measurement is stopped when this edge is seen at the receive input RX pin of the transmitter. The resolution of this measurement is one mtq (see Figure 45-7). The mtq (minimum time quantum) dimension is equal to the CAN clock period (MCAN_FCLK).

The use of a transmitter delay compensation filter window can be enabled by programming the MCAN_TDCR.TDCF field. This filter feature defines a minimum value for the SSP position to avoid the case in which a dominant glitch inside the received FDF bit ends the delay compensation measurement before the falling edge of the received res bit, resulting in an early taken SSP position. Dominant edges on the RX pin that result in an earlier SSP position are ignored for transmitter delay measurement. The measurement is stopped when the SSP position is at least MCAN_TDCR.TDCF field and the RX pin is low.

The following boundary conditions have to be considered:

- The sum of the measured delay from the TX pin to the RX pin and the configured transmitter delay compensation offset (MCAN_TDCR.TDCO field) has to be less than six bit times in the data phase.
- The sum of the measured delay from the TX pin to the RX pin and the configured transmitter delay compensation offset (MCAN_TDCR.TDCO) field has to be less or equal 127 mtq. In case this sum exceeds 127 mtq, the maximum value of 127 mtq is used for transmitter delay compensation.
- The data phase ends at the sample point of the CRC delimiter, that stops checking of receive bits at the SSPs.

45.5.5 Restricted Operation Mode

In restricted operation mode, the CAN node is able to receive data and remote frames and to give acknowledgment to valid frames, but the node does not send data frames, remote frames, active error frames, or overload frames. In case of an error condition or overload condition, the node does not send dominant bits; instead the node waits for the occurrence of bus idle condition to resynchronize itself to the CAN communication. The receive and transmit error counters (MCAN_ECR.REC and MCAN_ECR.TEC) are frozen while CAN error logging (MCAN_ECR.CEL) is active. The Host CPU can set the MCAN module into Restricted Operation Mode by setting the MCAN_CCCR.ASM bit. The bit can only be set by the Host CPU at any time when both MCAN_CCCR.CCE and MCAN_CCCR.INIT bits are set to 1.

The restricted operation mode is automatically entered when the Tx Handler is not able to read data from the Message RAM in time. To leave restricted operation mode, the Host CPU has to reset the MCAN_CCCR.ASM bit. This mode can be used in applications that adapt themselves to different CAN bit rates. In this case, the application tests different bit rates and leaves the restricted operation mode after the node has received a valid frame.

Note

The Restricted Operation Mode must not be combined with the Loop Back Mode.

45.5.6 Bus Monitoring Mode

Entering bus monitoring mode is done by setting the MCAN_CCCR.MON bit to 1. In this mode (see ISO 11898-1:2015, *Bus Monitoring* section), the MCAN module is able to receive valid data and remote frames, but cannot start a transmission. The MCAN module sends only recessive bits on the CAN bus. If the MCAN module is required to send a dominant bit (ACK bit, overload flag, active error flag), the bit is rerouted internally so that the MCAN module monitors this dominant bit, although the CAN bus can remain in recessive state. In bus monitoring mode, the MCAN_TXBRP register is held in reset state. The bus monitoring mode can be used to analyze the traffic on a CAN bus without affecting the bus by the transmission of dominant bits. Figure 45-8 shows the connection of the TX and RX signals to the MCAN module in bus monitoring mode.

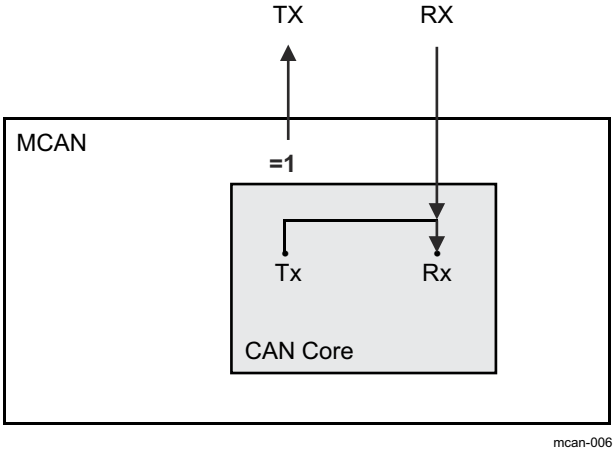


Figure 45-8. Connection of Signals in Bus Monitoring Mode

45.5.7 Disabled Automatic Retransmission (DAR) Mode

According to the CAN Specification (see ISO11898-1:2015, *Recovery Management* section), the MCAN module provides means for automatic retransmission of frames that have lost arbitration or that have been disturbed by errors during transmission. By default automatic retransmission is enabled (see the MCAN_CCCR.DAR bit).

45.5.7.1 Frame Transmission in DAR Mode

In DAR mode, the automatic retransmission of frames that have lost arbitration or that have been disturbed by errors during transmission is disabled. A Tx buffer's Tx Request Pending (MCAN_TXBRP[xx]) TRPx bit is reset after successful transmission, when a transmission has not yet been started at the point of cancellation, has been aborted due to lost arbitration, or when an error occurred during frame transmission.

Successful transmission:

- Corresponding Tx Buffer Transmission Occurred (MCAN_TXBTO[xx]) TOx bit is set
- Corresponding Tx Buffer Cancellation Finished (MCAN_TXBCF[xx]) CFx bit is not set

Successful transmission in spite of cancellation:

- Corresponding Tx Buffer Transmission Occurred (MCAN_TXBTO[xx]) TOx bit is set
- Corresponding Tx Buffer Cancellation Finished (MCAN_TXBCF[xx]) CFx bit is set

Arbitration lost or frame transmission disturbed:

- Corresponding Tx Buffer Transmission Occurred (MCAN_TXBTO[xx]) TOx bit is not set
- Corresponding Tx Buffer Cancellation Finished (MCAN_TXBCF[xx]) CFx bit is set

In the case of a successful frame transmission, and if storage of Tx events is enabled, a Tx Event FIFO element is written with Event Type ET = 10 (transmission in spite of cancellation).

45.5.8 Clock Stop Mode

Entering clock stop mode is controlled by the input clock stop request signal or MCAN_CCCR.CSR bit. As long as the clock stop request signal is active, the MCAN_CCCR.CSR bit is read as 1. When all pending transmission requests have completed, the MCAN module waits until bus idle state is detected. Then the MCAN module sets the MCAN_CCCR.INIT to 1 to prevent any further CAN transfers. The MCAN module acknowledges that the module is ready for power down by setting the output clock stop acknowledge signal to 1 and the MCAN_CCCR.CSA bit to 1. In this state, before the clocks are switched off, further register accesses can be made. A write access to the MCAN_CCCR.INIT bit has no effect. Now the module clock inputs MCAN_ICLK and MCAN_FCLK can be switched off.

To leave power-down mode, the application has to turn on the module clocks before resetting the input clock stop request signal respectively the MCAN_CCCR.CSR flag bit. The MCAN acknowledges this by resetting the output clock stop acknowledge signal respectively the MCAN_CCCR.CSA flag bit. Afterwards, the application can restart CAN communication by resetting the MCAN_CCCR.INIT bit.

Restoring the clocks from clock stop mode needs to be done according to how the clock stop was initiated.

The MCAN module supports two external clock stop modes:

- Immediate
- Graceful

In a graceful clock stop mode when the clock stop request is asserted, the MCAN core responds with a clock stop acknowledge when all pending Tx messages have been processed and an Idle line had been detected. The MCAN_CCCR.INIT bit is set, the MCAN core goes and stays Idle.

The automatic wakeup feature is enabled by setting the MCANSS_CTRL.AUTOWAKEUP and MCANSS_CTRL.WAKEUPREQEN bits to 1 (for more information, see Section 45.5.8.2). When an external clock stop request is removed and no suspend request is active, a read-modify-write to the MCAN_CCCR.INIT bit is performed to clear the bit.

45.5.8.1 Suspend Mode

The MCAN module supports two suspend modes:

- Immediate
- Graceful

In a graceful suspend mode (see the MCANSS_CTRL.DBGSUSP_FREE bit) when the suspend request is asserted, a clock stop request to the MCAN core is performed. The MCAN core responds with a clock stop acknowledge when all pending Tx messages have been processed and an Idle line had been detected. At that point, the MCAN_CCCR.INIT bit is set and the MCAN core stays Idle. The suspend state can be verified by reading the MCAN_CCCR.INIT bit.

The automatic wakeup feature is enabled by setting the MCANSS_CTRL.AUTOWAKEUP and MCANSS_CTRL.WAKEUPREQEN bits to 1 (for more information, see Section 45.5.8.2). When suspend request is removed, if no external clock stop request is active, a read-modify-write to the MCAN_CCCR.INIT bit is performed to clear the bit.

During suspend mode the auto-clear feature is disabled. The following register fields have an auto-clear feature:

- MCAN_ECR.CEL
- MCAN_PSR.LEC
- MCAN_PSR.DLEC
- MCAN_PSR.RESI
- MCAN_PSR.RBRS
- MCAN_PSR.RFDF
- MCAN_PSR.PXE

45.5.8.2 Wakeup Request

Issuing a clock stop request puts the MCAN module into power-down mode (Sleep Mode). During transition from IDLE to ACTIVE, if the MCANSS_CTRL.AUTOWAKEUP and MCANSS_CTRL.WAKEUPREQEN bits are enabled, after the MCAN Core responds to the removal of the clock stop request with removing the clock stop acknowledge, a read-modify-write is issued to clear the MCAN_CCCR.INIT bit and the MCAN core resumes operation.

If the MCANSS_CTRL.WAKEUPREQEN bit is set, the MCAN module provides a wakeup request on the following wakeup event:

- The receive RX pin is dominant (logical 0)

The wakeup request is deasserted when any of the following conditions occur:

- Clock stop request is removed and clock stop acknowledge is deasserted
- A reset is applied to the MCAN module

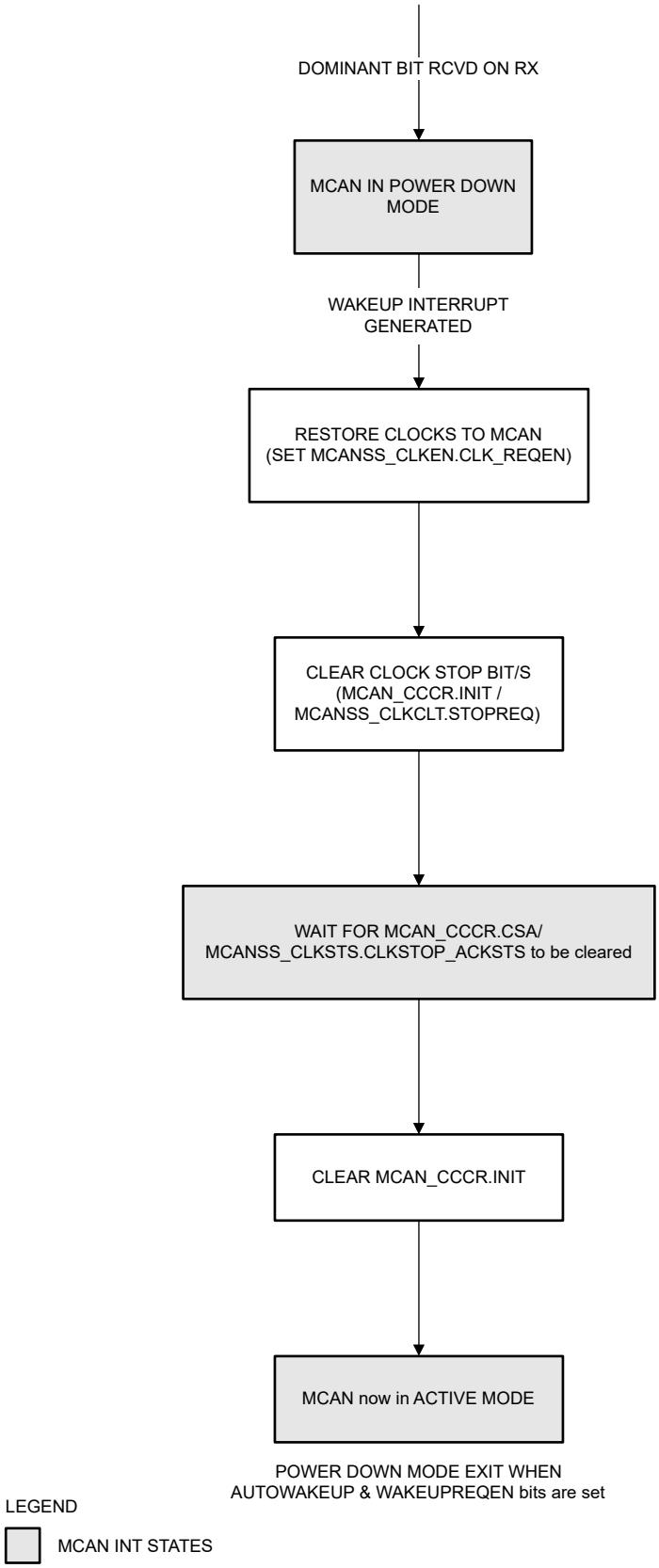


Figure 45-9. Auto Wakeup Enabled Exit from Power Down

45.5.9 Test Modes

The MCAN_TEST register write access is enabled by setting the test mode enable MCAN_CCCR.TEST bit to 1. The MCAN_TEST register allows the configuration of the test modes and test functions.

The transmit (TX) pin has four different output functions which can be selected by programming the MCAN_TEST.TX field. The default function is the serial data output. The pin can also be driven with a constant dominant or recessive value. It is also possible to drive the sample-point signal to monitor the bit-timing.

The actual value of the receive (RX) pin can be monitored from MCAN_TEST.RX bit. Both functions can be used to check the physical layer. Due to the synchronization mechanism between the CAN clock (MCANx_FCLK) and Host clock (MCANx_ICLK) domain, there can be a delay of several Host clock periods between writing to the MCAN_TEST.TX field until the new configuration is visible at the output TX pin. This applies also when reading input RX pin by way of the MCAN_TEST.RX bit.

Note

Test modes can be used for self-test only. The software control for TX pin interferes with all CAN protocol functions. It is not recommended to use test modes for an application.

45.5.9.1 External Loop Back Mode

The MCAN module can be set into external loop back mode by programming MCAN_TEST.LBCK to 1. In loop back mode, the MCAN treats their own transmitted messages as received messages and stores them (if the messages pass acceptance filtering) into an Rx Buffer or an Rx FIFO. Figure 45-10 shows the connection of the TX and RX pins to the MCAN module in external loop back mode.

This mode is provided for hardware self-test. To be independent from external stimulation, the MCAN module ignores acknowledge errors (recessive bit sampled in the acknowledge slot of a data/remote frame) in loop back mode. In this mode, the MCAN module performs an internal feedback from the Tx output to the Rx input. The actual value of the RX input pin is disregarded by the MCAN module. The transmitted messages are monitored at the TX pin.

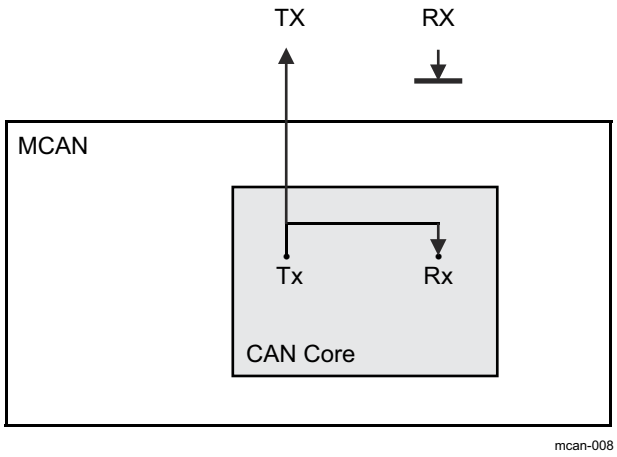


Figure 45-10. External Loop Back Mode

45.5.9.2 Internal Loop Back Mode

The MCAN module can be set into internal loop back mode by programming MCAN_TEST.LBCK and MCAN_CCCR.MON bits to 1. The internal loop back mode is used for a Hot Self-test. The Hot Self-test allows the MCAN module to be tested without affecting a running CAN system connected to the TX and RX pins. In this mode, the RX pin is disconnected from the MCAN module and the TX pin is held recessive. Figure 45-11 shows the connection of the TX and RX pins to the MCAN module in internal loop back mode.

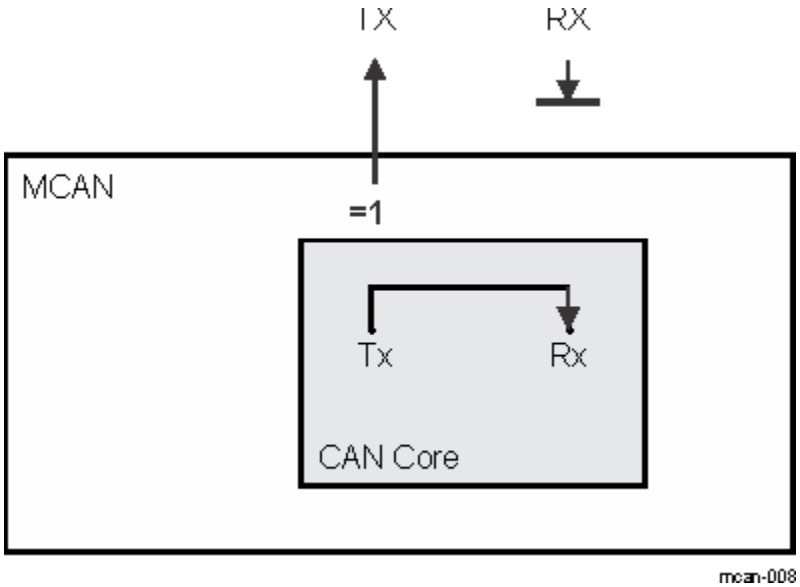


Figure 45-11. Internal Loop Back Mode

45.5.10 Timestamp Generation

The MCAN module has integrated a 16-bit wrap-around counter for timestamp generation. The timestamp counter prescaler MCAN_TSCC.TCP field can be configured to clock the counter in multiples of CAN bit times (1-16). The counter is readable by way of the MCAN_TSCV.TSC field. A write access to the MCAN_TSCV register resets the counter to zero. When the timestamp counter wraps around the interrupt MCAN_IR.TSW flag is set. On start of a frame reception/transmission the counter value is captured and stored into the timestamp section of an Rx Buffer/Rx FIFO (RXTS[15:0]) or Tx Event FIFO (TXTS[15:0]) element. For more information, see Section 45.5.16.

45.5.10.1 External Timestamp Counter

For CAN FD operation mode, the MCAN core requires an external timestamp counter (see Figure 45-12). An externally generated 16-bit vector can substitute the integrated 16-bit CAN bit time counter (internal timestamp counter) for receive and transmit timestamp generation. An external 16-bit timestamp counter can be used by programming the MCAN_TSCC.TSS field.

The external timestamp counter uses the interface clock (MCANx_ICLK) as a reference clock. The MCAN core accepts a 16-bit timestamp. A 24-bit prescaler provides a programmable resolution for the timestamp (see MCANSS_EXT_TS_PRESCALER.PRESCALER bit field). The external timestamp counter can be enabled or disabled through the MCANSS_CTRL.EXT_TS_CNTR_EN bit. When disabled, the counter is reset back to zero. While enabled, the counter keeps incrementing. When the timestamp rolls over, the MCAN_IRQ_TS interrupt is generated.

When the timestamp rolls over, the MCANSS_IRS register is set. The MCANSS_IE register can be affected by writing to the MCANSS_IESS register to set or to the MCANSS_IECS register to clear. The MCANSS_IESS register is a shadow register mapped to the same address as the MCANSS_IE register. The level interrupt is a reflection of both MCANSS_IRS and MCANSS_IE being set. The MCANSS_IES register reflects the level interrupt. When a rollover event occurs, the interrupt counter is incremented. Writing to the MCANSS_ICS register to clear the MCANSS_IRS register also decrements the interrupt counter. Writing to the MCANSS_EOI register issues another pulse, if the interrupt counter is not zero.

The rollover event can be artificially simulated by software through writing to the Interrupt Set Shadow register (MCANSS_ISS). The MCANSS_ISS register is a shadow register mapped to the same address as the MCANSS_IRS register.

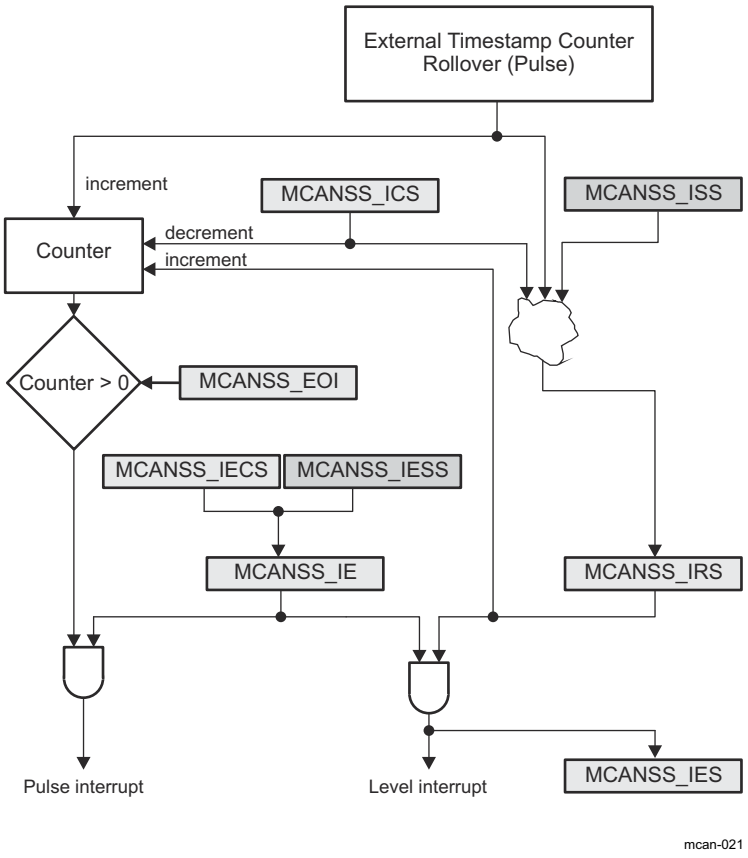


Figure 45-12. External Timestamp Counter Interrupt

45.5.11 Timeout Counter

The MCAN module has an integrated 16-bit timeout counter. The timeout counter is used to signal timeout conditions for the Rx FIFO 0, Rx FIFO 1, and Tx Event FIFO Message RAM elements. The timeout counter is configured using the MCAN_TOCC register and is enabled using the MCAN_TOCC.ETOC bit. The timeout counter operates as down-counter and uses the same prescaler programmed by the MCAN_TSCC.TCP field as the timestamp counter. The actual counter value can be monitored from the MCAN_TOCV.TOC field. The timeout counter can be started only when MCAN_CCCR.INIT = 0 and stopped when MCAN_CCCR.INIT = 1 (example: when the MCAN enters Bus_Off state). The operation mode is selected by the MCAN_TOCC.TOS field. When continuous mode is selected, the counter starts when MCAN_CCCR.INIT = 0, a write to the MCAN_TOCV register presets the counter to the value configured by the MCAN_TOCC.TOP field and continues down-counting.

In case the timeout counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the value configured by the MCAN_TOCC.TOP field. Down-counting is started when the first FIFO element is stored. Writing to the MCAN_TOCV register has no effect. When the counter reaches zero, the interrupt MCAN_IR.TOO flag is set.

In continuous mode, the counter is immediately restarted at the value configured by the MCAN_TOCC.TOP field.

Note

The clock signal for the timeout counter is derived from the CAN core sample point signal. Therefore, the point in time where the timeout counter is decremented can vary due to the synchronization/re-synchronization mechanism of the CAN core. If the baud rate switch feature in CAN FD is used, the timeout counter is clocked differently in arbitration and data field.

45.5.12 Safety

The Message Memory is wrapped in an ECC wrapper providing SECDED parity functionality. The ECC wrapper is controlled by an ECC aggregator.

45.5.12.1 ECC Wrapper

The ECC wrapper provides single error correction (SEC) and double error detection (DED) parity to the message memory content. The ECC wrapper has side band signals for error notification. The ECC Wrapper implements an error injection test mode.

The error correction is done using a lazy write back. When an error is detected, the error is noted in a FIFO queue that waits for an access gap to write the data back and refresh the memory. If a transaction writes new data to the compromised entry before the lazy write back completes, the write back is discarded.

45.5.12.2 ECC Aggregator

This section describes the functional details of the ECC aggregator module.

45.5.12.2.1 ECC Aggregator Overview

The ECC aggregator module supports the following general features:

- Provides a mechanism to control and monitor the ECC RAM in the MCAN module.
- Provides software access to all the ECC related registers.
- Supports software readable status of ECC single/double-bit errors and associated info such as RAM address and data bits that are in error.
- Aggregates level pending status from the ECC RAM into a single interrupt to the Host CPU.

The following feature is not supported:

- Statistics such as tracking the number of single and double-bit errors. If needed, these operations can be handled by software.

45.5.12.2.2 ECC Aggregator Registers

There are three groups of registers in the ECC aggregator module:

- **Global registers:** Aggregator Revision Register (MCANERR_REV), ECC Vector Register (MCANERR_VECTOR), Misc Status Register (MCANERR_STAT), ECC Control Register (MCANERR_CTRL), and ECC Wrapper Revision Register (MCANERR_WRAP_REV).
- **Control and status registers:** ECC Error Control Registers (MCANERR_ERR_CTRL1 and MCANERR_ERR_CTRL2) and ECC Error Status Registers (MCANERR_ERR_STAT1, MCANERR_ERR_STAT2, and MCANERR_ERR_STAT3).
- **Interrupt registers:** interrupt status, interrupt enable set, interrupt enable clear, and EOI (End Of Interrupt) registers that are part of a standard interrupt module. For more information, see the following registers:
  - MCANERR_SEC_EOI
  - MCANERR_SEC_STATUS
  - MCANERR_SEC_ENABLE_SET
  - MCANERR_SEC_ENABLE_CLR
  - MCANERR_DED_EOI
  - MCANERR_DED_STATUS
  - MCANERR_DED_ENABLE_SET
  - MCANERR_DED_ENABLE_CLR

45.5.12.3 Reads to ECC Control and Status Registers

The reads to the ECC control and status registers are triggered by writing a 'read message' to the ECC Vector Register as:

- Software writes value (the ECC RAM ID) to the MCANERR_VECTOR.ECC_VECTOR field to select the ECC RAM for control or status.
- Software writes 1 to the MCANERR_VECTOR.RD_SVBUS bit to trigger a read.
- Software writes read address to the MCANERR_VECTOR.RD_SVBUS_ADDRESS field.
- Software then polls the MCANERR_VECTOR.RD_SVBUS_DONE bit to check if the bit is 1. This bit indicates that the read operation has completed.
- Software reads the data from the ECC control or status register. The following clock cycle (MCAN_ICLK) returns the read data.

45.5.12.4 ECC Interrupts

The ECC aggregator module aggregates the level pending status from the ECC RAM into a single EOI-handshake based interrupt to the Host CPU. Software is expected to follow the sequence described:

- Software enables the interrupts for the ECC RAM by writing to the MCANERR_SEC_ENABLE_SET/ MCANERR_DED_ENABLE_SET register.
- Software writes the ECC RAM ID in the MCANERR_VECTOR.ECC_VECTOR.
- Software writes the MCANERR_VECTOR.RD_SVBUS bit to trigger the read.
- Software writes the MCANERR_ERR_STAT1 register address to the MCANERR_VECTOR.RD_SVBUS_ADDRESS field. Software needs to load the 'read message' in the rMCANERR_VECTOR register again, if the software needs to read the MCANERR_ERR_STAT2 register.
- Software polls the MCANERR_VECTOR.RD_SVBUS_DONE bit. When this bit is set, a read of the MCANERR_ERR_STAT1/MCANERR_ERR_STAT2 register is performed.
- After the interrupt has been serviced, software clears the interrupt status by writing to the MCANERR_ERR_STAT1.CLR_ECC_SEC or MCANERR_ERR_STAT1.CLR_ECC_DED bit depending on the type of the ECC error.
- Software polls the MCANERR_ERR_STAT1 register to verify that the status bit has been cleared.
- Software writes to the MCANERR_SEC_EOI/MCANERR_DED_EOI register to clear the interrupt.
- After clearing the ECC interrupt source, the application software must also write 1 to the MCANERR_SEC_EOI.EOI_WR /MCANERR_DED_EOI.EOI_WR bits.

45.5.13 Rx Handling

The Rx Handler controls the following operations:

- Acceptance filtering
- The transfer of received messages to the Rx buffers or to one of the two Rx FIFOs (Rx FIFO 0 or Rx FIFO 1)
- Rx FIFO Put and Get Index operations

45.5.13.1 Acceptance Filtering

The MCAN module employs two sets of acceptance filters - one set for standard and one set for extended identifiers. These filters can be assigned to an Rx Buffer or to one of the two Rx FIFOs.

The main features of the filter elements are:

- Each filter element can be configured as:
  - Range filter (from - to)
  - Filter for specific IDs (for one or two dedicated IDs)
  - Classic bit mask filter
- Each filter element can be enabled/disabled individually
- Each filter element can be configured for acceptance or rejection filtering
- Filters are checked sequentially and execution (acceptance filtering procedure) stops at the first matching filter element or when the end of the filter list is reached

Related configuration registers are:

- Global Filter Configuration (MCAN_GFC) register
- Standard ID Filter Configuration (MCAN_SIDFC) register
- Extended ID Filter Configuration (MCAN_XIDFC) register
- Extended ID AND Mask (MCAN_XIDAM) register

Depending on the configuration of the filter element (see SFEC/EFEC in Section 45.5.16) if filter matches, one of the following actions is performed:

- Received frame is stored in FIFO 0 or FIFO 1
- Received frame is stored in Rx Buffer
- Received frame is stored in Rx Buffer and generation of pulse at filter event pin is performed. This is high level single MCAN_ICLK pulse.
- Received frame is rejected
- Set High Priority Message interrupt flag MCAN_IR.HPM
- Set High Priority Message interrupt flag MCAN_IR.HPM and store received frame in FIFO 0 or FIFO 1

Acceptance filtering starts when complete Message ID is received. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If a filter element matches - the Rx Handler starts writing the received message data in portions of 32-bit to the matching Rx Buffer or Rx FIFO. If an error condition occurs (for example: CRC error), this message is rejected with the following impact on the affected Rx Buffer or Rx FIFO:

- Rx Buffer: New Data flag (MCAN_NDAT1/MCAN_NDAT2) of matching Rx Buffer is not set, but Rx Buffer (partly) overwritten with received data (for error type, see MCAN_PSR.LEC and MCAN_PSR.DLEC fields,respectively).
- Rx FIFO: Put index of matching Rx FIFO is not updated, but related Rx FIFO element (partly) overwritten with received data (for error type, see MCAN_PSR.LEC and MCAN_PSR.DLEC fields, respectively). If matching Rx FIFO is configured to operate in overwrite mode, the boundary conditions described in Section 45.5.13.2.2 have to be considered.

Note

When an accepted message is written to one of the two Rx FIFOs, or into an Rx Buffer, the unmodified received identifier is stored independent of the filters used. The result of the acceptance filter process is strongly depending on the sequence of configured filter elements.

45.5.13.1.1 Range Filter

Each filter element can be configured to operate as Range Filter (Standard Filter Type SFT = 00/Extended Filter Type EFT = 00). The filter matches for all received message frames with IDs in the range from SFID1 to SFID2 (SFID2 ≥ SFID1) respectively in the range from EFID1 to EFID2 (EFID2 ≥ EFID1). For more information see Section 45.5.16.5 and Section 45.5.16.6.

There are two options for range filtering of extended frames:

- Extended Filter Type EFT = 00: The Extended ID AND Mask (MCAN_XIDAM) is used for Range Filtering. The Message ID of received frames is ANDed with the Extended ID AND Mask (MCAN_XIDAM) before the range filter is applied.
- Extended Filter Type EFT = 11: The Extended ID AND Mask (MCAN_XIDAM) is not used for Range Filtering.

45.5.13.1.2 Filter for Specific IDs

Each filter element can be configured to filter one or two dedicated Message IDs (Standard Filter Type SFT =01/ Extended Filter Type EFT =01). To filter only one specific Message ID, the filter element has to be configured with SFID1 = SFID2 respectively EFID1 = EFID2. For more information, see Section 45.5.16.5 and Section 45.5.16.6.

45.5.13.1.3 Classic Bit Mask Filter

Classic bit mask filtering can filter groups of Message IDs (Standard Filter Type SFT =10/Extended Filter Type EFT =10). This is done by masking single bits of a received Message ID. In this case SFID1/EFID1 element is used as Message ID filter, while the SFID2/EFID2 element is used as filter mask.

A 0 bit at the filter mask (SFID2/EFID2) masks out the corresponding bit position of the configured Message ID filter (SFID1/EFID1) and the value of the received Message ID at that bit position is not relevant for acceptance filtering. Only those bits of the received Message ID where the corresponding mask bits are 1 are relevant for acceptance filtering.

There are two interesting cases:

- All mask bits are 1: a match occurs only when the received Message ID and the configured Message ID filter are identical.
- All mask bits are 0: all Message IDs match.

45.5.13.1.4 Standard Message ID Filtering

Figure 45-13 shows the standard Message ID (11-bit ID) filtering flow. Section 45.5.16.5 describes the standard Message ID filter element.

The Remote Transmission Request (RTR) and Extended Identifier (XTD) bits of the received frames are compared against the list of configured filter elements. This is controlled by the following registers:

- Global Filter Configuration (MCAN_GFC) register
- Standard ID Filter Configuration (MCAN_SIDFC) register

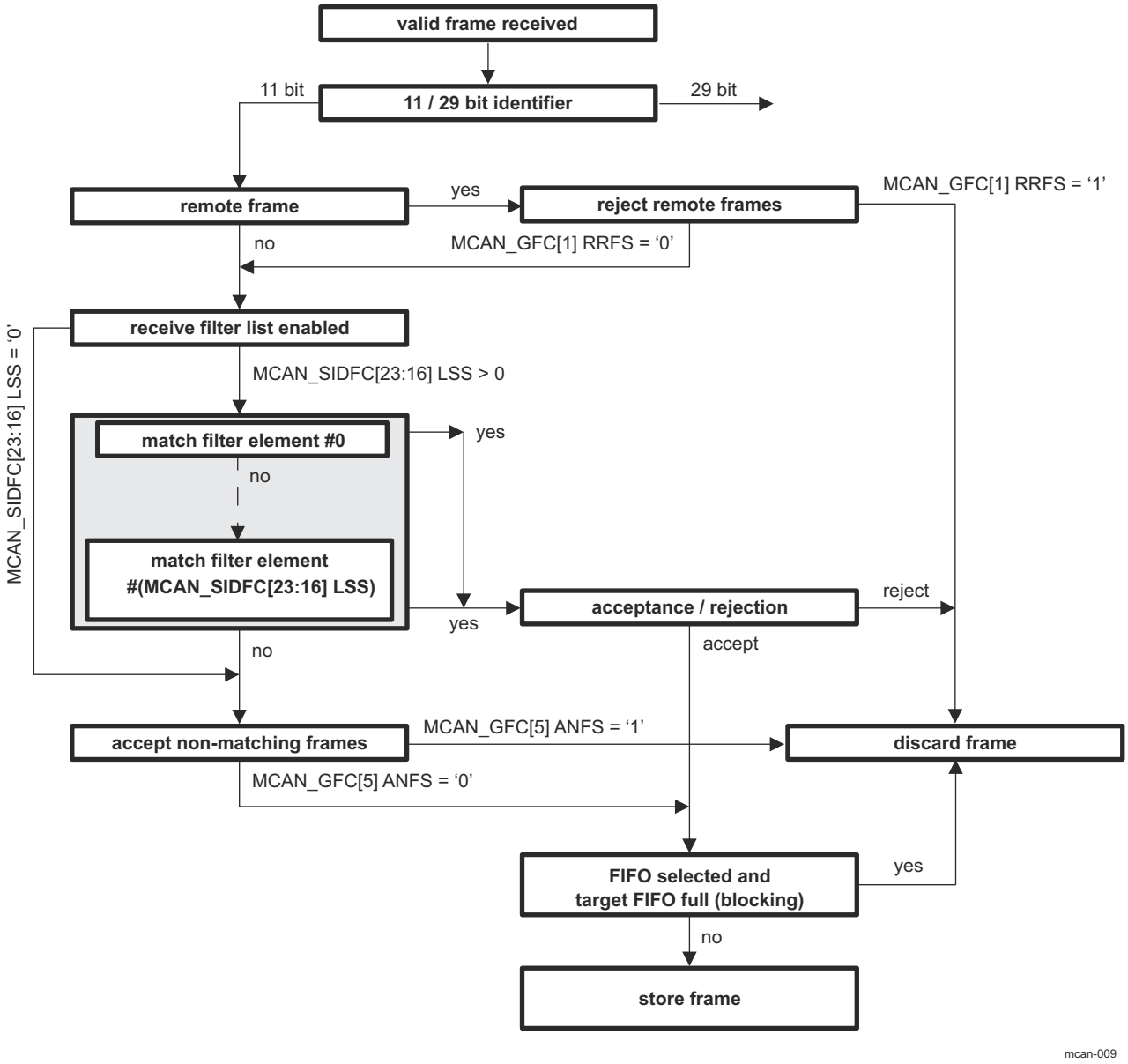


Figure 45-13. Standard Message ID Filter Path

45.5.13.1.5 Extended Message ID Filtering

Figure 45-14 shows the extended Message ID (29-bit ID) filtering flow. Section 45.5.16.6 describes the extended Message ID filter element.

The Remote Transmission Request (RTR) and Extended Identifier (XTD) bits of the received frames are compared against the list of configured filter elements. This is controlled by the following registers:

- Global Filter Configuration (MCAN_GFC) register
- Extended ID Filter Configuration (MCAN_XIDFC) register

Note that before the filter list is executed, the received identifier is ANDed with the Extended ID AND Mask (MCAN_XIDAM).

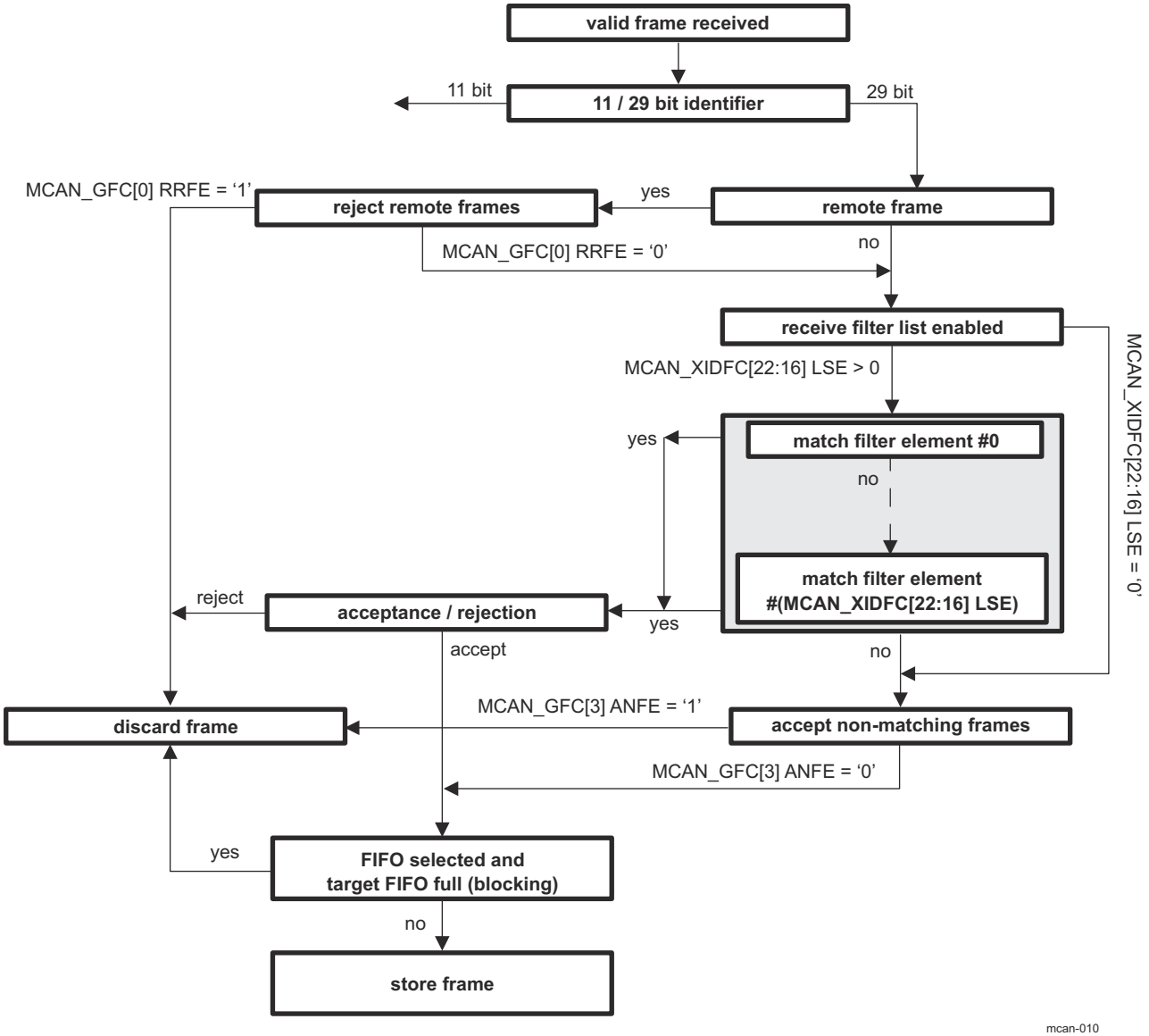


Figure 45-14. Extended Message ID Filter Path

45.5.13.2 Rx FIFOs

The configuration of the Rx FIFOs (Rx FIFO 0 and Rx FIFO 1) can be done by way of the MCAN_RXF0C and MCAN_RXF1C registers. Each Rx FIFO can be configured to store up to 64 received messages.

After acceptance filtering the received messages that passed are transferred to the Rx FIFO. The filter mechanisms available for the Rx FIFO 0 and Rx FIFO 1 are described in Section 45.5.13.1. The Rx FIFO element is described in Section 45.5.16.2.

The Rx FIFO watermark can be used to prevent an Rx FIFO overflow. If the Rx FIFO fill level reaches the Rx FIFO watermark configured by the MCAN_RXFnC[30:24] FnWM filed (where: n = 0 or 1), an interrupt flag MCAN_IR.RF0W/MCAN_IR.RF1W is set.

When the Rx FIFO Put Index reaches the Rx FIFO Get Index (MCAN_RXFnS[21:16] FnPI = MCAN_RXFnS[13:8] FnGI), an Rx FIFO Full condition is signaled by the MCAN_RXFnS[24] FnF status bit and interrupt flag MCAN_IR.RF0F/MCAN_IR.RF1F is set. Figure 45-15 shows Rx FIFO Status. The FIFOs fill level is presented in the MCAN_RXFnS[6:0] FnFL field (the number of elements stored in Rx FIFO).

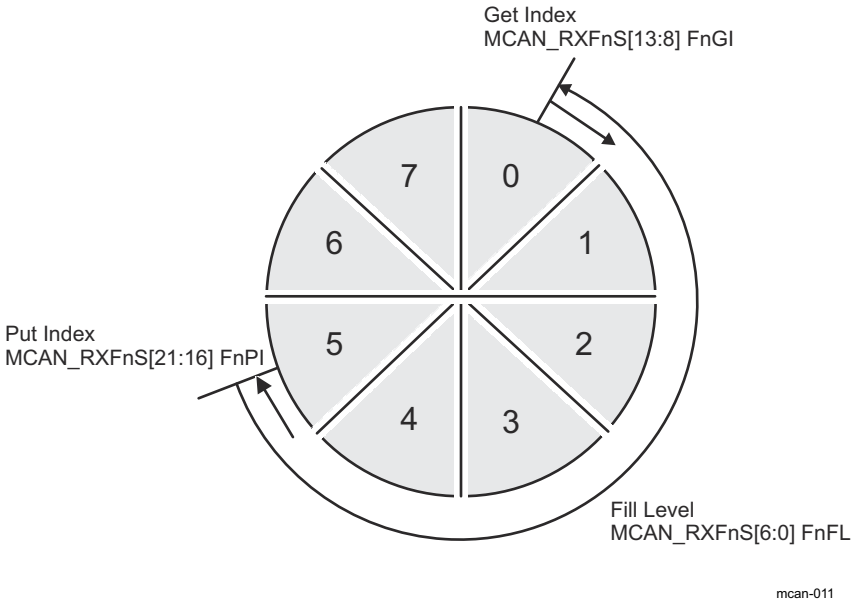


Figure 45-15. Rx FIFO Status

Rx FIFOs start address in the Message RAM (MCAN_RXFnC[15:2]FnSA field) has to be configured when reading from an Rx FIFO (Rx FIFO Get Index - MCAN_RXFnS[13:8] FnGI). Table 45-7 presents Rx Buffer/Rx FIFO Element Size for different Rx Buffer / Rx FIFO Data Field Size which is configured by way of the MCAN_RXESC register.

Table 45-7. Rx Buffer/Rx FIFO Element Size

MCAN_RXESC Register RBDS/F0DS/F1DS Bits	Data Field [bytes]	FIFO Element Size [RAM words]
000	8	4
001	12	5
010	16	6
011	20	7
100	24	8
101	32	10
110	48	14
111	64	18

45.5.13.2.1 Rx FIFO Blocking Mode

The Rx FIFO blocking mode is the default operation mode for the Rx FIFOs and is configured by MCAN_RXFnC[31] FnOM = 0.

If an Rx FIFO full condition is reached (MCAN_RXFnS[21:16] FnPI = MCAN_RXFnS[13:8] FnGI), no further messages are written to the corresponding Rx FIFO until at least one message has been read out and the Rx FIFO Get Index has been incremented. An Rx FIFO full condition is signaled by the MCAN_RXFnS[24] FnF = 1 and interrupt flag MCAN_IR.RF0F/MCAN_IR.RF1F is set.

In case a message is received while the corresponding Rx FIFO is full, this message is rejected and the message lost condition is signaled by MCAN_RXFnS[25] RFnL = 1 and interrupt flag MCAN_IR.RF0L/ MCAN_IR.RF1L is set.

45.5.13.2.2 Rx FIFO Overwrite Mode

The Rx FIFO overwrite mode is configured by MCAN_RXFnC[31] FnOM = 1. When an Rx FIFO full condition is reached (MCAN_RXFnS[21:16] FnPI = MCAN_RXFnS[13:8] FnGI) signaled by MCAN_RXFnS[24] FnF = 1, the next accepted message for the FIFO overwrites the oldest FIFO message. Put index/Get index are both incremented by one.

In overwrite mode if an Rx FIFO full condition is signaled, reading of the Rx FIFO elements starts at least at get index + 1. The reason for this is a received message is written to the Message RAM (Put index) while the Host CPU is reading from the Message RAM (Get index). In this case, inconsistent data can be read from the respective Rx FIFO element. The problem is solved by adding an offset to the Get index when reading from the Rx FIFO. The offset depends on how fast the Host CPU accesses the Rx FIFO. Figure 45-16 shows an offset of two with respect to the Get index when reading the Rx FIFO. In this case, the two messages stored in element 1 and 2 are lost.

After reading from the Rx FIFO, the number of the last element read has to be written to the Rx FIFO Acknowledge Index MCAN_RXFnA[5:0] FnAI. This increments the get index to that element number. In case the Put index has not been incremented to this Rx FIFO element, the Rx FIFO full condition is reset (MCAN_RXFnS[24] FnF = 0).

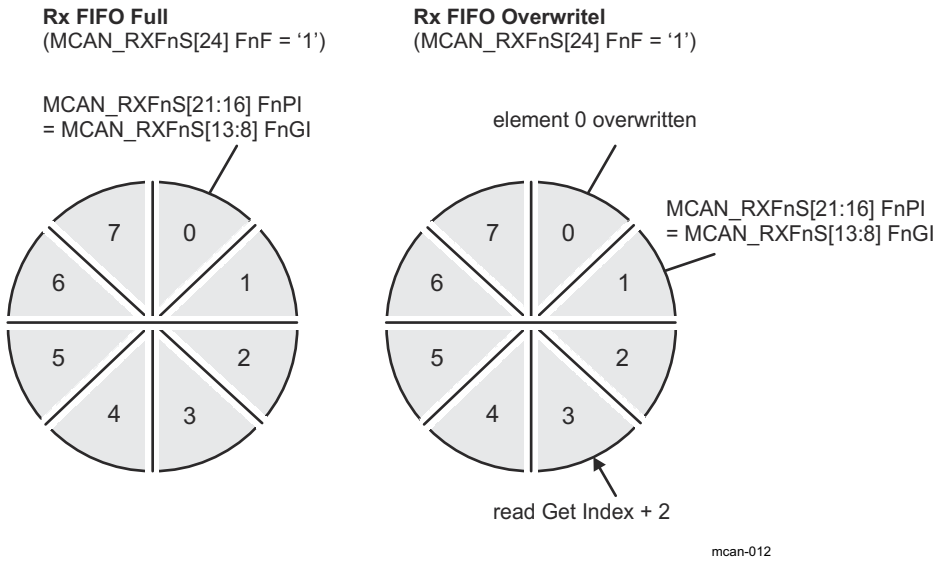


Figure 45-16. Rx FIFO Overflow Handling

45.5.13.3 Dedicated Rx Buffers

The MCAN supports up to 64 dedicated Rx buffers. The start address of the Rx buffers section in the Message RAM is configured by way of the MCAN_RXBC.RBSA field. To store in an Rx Buffer a Standard or Extended Message ID Filter Element with SFEC/EFEC = 111 and SFID2/EFID2[10:9] = 00 has to be configured (see Section 45.5.16.5 and Section 45.5.16.6).

After a received message has been accepted by a filter element, the message is stored into the Rx Buffer in the Message RAM referenced by the filter element (the format is the same as for an Rx FIFO element). In addition, the flag MCAN_IR.DRX (message stored in Dedicated Rx Buffer) is set.

Table 45-8 shows an example filter configuration for Rx buffers.

Table 45-8. Example Filter Configuration for Rx Buffers

Filter Element	SFID1[10:0] EFID1[28:0]	SFID2[10:9] EFID2[10:9]	SFID2[5:0] EFID2[5:0]
0	ID message 1	00	00 0000
1	ID message 2	00	00 0001
2	ID message 3	00	00 0010

After the last word of a matching received message has been written to the Message RAM, the respective New Data flag in register MCAN_NDAT1/MCAN_NDAT1 is set. As long as the New Data flag is set, the respective Rx Buffer is locked against updates from received matching frames. The New Data flags have to be reset by the Host CPU by writing a 1 to the respective bit position.

While an Rx buffer New Data flag is set, a Message ID Filter Element referencing this specific Rx Buffer does not match, causing the acceptance filtering to continue. Following Message ID Filter Elements can cause the received message to be stored into another Rx Buffer, into an Rx FIFO, or the message can be rejected, depending on filter configuration.

45.5.13.3.1 Rx Buffer Handling

Rx Buffer Handling include the following steps:

- Reset interrupt flag MCAN_IR.DRX
- Read New Data registers
- Read messages from Message RAM
- Reset New Data flags of processed messages

45.5.14 Tx Handling

The Tx handler is used to handle the Tx requests. The Tx handler controls the transfer of transmit messages from the dedicated Tx buffers, the Tx FIFO, and the Tx Queue to the CAN Core, the Tx Event FIFO, and the Put and Get Index operations. The MCAN module supports up to 32 Tx buffers. These Tx buffers can be configured as dedicated Tx buffers, Tx FIFO, or Tx Queue and as combination of dedicated Tx buffers/Tx FIFO or dedicated Tx buffers/Tx Queue. For each Tx Buffer element Classical CAN or CAN FD transmission mode can be configured. Section 45.5.16.3 describes the Tx Buffer Element. Table 45-9 shows the possible configurations for message transmission.

Table 45-9. Possible Configurations for Message Transmission

MCAN_CCCR Register		Tx Buffer Element		Frame Transmission
BRSE	FDOE	FDF	BRS	
ignored	0	ignored	ignored	Classic CAN
0	1	0	ignored	Classic CAN
0	1	1	ignored	CAN FD without bit rate switching
1	1	0	ignored	Classic CAN
1	1	1	0	CAN FD without bit rate switching
1	1	1	1	CAN FD with bit rate switching

When the Tx Buffer Request Pending (MCAN_TXBRP) register is updated, or when a transmission has been started the Tx Handler starts scanning to check for the highest priority pending Tx request. The Tx Buffer with the lowest Message ID has highest priority.

Note

AUTOSAR requires at least three Tx Queue buffers and support of transmit cancellation.

45.5.14.1 Transmit Pause

The transmit pause feature is intended for use in CAN networks where the CAN Message IDs are specific and cannot easily be changed. These Message IDs can have a higher priority than other defined Message IDs, while in a specific application their relative priority can be inverse. This allows for a case where one ECU sends a burst of CAN messages that cause another ECU CAN messages to be delayed (paused).

The transmit pause feature is enabled by the MCAN_CCCR.TXP bit. By default this bit is disabled (MCAN_CCCR.TXP = 0). Each time after successfully transmitted message, a pause for two CAN bit times occurs before the start of the next transmission. This allows the other CAN nodes in the network to transmit messages even if their Message IDs have lower priority.

45.5.14.2 Dedicated Tx Buffers

Dedicated Tx buffers are intended for message transmission under complete control of the Host CPU.

There are two options:

- Each dedicated Tx Buffer is configured with a specific Message ID.
- Two or more dedicated Tx buffers are configured with the same Message ID. In this case the Tx Buffer with the lowest buffer number is transmitted first.

After the data section has been updated, a transmission is requested by an Add Request. This is done using the MCAN_TXBAR[x]ARn bit (where x = 0 - 31). The requested messages arbitrate internally with messages from an optional Tx FIFO or Tx Queue and externally with messages on the CAN bus, and are sent out according to their Message ID.

Table 45-10 shows Tx Buffer/Tx FIFO/Tx Queue Element Size. A Dedicated Tx Buffer allocates element size 32-bit words in the Message RAM. The start address of a dedicated Tx Buffer in the Message RAM is calculated by adding transmit buffer index from 0 to 31 (MCAN_TXFQS.TFQP) × Element size to the Tx Buffer start address MCAN_TXBC.TBSA field.

Table 45-10. Tx Buffer, Tx FIFO, Tx Queue Element Size

MCAN_TXESC.TBDS	Data Field (bytes)	Element Size (RAM Words)
000	8	4
001	12	5
010	16	6
011	20	7
100	24	8
101	32	10
110	48	14
111	64	18

45.5.14.3 Tx FIFO

Tx FIFO mode is configured by setting bit MCAN_TXBC.TFQM = 0. The stored in the Tx FIFO messages are transmitted starting with the message referenced by the Get Index MCAN_TXFQS.TFGI field. After each transmission the Get Index is incremented until the Tx FIFO is empty. The Tx FIFO Free Level MCAN_TXFQS.TFFL field indicates the number of the available free Tx FIFO elements. The Tx FIFO allows transmission of messages with the same Message ID from different Tx buffers in the order these messages have been written to the Tx FIFO.

New transmit messages have to be written to the Tx FIFO starting with the Tx Buffer referenced by the Put Index MCAN_TXFQS.TFQP field. After each Add Request (MCAN_TXBAR[x] ARn = 1) the Put Index is incremented to the next free Tx FIFO element. When the Put Index reaches the Get Index (MCAN_TXFQS.TFQP = MCAN_TXFQS.TFGI), Tx FIFO Full condition is signaled by bit MCAN_TXFQS.TFQF = 1. In this case, no further messages should be written to the Tx FIFO until the next message has been transmitted and the Get Index has been incremented.

The number of requested Tx buffers should not exceed the number of free Tx buffers as indicated by the Tx FIFO Free Level MCAN_TXFQS.TFFL field.

In case a transmission request for the Tx Buffer referenced by the Get Index is canceled, the Get Index is incremented to the next Tx Buffer with pending transmission request and the Tx FIFO Free Level MCAN_TXFQS.TFFL field is recalculated. In case transmission cancellation is applied to any other Tx Buffer - the Get Index and the FIFO Free Level remain unchanged.

A Tx FIFO element allocates element size 32-bit words in the Message RAM (see Table 45-10). The start address of the next available (free) Tx FIFO Buffer is calculated by adding Tx FIFO/Queue Put Index MCAN_TXFQS.TFQP (from 0 to 31) × Element Size to the Tx Buffer Start Address MCAN_TXBC.TBSA field.

45.5.14.4 Tx Queue

Tx Queue mode is configured by setting bit MCAN_TXBC.TFQM = 1. The stored in the Tx Queue messages are transmitted starting with the highest priority message (lowest Message ID). In case two or more Queue buffers are configured with the same Message ID, the Queue Buffer with the lowest buffer number is transmitted first.

New transmit messages have to be written to the Tx FIFO starting with the Tx Buffer referenced by the Put Index MCAN_TXFQS.TFQP field. Each Add Request cyclically increments the Put Index to the next free Tx Buffer. In case of Tx Queue Full condition (MCAN_TXFQS.TFQF = 1), the Put Index is not valid and no further message should be written to the Tx Queue until at least one of the requested messages has been sent out or a pending transmission request has been canceled.

The application can use the MCAN_TXBRP register instead of the Put Index and can place messages to any Tx Buffer without pending transmission request.

A Tx Queue Buffer allocates element size 32-bit words in the Message RAM (see Table 45-10). The start address of the next available (free) Tx Queue Buffer is calculated by adding Tx FIFO/Queue Put Index MCAN_TXFQS.TFQP (from 0 to 31) × Element Size to the Tx Buffer Start Address MCAN_TXBC.TBSA field.

45.5.14.5 Mixed Dedicated Tx Buffers/Tx FIFO

For this combination the Tx buffers section in the Message RAM is separated in two parts:

- Dedicated Tx buffers: the number of Dedicated Tx buffers is configured by the MCAN_TXBC.NDTB field
- Tx FIFO: the number of Tx buffers assigned to the Tx FIFO is configured by the MCAN_TXBC.TFQS field

If the MCAN_TXBC.TFQS field is empty (zero) - only Dedicated Tx buffers are used.

Tx prioritization:

- Scan Dedicated Tx buffers and oldest pending Tx FIFO Buffer (referenced by the MCAN_TXFQS.TFGI field)
- Buffer with lowest Message ID gets highest priority and is transmitted next

Figure 45-17 shows Mixed Dedicated Tx buffers/Tx FIFO example.

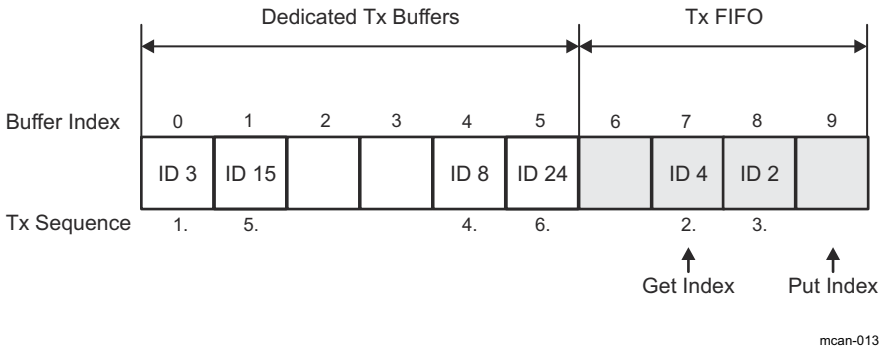


Figure 45-17. Mixed Dedicated Tx Buffers /Tx FIFO (example)

45.5.14.6 Mixed Dedicated Tx Buffers/Tx Queue

For this combination the Tx buffers section in the Message RAM is separated in two parts:

- Dedicated Tx buffers: the number of Dedicated Tx buffers is configured by the MCAN_TXBC.NDTB field
- Tx Queue: the number of Tx buffers assigned to the Tx Queue is configured by the MCAN_TXBC.TFQS field

If MCAN_TXBC.TFQS field is empty (zero) - only Dedicated Tx buffers are used.

Tx prioritization:

- Scan all Tx buffers with activated transmission request
- Tx Buffer with lowest Message ID gets highest priority and is transmitted next

Figure 45-18 shows Mixed Dedicated Tx buffers/Tx Queue example.

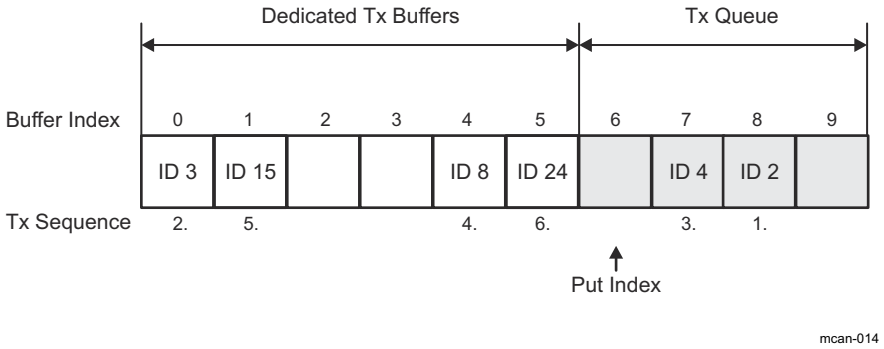


Figure 45-18. Mixed Dedicated Tx Buffers /Tx Queue (example)

45.5.14.7 Transmit Cancellation

This feature is especially intended for gateway and AUTOSAR based applications. The Host CPU can cancel a requested transmission from a dedicated Tx Buffer or a Tx Queue Buffer by setting bit MCAN_TXBCR[n] CRn = 1 (where n = 0 - 31). The corresponding bit position n is equivalent to the number of the Tx buffer.

Transmit cancellation is not intended for Tx FIFO operation.

Successful cancellation is signaled by setting the corresponding bit of the MCAN_TXBCF register (MCAN_TXBCF[n] CFn = 1).

If transmission from a Tx Buffer is already ongoing and a transmit cancellation is requested, the corresponding MCAN_TXBRP[n] TRPn bit remains set as long as the transmission is in progress. If the transmission was successful, the corresponding MCAN_TXBTO[n] TOn and MCAN_TXBCF[n] CFn bits are set. If the transmission was not successful, only the corresponding bit MCAN_TXBCF[n] CFn = 1.

Note

If a pending transmission is canceled immediately before this transmission has started, a short time window occurs where no transmission is started even if another message is also pending in this node. This can enable another node to transmit a message that can have a lower priority than the second message in this node.

45.5.14.8 Tx Event Handling

To support Tx Event Handling, the Message RAM has implemented a Tx Event FIFO section. Up to 32 Tx Event FIFO elements can be configured. Section 45.5.16.4 describes the Tx Event FIFO element. After message transmission on the CAN bus, Message ID and Timestamp are stored in a Tx Event FIFO element. To link a Tx Event to a Tx Event FIFO element, the Message Marker from the transmitted Tx Buffer is copied into the Tx Event FIFO element.

A Tx Event FIFO full condition is signaled by the MCAN_IR.TEFF bit. In this case no further elements are written to the Tx Event FIFO until at least one element has been read out and the Tx Event FIFO Get Index has been incremented (MCAN_TXEFS.EFGI). In case a Tx Event occurs while the Tx Event FIFO is full, this event is rejected and interrupt flag MCAN_IR.TEFL bit is set.

The Tx Event FIFO watermark can be configured to avoid a Tx Event FIFO overflow. When the Tx Event FIFO fill level reaches the Tx Event FIFO watermark configured by the MCAN_TXEFC.EFWM field, interrupt flag MCAN_IR.TEFW is set. When reading from the Tx Event FIFO, two times the Tx Event FIFO Get Index MCAN_TXEFS.EFGI field has to be added to the Tx Event FIFO start address MCAN_TXEFC.EFSA field.

45.5.15 FIFO Acknowledge Handling

The Get Indices of the two Rx FIFOs (Rx FIFO 0 or Rx FIFO 1) and the Tx Event FIFO are controlled by writing to the corresponding FIFO Acknowledge Index (see MCAN_RXF0A, MCAN_RXF1A, and MCAN_TXEFA). Writing to the FIFO Acknowledge Index sets the FIFO Get Index to the FIFO Acknowledge Index plus one and, thereby, updates the FIFO Fill Level.

There are two use cases:

- A single element has been read from the FIFO: the Get Index value is written to the FIFO Acknowledge Index.
- A sequence of elements has been read from the FIFO: the Get Index value (Index of the last element read) is written to the FIFO Acknowledge Index at the end of that read sequence.

The Host CPU has free access to the Message RAM. Special care has to be taken when reading FIFO elements in an arbitrary order (Get Index not considered). This can be useful when reading a High Priority Message from one of the two Rx FIFOs. In this case, the FIFO's Acknowledge Index should not be written because this would set the Get Index to a wrong position and also changes the FIFO's Fill Level. In this case, some of the older FIFO elements would be lost.

Note

The application has to make sure that a valid value is written to the FIFO Acknowledge Index. The MCAN module does not check for erroneous values.

45.5.16 Message RAM

The MCAN module has a Message RAM. The main purpose of the Message RAM is to store:

- Received Messages
- Transmit Messages
- Tx Event Elements
- Message ID Filter Elements

45.5.16.1 Message RAM Configuration

The MCAN module is configured to allocate up to 4352 words in the Message RAM. The Message RAM has a width of 32 bits.

The address range of the Message RAM (when MCAN is interfaced to M4 processor) is from 0x4007 8000 to 0x4007 C3FF. The address range of the Message RAM (when MCAN is interfaced to CPU1) is from 0x5 8000 to 0x5 C3FF.

The Message RAM is capable to include each of the sections listed in Figure 45-19. It is not necessary to configure each of the sections (a section in the Message RAM can be 0) and there is no restriction with respect to the sequence of the sections. For parity checking or ECC, a respective number of bits has to be added to each word. When the MCAN module addresses the Message RAM, the MCAN addresses 32-bit words. The start addresses are configurable and are 32-bit word addresses.

The element size can be configured for:

- Rx FIFO 0, by way of the MCAN_RXESC.F0DS field
- Rx FIFO, 1 by way of the MCAN_RXESC.F1DS field
- Rx buffers, by way of the MCAN_RXESC.RBDS field
- Tx buffers, by way of the MCAN_TXESC.TBDS field

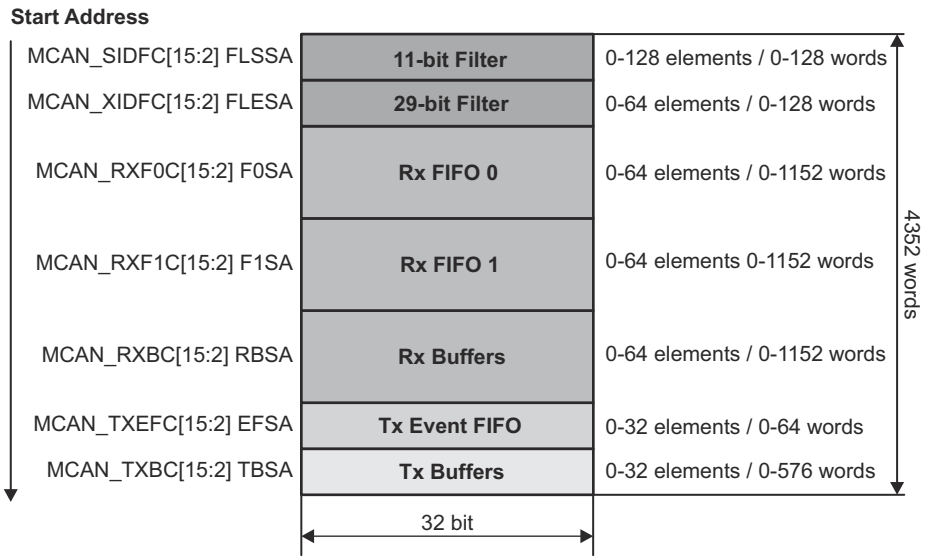


Figure 45-19. Message RAM Configuration

The Host CPU configures the following information in the Message RAM:

- Start addresses of the memory sections
- Number of elements in each section
- The size of the elements in some sections

Note

The MCAN module does not check for errors in the Message RAM configuration. The configuration of the start addresses of the different sections and the number of elements of each section has to be done carefully. This prevents falsification or loss of data.

45.5.16.2 Rx Buffer and FIFO Element

Up to 64 Rx buffers and two Rx FIFOs can be configured in the Message RAM. Each Rx FIFO section can be configured to store up to 64 received messages. The element size can be configured for storage of CAN FD messages with up to 64 bytes data field by way of the MCAN_RXESC register.

Figure 45-20 shows the Rx Buffer/Rx FIFO element structure. Table 45-11 shows the Rx Buffer/Rx FIFO element field descriptions.

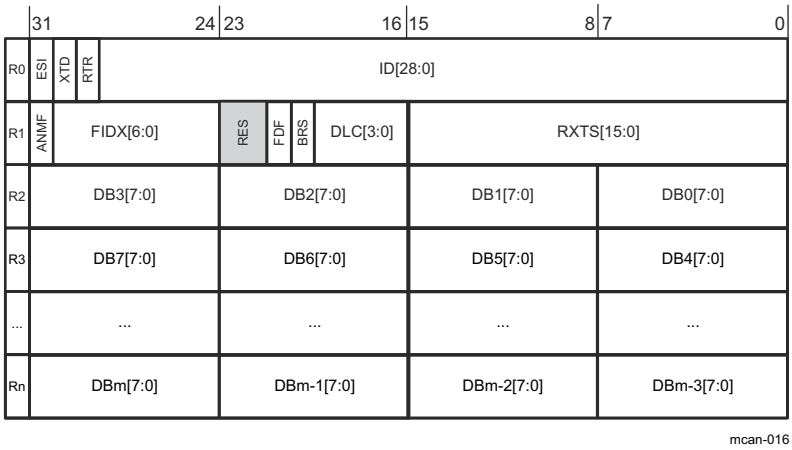


Figure 45-20. Rx Buffer/Rx FIFO Element Structure

Table 45-11. Rx Buffer/Rx FIFO Element Field Descriptions

Word	Bits	Field Name	Description
R0	31	ESI	Error State Indicator • 0x0: Transmitting node is error active • 0x1: Transmitting node is error passive
	30	XTD	Extended Identifier Signals to the Host CPU whether the received frame has a standard or extended identifier. • 0x0: 11-bit standard identifier • 0x1: 29-bit extended identifier
	29	RTR	Remote Transmission Request Signals to the Host CPU whether the received frame is a data frame or a remote frame. • 0x0: Received frame is a data frame • 0x1: Received frame is a remote frame **Note:** There are no remote frames in CAN FD format. In case a CAN FD frame was received (FDF = 1), RTR bit reflects the state of the reserved r1 bit (RES[23]). In CAN FD frames (FDF=1), the dominant RRS (Remote Request Substitution) bit replaces the RTR (Remote Transmission Request) bit.
	28:0	ID[28:0]	Identifier Standard or extended identifier depending on XTD bit. A standard identifier is stored into ID[28:18].

Table 45-11. Rx Buffer/Rx FIFO Element Field Descriptions (continued)

Word	Bits	Field Name	Description
R1	31	ANMF	Accepted Non-matching Frame Acceptance of non-matching frames can be enabled using the MCAN_GFC.ANFS and MCAN_GFC.ANFE fields. • 0x0: Received frame matching filter index FIDX field • 0x1: Received frame did not match any Rx filter element
	30:24	FIDX[6:0]	Filter Index 0x0-0x7F (0-127): Index of matching Rx acceptance filter element (invalid if ANMF = 1). Range is 0 to MCAN_SIDFC.LSS - 1 respectively MCAN_XIDFC.LSE - 1.
	23:22	RES	Reserved
	21	FDF	FD Format • 0x0: Standard frame format • 0x1: CAN FD frame format (new DLC-coding and CRC)
	20	BRS	Bit Rate Switch • 0x0: Frame received without bit rate switching • 0x1: Frame received with bit rate switching
	19:16	DLC[3:0]	Data Length Code • 0x0-0x8 (0-8): CAN + CAN FD: received frame has 0-8 data bytes • 0x9-0xF (9-15): CAN: received frame has 8 data bytes • 0x9-0xF (9-15): CAN FD: received frame has 12/16/20/24/32/48/64 data bytes
	15:0	RXTS[15:0]	Rx Timestamp Timestamp Counter value captured on start of frame reception. Resolution depending on configuration of the Timestamp Counter Prescaler MCAN_TSCC.TCP.
R2	31:24	DB3[7:0]	Data Byte 3
	23:16	DB2[7:0]	Data Byte 2
	15:8	DB1[7:0]	Data Byte 1
	7:0	DB0[7:0]	Data Byte 0
R3	31:24	DB7[7:0]	Data Byte 7
	23:16	DB6[7:0]	Data Byte 6
	15:8	DB5[7:0]	Data Byte 5
	7:0	DB4[7:0]	Data Byte 4
...	...	...	...
Rn	31:24	DBm[7:0]	Data Byte m
	23:16	DBm-1[7:0]	Data Byte m-1
	15:8	DBm-2[7:0]	Data Byte m-2
	7:0	DBm-3[7:0]	Data Byte m-3

Note: Depending on the configuration of the element size (MCAN_RXESC), between two and sixteen 32-bit words (Rn = 3-17) are used for storage of a CAN message's data field.

45.5.16.3 Tx Buffer Element

The Tx buffers section can be configured to hold dedicated Tx buffers as well as a Tx FIFO/Tx Queue. In case that the Tx buffers section is shared by dedicated Tx buffers and a Tx FIFO/Tx Queue, the dedicated Tx buffers start at the beginning of the Tx buffers section followed by the buffers assigned to the Tx FIFO or Tx Queue. The Tx Handler makes difference between dedicated Tx buffers and Tx FIFO/Tx Queue by way of the MCAN_TXBC.TFQS and MCAN_TXBC.NDTB fields. The element size can be configured for storage of CAN FD messages with up to 64 bytes data field by way of the MCAN_TXESC register.

Figure 45-21 shows the Tx Buffer element structure. Table 45-12 shows the Tx Buffer element field descriptions.

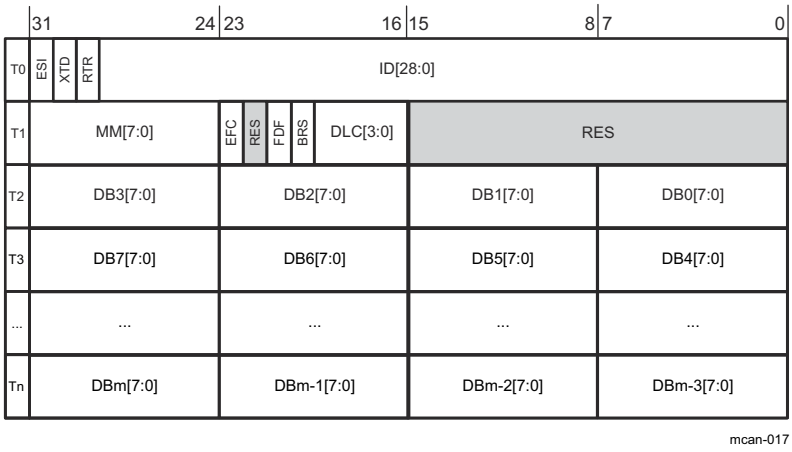


Figure 45-21. Tx Buffer Element Structure

Table 45-12. Tx Buffer Element Field Descriptions

Word	Bits	Field Name	Description
T0	31	ESI	Error State Indicator • 0x0: ESI bit in CAN FD format depends only on error passive flag • 0x1: ESI bit in CAN FD format transmitted recessive **Note:** The ESI bit of the transmit buffer is ORed with the error passive flag to decide the value of the ESI bit in the transmitted CAN FD frame. As required by the CAN FD protocol specification, an error active node can optionally transmit the ESI bit recessive, but an error passive node always transmits the ESI bit recessive.
	30	XTD	Extended Identifier • 0x0: 11-bit standard identifier • 0x1: 29-bit extended identifier
	29	RTR	Remote Transmission Request • 0x0: Transmit data frame • 0x1: Transmit remote frame **Note:** When RTR = 1, the MCAN module transmits a remote frame according to ISO11898-1:2015, even if the MCAN_CCCR.FDOE bit enables the transmission in CAN FD format.
	28:0	ID[28:0]	Identifier Standard or extended identifier depending on XTD bit. A standard identifier has to be written to ID[28:18].

Table 45-12. Tx Buffer Element Field Descriptions (continued)

Word	Bits	Field Name	Description
T1	31:24	MM[7:0]	Message Marker Written by Host CPU during Tx Buffer configuration. Copied into Tx Event FIFO element for identification of Tx message status (see also MM[7:0] field in Table 45-13).
	23	EFC	Event FIFO Control • 0x0: Don't store Tx events • 0x1: Store Tx events
	22	RES	Reserved
	21	FDF	FD Format • 0x0: Frame transmitted in Classic CAN format • 0x1: Frame transmitted in CAN FD format
	20	BRS	Bit Rate Switch • 0x0: CAN FD frames transmitted without bit rate switching • 0x1: CAN FD frames transmitted with bit rate switching **Note:** ESI, FDF, and BRS bits are only evaluated when CAN FD operation is enabled using the MCAN_CCCR.FDOE bit. BRS bit is only evaluated when MCAN_CCCR.BRSE = 1.
	19:16	DLC[3:0]	Data Length Code • 0x0-0x8 (0-8): CAN + CAN FD: transmit frame has 0-8 data bytes • 0x9-0xF (9-15): CAN: transmit frame has 8 data bytes • 0x9-0xF (9-15): CAN FD: transmit frame has 12/16/20/24/32/48/64 data bytes
	15:0	RES	Reserved
T2	31:24	DB3[7:0]	Data Byte 3
	23:16	DB2[7:0]	Data Byte 2
	15:8	DB1[7:0]	Data Byte 1
	7:0	DB0[7:0]	Data Byte 0
T3	31:24	DB7[7:0]	Data Byte 7
	23:16	DB6[7:0]	Data Byte 6
	15:8	DB5[7:0]	Data Byte 5
	7:0	DB4[7:0]	Data Byte 4
...	...	...	...
Tn	31:24	DBm[7:0]	Data Byte m
	23:16	DBm-1[7:0]	Data Byte m-1
	15:8	DBm-2[7:0]	Data Byte m-2
	7:0	DBm-3[7:0]	Data Byte m-3

Note

Depending on the configuration of the element size (MCAN_TXESC), between two and sixteen 32-bit words (Tn = 3-17) are used for storage of a CAN message's data field.

45.5.16.4 Tx Event FIFO Element

Each element stores information about transmitted messages. By reading the Tx Event FIFO the Host CPU gets this information in the order the messages were transmitted. Status information about the Tx Event FIFO can be obtained from the MCAN_TXEFS register.

Figure 45-22 shows the Tx Event FIFO element structure. Table 45-13 shows the Tx Event FIFO element field descriptions.

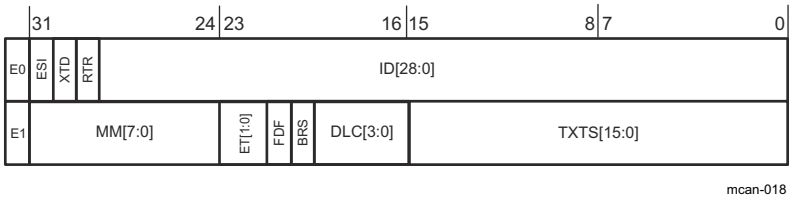


Figure 45-22. Tx Event FIFO Element Structure

Table 45-13. Tx Event FIFO Element Field Descriptions

Word	Bits	Field Name	Description
E0	31	ESI	Error State Indicator • 0x0: Transmitting node is error active • 0x1: Transmitting node is error passive
	30	XTD	Extended Identifier • 0x0: 11-bit standard identifier • 0x1: 29-bit extended identifier
	29	RTR	Remote Transmission Request • 0x0: Data frame transmitted • 0x1: Remote frame transmitted
	28:0	ID[28:0]	Identifier Standard or extended identifier depending on XTD bit. A standard identifier has to be written to ID[28:18].

Table 45-13. Tx Event FIFO Element Field Descriptions (continued)

Word	Bits	Field Name	Description
E1	31:24	MM[7:0]	Message Marker Copied from Tx Buffer into Tx Event FIFO element for identification of Tx message status (see also MM[7:0] field in Table 45-12).
	23:22	ET[1:0]	Event Type • 0x0: Reserved • 0x1: Tx event • 0x2: Transmission in spite of cancellation (always set for transmissions in DAR mode) • 0x3: Reserved
	21	FDF	FD Format • 0x0: Standard frame format • 0x1: CAN FD frame format (new DLC-coding and CRC)
	20	BRS	Bit Rate Switch • 0x0: Frame transmitted without bit rate switching • 0x1: Frame transmitted with bit rate switching
	19:16	DLC[3:0]	Data Length Code • 0x0-0x8 (0-8): CAN + CAN FD: frame with 0-8 data bytes transmitted • 0x9-0xF (9-15): CAN: frame with 8 data bytes transmitted • 0x9-0xF (9-15): CAN FD: frame with 12/16/20/24/32/48/64 data bytes transmitted
	15:0	TXTS[15:0]	Tx Timestamp Timestamp Counter value captured on start of frame transmission. Resolution depending on configuration of the Timestamp Counter Prescaler MCAN_TSCC.TCP filed.

45.5.16.5 Standard Message ID Filter Element

Up to 128 filter elements can be configured for 11-bit standard IDs. When accessing a Standard Message ID Filter element, the element address is the Filter List Standard Start Address MCAN_SIDFC.FLSSA field plus the index of the filter element (0-127).

Figure 45-23 shows the Standard Message ID Filter element structure. Table 45-14 shows the Standard Message ID Filter element field descriptions.

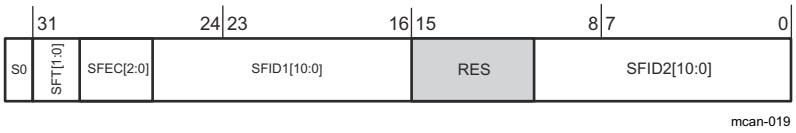


Figure 45-23. Standard Message ID Filter Element Structure

Table 45-14. Standard Message ID Filter Element Field Descriptions

Word	Bits	Field Name	Description
S0	31:30	SFT[1:0]	Standard Filter Type • 0x0: Range filter from SFID1 to SFID2 (SFID2 ≥ SFID1) • 0x1: Dual ID filter for SFID1 or SFID2 • 0x2: Classic filter: SFID1 = filter; SFID2 = mask • 0x3: Filter element disabled **Note:** With SFT = 11 the filter element is disabled and the acceptance filtering continues (same behavior as with SFEC = 000)
	29:27	SFEC[2:0]	Standard Filter Element Configuration All enabled filter elements are used for acceptance filtering of standard frames. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If SFEC = 100, 101, or 110 match sets interrupt flag MCAN_IR.HPM and, if enabled, an interrupt is generated. In this case, the MCAN_HPMS register is updated with the status of the priority match. • 0x0: Disable filter element • 0x1: Store in Rx FIFO 0 if filter matches • 0x2: Store in Rx FIFO 1 if filter matches • 0x3: Reject ID if filter matches • 0x4: Set priority if filter matches • 0x5: Set priority and store in FIFO 0 if filter matches • 0x6: Set priority and store in FIFO 1 if filter matches • 0x7: Store into Rx Buffer , configuration of SFT[1:0] ignored
	26:16	SFID1[10:0]	Standard Filter ID 1 When filtering for Rx buffers this field defines the ID of a standard message to be stored. The received identifiers must match exactly, no masking mechanism is used.
	15:11	RES	Reserved
	10:0	SFID2[10:0]	Standard Filter ID 2 This bit field has a different meaning depending on the configuration of SFEC: • SFEC = 001 - 110: Second ID of standard ID filter element • SFEC = 111: Filter for Rx buffers
		SFID2[10:9]	This filed is decides whether the received message is stored into an Rx Buffer or treated as message A, B, or C of the debug message sequence. • 0x0: Store message into an Rx Buffer • 0x1: Debug Message A • 0x2: Debug Message B • 0x3: Debug Message C **Note:** Debug feature is not supported.
		SFID2[8:6]	This field is used to control the filter event pins at the Extension Interface. A one at the respective bit position enables generation of a pulse at the related filter event pin with the duration of one MCAN_ICKL period in case the filter matches. **Note:** Only two filter event pins are supported.
		SFID2[5:0]	This field defines the offset to the Rx Buffer Start Address MCAN_RXBC.RBSA field for storage of a matching message.

45.5.16.6 Extended Message ID Filter Element

Up to 64 filter elements can be configured for 29-bit extended IDs. When accessing an Extended Message ID Filter element, the element address is the Filter List Extended Start Address MCAN_XIDFC.FLESA field plus two times the index of the filter element (0-63).

Figure 45-24 shows the Extended Message ID Filter element structure. Table 45-15 shows the Extended Message ID Filter element field descriptions.

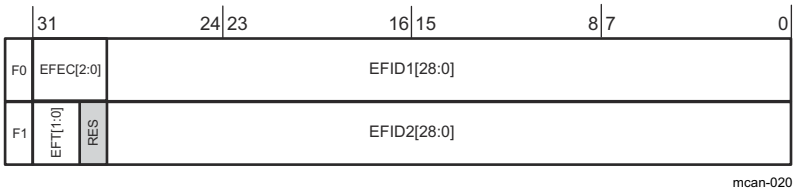


Figure 45-24. Extended Message ID Filter Element Structure

Table 45-15. Extended Message ID Filter Element Field Descriptions

Word	Bits	Field Name	Description
F0	31:29	EFEC[2:0]	Extended Filter Element Configuration All enabled filter elements are used for acceptance filtering of extended frames. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If EFEC = 100, 101, or 110 match sets interrupt flag MCAN_IR.HPM and, if enabled, an interrupt is generated. In this case, the MCAN_HPMS register is updated with the status of the priority match. • 0x0: Disable filter element • 0x1: Store in Rx FIFO 0 if filter matches • 0x2: Store in Rx FIFO 1 if filter matches • 0x3: Reject ID if filter matches • 0x4: Set priority if filter matches • 0x5: Set priority and store in FIFO 0 if filter matches • 0x6: Set priority and store in FIFO 1 if filter matches • 0x7: Store into Rx Buffer or as debug message, configuration of EFT[1:0] ignored
	28:0	EFID1[28:0]	Extended Filter ID 1 First ID of extended ID filter element. When filtering for Rx buffers this field defines the ID of an extended message to be stored. The received identifiers must match exactly, only XIDAM masking mechanism (see Section 45.5.13.1.5) is used.

Table 45-15. Extended Message ID Filter Element Field Descriptions (continued)

Word	Bits	Field Name	Description
F1	31:30	EFT[1:0]	Extended Filter Type • 0x0: Range filter from EFID1 to EFID2 (EFID2 ≥ EFID1) • 0x1: Dual ID filter for EFID1 or EFID2 • 0x2: Classic filter: EFID1 = filter, EFID2 = mask • 0x3: Range filter from EFID1 to EFID2 (EFID2 ≥ EFID1), XIDAM mask not applied
	29	RES	Reserved
	28:0	EFID2[28:0]	Extended Filter ID 2 This bit field has a different meaning depending on the configuration of EFEC: • EFEC = 001 - 110: Second ID of extended ID filter element • EFEC = 111: Filter for Rx buffers
		EFID2[10:9]	This field decides whether the received message is stored into an Rx Buffer or treated as message A, B, or C of the debug message sequence. • 0x0: Store message into an Rx Buffer • 0x1: Debug Message A • 0x2: Debug Message B • 0x3: Debug Message C **Note:** Debug feature is not supported.
		EFID2[8:6]	This field is used to control the filter event pins at the Extension Interface. A one at the respective bit position enables generation of a pulse at the related filter event pin with the duration of one MCAN_ICKL period in case the filter matches. **Note:** Only two filter event pins are supported.
		EFID2[5:0]	This field defines the offset to the Rx Buffer Start Address MCAN_RXBC.RBSA field for storage of a matching message.

45.6 Software

45.6.1 MCAN Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/mcan

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

45.6.1.1 MCAN Internal Loopback with Interrupt - CM

FILE: mcan_ex1_loopback.c

This example demonstrates Loopback functionality of the MCAN (CAN FD) module. The internal loopback mode is chosen. The transmitted message will be received by the node. All action is internal to the device, hence transmission will not be visible on the MCAN_TX pin. Uses the last address of memory for Rx buffer.

Before running this example, please run the mcan_config_c28x example It will initialize the clock, configure the GPIOs.

External Connections
- None.

Watch Variables
- error - Checks if there is an error that occurred when the data was sent using internal loopback.

45.6.1.2 MCAN Internal Loopback with Interrupt

FILE: mcan_ex1_loopback.c

This example shows the MCAN Loopback functionality. The internal loopback mode is entered. The sent message should be received by the node. Use the last address of memory for Rx buffer.

External Connections
- None.

Watch Variables
- error - Checks if there is an error that occurred when the data was sent using internal loopback.

45.6.1.3 MCAN External Loopback with Interrupt - CM

FILE: mcan_ex2_external_loopback.c

This example shows the MCAN External Loopback functionality. The external loopback is done between two MCAN Controllers. As there is only one MCAN that exist this example can be changed to make MCAN Transmit or Receive based on define selected. The GPIOs of MCAN should be connected to a CAN Transceiver

Before running this example, please run the mcan_config_c28x example It will initialize the clock, configure the GPIOs.

Selection of Mode : A define has to be selected to make the MCAN to transmit or receive.
- TRANSMIT - MCAN to Transmit messages.
- RECEIVE - MCAN to Receive Messages.

Run the example as with RECEIVE Define on one MCAN Controller before running it as Transmit.

Hardware Required
- A C2000 board with CAN transceiver

External Connections
- MCAN is on GPIO30 (MCANRXA) and GPIO31 (MCANTXA)

Watch Variables
- isrIntr0Flag - The flag has initial value as no. of messages to be transmitted and its value decrements after a message is transmitted.

- isrIntr1Flag - The flag has initial value as no. of messages that are received and its value decrements after a message is successfully received.
- error - Checks if there is an error that occurred when the data was sent using internal loopback

45.6.1.4 MCAN Loopback with Interrupts Example Using SYSCONFIG Tool

FILE: mcan_ex3_loopback_syscfg.c

This example illustrates the MCAN Loopback functionality. The internal loopback mode is entered. The message transmitted would be received by the node. The last address of memory is used for the Rx buffer. Peripheral configuration is done through SYSCONFIG

External Connections
- None.

Watch Variables
- error - Checks if there is an error that occurred when the data was sent using internal loopback.

45.6.1.5 MCAN receive using Rx Buffer

FILE: mcan_ex4_receive.c

This example demonstrates the MCAN receive function. Communication is done between two CAN nodes. The transmitting node could be another MCU or a CAN bus analysis tool capable of transmitting CAN FD frames. The transmit and receive pins of the MCAN module should be connected to a CAN transceiver. Nominal Bit Rate of 500 kbps & Data bit rate of 1 Mbps is used

Only Standard frame with message ID 0x4 is received.

If another C2000 MCU is used as the transmitter, mcan_ex3_transmit.c can be run on it for the transmit function.

Hardware Required
- A C2000 board with CAN transceiver

External Connections
Both nodes should communicate through CAN FD capable transceivers.

- MCAN is on DEVICE_GPIO_PIN_CANRXA (MCANRXA)
- and DEVICE_GPIO_PIN_CANTXA (MCANTXA)

Watch Variables
- rxMsg

45.6.1.6 MCAN External Reception (with mask filter) into RX-FIFO1

FILE: mcan_ex5_mask_filter_receive.c

This example demonstrates Receiving, with mask filter configuration. The transmitting node could be a CAN FD capable controller or a CAN bus analysis tool capable of transmitting CAN FD frames. Bits 0, 1 & 3 of the identifier are masked. So these bits can have any value. This is achieved by using stdFiltelem.sfid1 = 00000001111 and stdFiltelem.sfid2 (mask 0 for X) = 11111110100, which means any message with an ID of 0b0000000X1XX are received and stored into the FIFO. i.e. Following STD IDs are received: 0x004, 0x005, 0x006, 0x007, 0x00C, 0x00D, 0x00E, 0x00F. All other IDs are not received. Classic bit-mask filter is used. This example may be used in conjunction with mcan_ex3_transmit.

The transmit and receive pins of the MCAN module should be connected to a CAN Transceiver. Nominal Bit Rate of 500 kbps and Data bit rate of 1 Mbps is used.

Hardware Required
- A C2000 board with CAN transceiver

External Connections
Both nodes should communicate through CAN FD capable transceivers.

- MCAN is on DEVICE_GPIO_PIN_CANRXA (MCANRXA)

- and DEVICE_GPIO_PIN_CANTXA (MCANTXA)

Watch Variables

- rxMsg

45.6.1.7 MCAN Classic frames transmission using Tx Buffer

FILE: mcan_ex7_classic_transmit.c

This simple example shows external communication between the MCAN module and another CAN node. It shows how to transmit classic CAN frames. The GPIOs of MCAN should be connected to a CAN Transceiver. Bit Rate is 500 kbps. Extended Identifier 0x15A5A5A5 is transmitted with 8 data bytes.

Hardware Required

- A C2000 board with CAN transceiver

External Connections

Both nodes should communicate through CAN transceivers.

- MCAN is on DEVICE_GPIO_PIN_CANRXA (MCANRXA)
- and DEVICE_GPIO_PIN_CANTXA (MCANTXA)

Watch Variables

- txMsg

45.6.1.8 MCAN External Reception (with RANGE filter) into RX-FIFO1

FILE: mcan_ex8_range_filter_receive.c

This example demonstrates Receiving, with RANGE filter configuration. The transmitting node could be a CAN FD capable controller or a CAN bus analysis tool capable of transmitting CAN FD frames. Only Extended IDs from 0x1FFFFF23 to 0x1FFFFF46 are received. Other IDs are not received. RXFIFO1 starts at an offset of 748 (2EC). MCAN Message RAM starts at 0x58000, so received messages will be copied starting at address 0x582EC. Note that as long as the ID matches, "classic" CAN frames will also be received.

The transmit and receive pins of the MCAN module should be connected to a CAN Transceiver. Nominal Bit Rate of 500 kbps and Data bit rate of 1 Mbps is used.

Hardware Required

- A C2000 board with CAN transceiver

External Connections

Both nodes should communicate through CAN FD capable transceivers.

- MCAN is on DEVICE_GPIO_PIN_CANRXA (MCANRXA)
- and DEVICE_GPIO_PIN_CANTXA (MCANTXA)

Watch Variables

- rxMsg

45.6.1.9 MCAN External Transmit using Tx Buffer

FILE: mcan_ex9_transmit.c

This example demonstrates the MCAN External Transmit function. External communication is done between two CAN nodes. The receiving node could be another MCU or a CAN bus analysis tool capable of Receiving/ ACKnowledging transmitted frames. The transmit and receive pins of the MCAN module should be connected to a CAN Transceiver. Nominal Bit Rate of 500 kbps and Data bit rate of 1 Mbps is used. Standard Identifier (STD ID) 0x4 is transmitted with 64 data bytes. #defines that are not required for this test case have been commented out. However, they have been left in the code should the scope of this code be expanded to include Receive and FIFO functions.

If another C2000 MCU is used as the receiver, mcan_ex4_receive.c can be run on it for the receive function.

Hardware Required

- A C2000 board with CAN transceiver

External Connections
Both nodes should communicate through CAN FD capable transceivers.

- MCAN is on DEVICE_GPIO_PIN_CANRXA (MCANRXA)
- and DEVICE_GPIO_PIN_CANTXA (MCANTXA)

Watch Variables
- txMsg

45.6.1.10 MCAN receive using Rx Buffer

FILE: mcan_ex10_receive_multiple_buffers.c

This example demonstrates how to receive MCAN messages in multiple buffers, including the configuration and handling of the messages. Communication is done between atleast two CAN nodes. The transmitting node could be another MCU or a CAN bus analysis tool capable of transmitting CAN FD frames. The transmit and receive pins of the MCAN module should be connected to a CAN transceiver. Nominal Bit Rate of 500 kbps & Data bit rate of 1 Mbps is used

Standard frames with message IDs 0x123, 0x124, 0x125, 0x126 are received. (For Extended frames, same procedure to be followed for configuration by adding Extended Message ID Filter Element(s))

Hardware Required
- A C2000 board with CAN transceiver

External Connections
Both nodes should communicate through CAN FD capable transceivers.

- MCAN is on DEVICE_GPIO_PIN_CANRXA (MCANRXA)
- and DEVICE_GPIO_PIN_CANTXA (MCANTXA)

Watch Variables
- rxMsg

45.7 MCAN Registers

This section describes the Modular Controller Area Network (MCAN) module registers.

45.7.1 MCAN Base Address Table (C28)

Bit Field Name		DriverLib Name	Base Address	CPU1	CPU2	DMA	CLA	Pipeline Protected
Instance	Structure							
McanaSsRegs	MCANSS_REGS	MCANASS_BASE	0x0005_C400	YES	-	-	-	YES
McanaRegs	MCAN_REGS	MCANA_BASE	0x0005_C600	YES	-	-	-	YES
McanaErrRegs	MCAN_ERROR_REGS	MCANA_ERROR_BASE	0x0005_C800	YES	-	-	-	YES
MCAN Message RAM	-	-	0x0005_8000	YES	-	-	-	YES

45.7.2 CM MCAN Base Address Table (CM)

Bit Field Name		DriverLib Name	Base Address	DMA	Pipeline Protected
Instance	Structure				
McanaSsRegs	MCANSS_REGS	MCANASS_BASE	0x4007_C400	-	YES
McanaRegs	MCAN_REGS	MCANA_BASE	0x4007_C600	-	YES
McanaErrRegs	MCAN_ERROR_REGS	MCANA_ERROR_BASE	0x4007_C800	-	YES
MCAN Message RAM	-	-	0x4007_8000	-	YES

45.7.3 MCANSS_REGS Registers

Table 45-16 lists the memory-mapped registers for the MCANSS_REGS registers. All register offset addresses not listed in Table 45-16 should be considered as reserved locations and the register contents should not be modified.

Table 45-16. MCANSS_REGS Registers

Offset (x8)	Offset (x16)	Acronym	Register Name	Write Protection	Section
0h	0h	MCANSS_PID	MCAN Subsystem Revision Register		Go
4h	2h	MCANSS_CTRL	MCAN Subsystem Control Register		Go
8h	4h	MCANSS_STAT	MCAN Subsystem Status Register		Go
Ch	6h	MCANSS_ICS	MCAN Subsystem Interrupt Clear Shadow Register		Go
10h	8h	MCANSS_IRS	MCAN Subsystem Interrupt Raw Satus Register		Go
14h	Ah	MCANSS_IECS	MCAN Subsystem Interrupt Enable Clear Shadow Register		Go
18h	Ch	MCANSS_IE	MCAN Subsystem Interrupt Enable Register		Go
1Ch	Eh	MCANSS_IES	MCAN Subsystem Interrupt Enable Status		Go
20h	10h	MCANSS_EOI	MCAN Subsystem End of Interrupt		Go
24h	12h	MCANSS_EXT_TS_PRESCALER	MCAN Subsystem External Timestamp Prescaler 0		Go
28h	14h	MCANSS_EXT_TS_UNSERVICED_INTR_CNTR	MCAN Subsystem External Timestamp Unserviced Interrupts Counter		Go

Complex bit access types are encoded to fit into small table cells. Table 45-17 shows the codes that are used for access types in this section.

Table 45-17. MCANSS_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 45-17. MCANSS_REGS Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

45.7.3.1 MCANSS_PID Register (Offset (x8) = 0h, Offset (x16) = 0h) [Reset = 68E03901h]

MCANSS_PID is shown in Figure 45-25 and described in Table 45-18.

Return to the Summary Table.

MCAN Subsystem Revision Register

Figure 45-25. MCANSS_PID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SCHEME		RESERVED		MODULE_ID											
R-1h		R-2h		R-8E0h											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					MAJOR			RESERVED		MINOR					
R-7h					R-1h			R-0h		R-1h					

Table 45-18. MCANSS_PID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	SCHEME	R	1h	PID Register Scheme Reset type: SYSRSn
29-28	RESERVED	R	2h	Reserved
27-16	MODULE_ID	R	8E0h	Module Identification Number Reset type: SYSRSn
15-11	RESERVED	R	7h	Reserved
10-8	MAJOR	R	1h	Major Revision of the MCAN Subsystem Reset type: SYSRSn
7-6	RESERVED	R	0h	Reserved
5-0	MINOR	R	1h	Minor Revision of the MCAN Subsystem Reset type: SYSRSn

45.7.3.2 MCANSS_CTRL Register (Offset (x8) = 4h, Offset (x16) = 2h) [Reset = 8h]

MCANSS_CTRL is shown in Figure 45-26 and described in Table 45-19.

Return to the Summary Table.

MCAN Subsystem Control Register

Figure 45-26. MCANSS_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	EXT_TS_CNTR_EN	AUTOWAKEUP	WAKEUPREQEN	DBGSUSP_FREE	RESERVED		
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h	R-0h		

Table 45-19. MCANSS_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	EXT_TS_CNTR_EN	R/W	0h	External Timestamp Counter Enable. 0 External timestamp counter disabled 1 External timestamp counter enabled Reset type: SYSRSn
5	AUTOWAKEUP	R/W	0h	Automatic Wakeup Enable. Enables the MCANSS to automatically clear the MCAN CCCR.INIT bit, fully waking the MCAN up, on an enabled wakeup request. 0 Disable the automatic write to CCCR.INIT 1 Enable the automatic write to CCCR.INIT Reset type: SYSRSn
4	WAKEUPREQEN	R/W	0h	Wakeup Request Enable. Enables the MCANSS to wakeup on CAN RXD activity. 0 Disable wakeup request 1 Enables wakeup request Reset type: SYSRSn
3	DBGSUSP_FREE	R/W	1h	Debug Suspend Free Bit. Enables debug suspend. 0 Disable debug suspend 1 Enable debug suspend Reset type: SYSRSn
2-0	RESERVED	R	0h	Reserved

45.7.3.3 MCANSS_STAT Register (Offset (x8) = 8h, Offset (x16) = 4h) [Reset = X]

MCANSS_STAT is shown in Figure 45-27 and described in Table 45-20.

Return to the Summary Table.

MCAN Subsystem Status Register

Figure 45-27. MCANSS_STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					ENABLE_FDOE	MEM_INIT_DONE	RESET
R-0h					R-X	R-0h	R-0h

Table 45-20. MCANSS_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2	ENABLE_FDOE	R	X	Flexible Datarate Operation Enable. Determines whether CAN FD operation may be enabled via the MCAN core CCCR.FDOE bit (bit 8) or if only standard CAN operation is possible with this instance of the MCAN. 0 MCAN is only capable of standard CAN communication 1 MCAN may be configured to perform CAN FD communication Reset type: SYSRSn
1	MEM_INIT_DONE	R	0h	Memory Initialization Done. 0 Message RAM initialization is in progress 1 Message RAM is initialized for use Reset type: SYSRSn
0	RESET	R	0h	Soft Reset Status. 0 Not in reset 1 Reset is in progress Reset type: SYSRSn

45.7.3.4 MCANSS_ICS Register (Offset (x8) = Ch, Offset (x16) = 6h) [Reset = 0h]

MCANSS_ICS is shown in Figure 45-28 and described in Table 45-21.

Return to the Summary Table.

MCAN Subsystem Interrupt Clear Shadow Register

Figure 45-28. MCANSS_ICS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EXT_TS_CNTR_OVFL
R-0h							R-0/W1C-0h

Table 45-21. MCANSS_ICS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EXT_TS_CNTR_OVFL	R-0/W1C	0h	External Timestamp Counter Overflow Interrupt Status Clear. Reads always return a 0. 0 Write of '0' has no effect 1 Write of '1' clears the MCANSS_IRS.EXT_TS_CNTR_OVFL bit Reset type: SYSRSn

45.7.3.5 MCANSS_IRS Register (Offset (x8) = 10h, Offset (x16) = 8h) [Reset = 0h]

MCANSS_IRS is shown in Figure 45-29 and described in Table 45-22.

Return to the Summary Table.

MCAN Subsystem Interrupt Raw Satus Register

Figure 45-29. MCANSS_IRS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EXT_TS_CNTR_OVFL
R-0h							R/W1S-0h

Table 45-22. MCANSS_IRS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EXT_TS_CNTR_OVFL	R/W1S	0h	External Timestamp Counter Overflow Interrupt Status. This bit is set by HW or by a SW write of '1'. To clear, use the MCANSS_ICS.EXT_TS_CNTR_OVFL bit. 0 External timestamp counter has not overflowed 1 External timestamp counter has overflowed When this bit is set to '1' by HW or SW, the MCANSS_EXT_TS_UNSERVICED_INTR_CNTR.EXT_TS_INTR_CNTR bit field will increment by 1. Reset type: SYSRSn

45.7.3.6 MCANSS_IECS Register (Offset (x8) = 14h, Offset (x16) = Ah) [Reset = 0h]

MCANSS_IECS is shown in Figure 45-30 and described in Table 45-23.

Return to the Summary Table.

MCAN Subsystem Interrupt Enable Clear Shadow Register

Figure 45-30. MCANSS_IECS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EXT_TS_CNTR_OVFL
R-0h							R-0/W1C-0h

Table 45-23. MCANSS_IECS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EXT_TS_CNTR_OVFL	R-0/W1C	0h	External Timestamp Counter Overflow Interrupt Enable Clear. Reads always return a 0. 0 Write of '0' has no effect 1 Write of '1' clears the MCANSS_IES.EXT_TS_CNTR_OVFL bit Reset type: SYSRSn

45.7.3.7 MCANSS_IE Register (Offset (x8) = 18h, Offset (x16) = Ch) [Reset = 0h]

MCANSS_IE is shown in Figure 45-31 and described in Table 45-24.

Return to the Summary Table.

MCAN Subsystem Interrupt Enable Register

Figure 45-31. MCANSS_IE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EXT_TS_CNTR_OVFL
R-0h							R/W1S-0h

Table 45-24. MCANSS_IE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EXT_TS_CNTR_OVFL	R/W1S	0h	External Timestamp Counter Overflow Interrupt Enable. A write of '0' has no effect. A write of '1' sets the MCANSS_IES.EXT_TS_CNTR_OVFL bit. Reset type: SYSRSn

45.7.3.8 MCANSS_IES Register (Offset (x8) = 1Ch, Offset (x16) = Eh) [Reset = 0h]

MCANSS_IES is shown in Figure 45-32 and described in Table 45-25.

Return to the Summary Table.

MCAN Subsystem Interrupt Enable Status

Figure 45-32. MCANSS_IES Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EXT_TS_CNTR_OVFL
R-0h							R-0h

Table 45-25. MCANSS_IES Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EXT_TS_CNTR_OVFL	R	0h	External Timestamp Counter Overflow Interrupt Enable Status. To set, use the CANSS_IE.EXT_TS_CNTR_OVFL bit. To clear, use the MCANSS_IECS.EXT_TS_CNTR_OVFL bit. 0 External timestamp counter overflow interrupt is not enabled 1 External timestamp counter overflow interrupt is enabled Reset type: SYSRSn

45.7.3.9 MCANSS_EOI Register (Offset (x8) = 20h, Offset (x16) = 10h) [Reset = 0h]

MCANSS_EOI is shown in Figure 45-33 and described in Table 45-26.

Return to the Summary Table.

MCAN Subsystem End of Interrupt

Figure 45-33. MCANSS_EOI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								EOI							
R-0h																								R-0/W1S-0h							

Table 45-26. MCANSS_EOI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	EOI	R-0/W1S	0h	End of Interrupt. A write to this register will clear the associated interrupt. If the unserviced interrupt counter is > 1, another interrupt is generated. 0x00 External TS Interrupt is cleared 0x01 MCAN[0] interrupt is cleared 0x02 MCAN[1] interrupt is cleared Other writes are ignored. Reset type: SYSRSn

45.7.3.10 MCANSS_EXT_TS_PRESCALER Register (Offset (x8) = 24h, Offset (x16) = 12h) [Reset = 0h]

MCANSS_EXT_TS_PRESCALER is shown in Figure 45-34 and described in Table 45-27.

Return to the Summary Table.

MCAN Subsystem External Timestamp Prescaler 0

Figure 45-34. MCANSS_EXT_TS_PRESCALER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								PRESCALER																							
R-0h								R/W-0h																							

Table 45-27. MCANSS_EXT_TS_PRESCALER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-0	PRESCALER	R/W	0h	External Timestamp Prescaler Reload Value. The external timestamp count rate is the host (system) clock rate divided by this value, except in the case of 0. A zero value in this bit field will act identically to a value of 0x000001. Reset type: SYSRSn

45.7.3.11 MCANSS_EXT_TS_UNSERVICED_INTR_CNTR Register (Offset (x8) = 28h, Offset (x16) = 14h) [Reset = 0h]

MCANSS_EXT_TS_UNSERVICED_INTR_CNTR is shown in Figure 45-35 and described in Table 45-28.

Return to the Summary Table.

MCAN Subsystem External Timestamp Unserviced Interrupts Counter

Figure 45-35. MCANSS_EXT_TS_UNSERVICED_INTR_CNTR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			EXT_TS_INTR_CNTR				
R-0h			R-0h				

Table 45-28. MCANSS_EXT_TS_UNSERVICED_INTR_CNTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4-0	EXT_TS_INTR_CNTR	R	0h	External Timestamp Counter Unserviced Rollover Interrupts. If this value is > 1, an MCANSS_EOI write of '1' to bit 0 will issue another interrupt. The status of this bit field is affected by the MCANSS_IRS.EXT_TS_CNTR_OVFL bit field. Reset type: SYSRSn

45.7.4 MCAN_REGS Registers

Table 45-29 lists the memory-mapped registers for the MCAN_REGS registers. All register offset addresses not listed in Table 45-29 should be considered as reserved locations and the register contents should not be modified.

Table 45-29. MCAN_REGS Registers

Offset (x8)	Offset (x16)	Acronym	Register Name	Write Protection	Section
0h	0h	MCAN_CREL	MCAN Core Release Register		Go
4h	2h	MCAN_ENDN	MCAN Endian Register		Go
Ch	6h	MCAN_DBTP	MCAN Data Bit Timing and Prescaler Register		Go
10h	8h	MCAN_TEST	MCAN Test Register		Go
14h	Ah	MCAN_RWD	MCAN RAM Watchdog		Go
18h	Ch	MCAN_CCCR	MCAN CC Control Register		Go
1Ch	Eh	MCAN_NBTP	MCAN Nominal Bit Timing and Prescaler Register		Go
20h	10h	MCAN_TSCC	MCAN Timestamp Counter Configuration		Go
24h	12h	MCAN_TSCV	MCAN Timestamp Counter Value		Go
28h	14h	MCAN_TOCC	MCAN Timeout Counter Configuration		Go
2Ch	16h	MCAN_TOCV	MCAN Timeout Counter Value		Go
40h	20h	MCAN_ECR	MCAN Error Counter Register		Go
44h	22h	MCAN_PSR	MCAN Protocol Status Register		Go
48h	24h	MCAN_TDCR	MCAN Transmitter Delay Compensation Register		Go
50h	28h	MCAN_IR	MCAN Interrupt Register		Go
54h	2Ah	MCAN_IE	MCAN Interrupt Enable		Go
58h	2Ch	MCAN_ILS	MCAN Interrupt Line Select		Go
5Ch	2Eh	MCAN_ILE	MCAN Interrupt Line Enable		Go
80h	40h	MCAN_GFC	MCAN Global Filter Configuration		Go
84h	42h	MCAN_SIDFC	MCAN Standard ID Filter Configuration		Go
88h	44h	MCAN_XIDFC	MCAN Extended ID Filter Configuration		Go
90h	48h	MCAN_XIDAM	MCAN Extended ID and Mask		Go
94h	4Ah	MCAN_HPMS	MCAN High Priority Message Status		Go
98h	4Ch	MCAN_NDAT1	MCAN New Data 1		Go
9Ch	4Eh	MCAN_NDAT2	MCAN New Data 2		Go
A0h	50h	MCAN_RXF0C	MCAN Rx FIFO 0 Configuration		Go
A4h	52h	MCAN_RXF0S	MCAN Rx FIFO 0 Status		Go
A8h	54h	MCAN_RXF0A	MCAN Rx FIFO 0 Acknowledge		Go
ACh	56h	MCAN_RXBC	MCAN Rx Buffer Configuration		Go
B0h	58h	MCAN_RXF1C	MCAN Rx FIFO 1 Configuration		Go
B4h	5Ah	MCAN_RXF1S	MCAN Rx FIFO 1 Status		Go
B8h	5Ch	MCAN_RXF1A	MCAN Rx FIFO 1 Acknowledge		Go
BCh	5Eh	MCAN_RXESC	MCAN Rx Buffer / FIFO Element Size Configuration		Go
C0h	60h	MCAN_TXBC	MCAN Tx Buffer Configuration		Go
C4h	62h	MCAN_TXFQS	MCAN Tx FIFO / Queue Status		Go
C8h	64h	MCAN_TXESC	MCAN Tx Buffer Element Size Configuration		Go
CCh	66h	MCAN_TXBRP	MCAN Tx Buffer Request Pending		Go

Table 45-29. MCAN_REGS Registers (continued)

Offset (x8)	Offset (x16)	Acronym	Register Name	Write Protection	Section
D0h	68h	MCAN_TXBAR	MCAN Tx Buffer Add Request		Go
D4h	6Ah	MCAN_TXBCR	MCAN Tx Buffer Cancellation Request		Go
D8h	6Ch	MCAN_TXBTO	MCAN Tx Buffer Transmission Occurred		Go
DCh	6Eh	MCAN_TXBCF	MCAN Tx Buffer Cancellation Finished		Go
E0h	70h	MCAN_TXBTIE	MCAN Tx Buffer Transmission Interrupt Enable		Go
E4h	72h	MCAN_TXBCIE	MCAN Tx Buffer Cancellation Finished Interrupt Enable		Go
F0h	78h	MCAN_TXEFC	MCAN Tx Event FIFO Configuration		Go
F4h	7Ah	MCAN_TXEFS	MCAN Tx Event FIFO Status		Go
F8h	7Ch	MCAN_TXEFA	MCAN Tx Event FIFO Acknowledge		Go

Complex bit access types are encoded to fit into small table cells. Table 45-30 shows the codes that are used for access types in this section.

Table 45-30. MCAN_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
RC	R C	Read to Clear
RS	R S	Read to Set
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1SQ	W 1S Q	Write 1 to set Qualified. A condition must be met for this operation to occur.
WQ	W Q	Write Qualified. A condition must be met for this operation to occur.
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

45.7.4.1 MCAN_CREL Register (Offset (x8) = 0h, Offset (x16) = 0h) [Reset = 32270530h]

MCAN_CREL is shown in Figure 45-36 and described in Table 45-31.

Return to the Summary Table.

MCAN Core Release Register

Figure 45-36. MCAN_CREL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REL				STEP				SUBSTEP				YEAR			
R-3h				R-2h				R-2h				R-7h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MON								DAY							
R-5h								R-30h							

Table 45-31. MCAN_CREL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	REL	R	3h	Core Release. One digit, BCD-coded. Reset type: SYSRSn
27-24	STEP	R	2h	Step of Core Release. One digit, BCD-coded. Reset type: SYSRSn
23-20	SUBSTEP	R	2h	Sub-Step of Core Release. One digit, BCD-coded. Reset type: SYSRSn
19-16	YEAR	R	7h	Time Stamp Year. One digit, BCD-coded. Reset type: SYSRSn
15-8	MON	R	5h	Time Stamp Month. Two digits, BCD-coded. Reset type: SYSRSn
7-0	DAY	R	30h	Time Stamp Day. Two digits, BCD-coded. Reset type: SYSRSn

45.7.4.2 MCAN_ENDN Register (Offset (x8) = 4h, Offset (x16) = 2h) [Reset = 87654321h]

MCAN_ENDN is shown in Figure 45-37 and described in Table 45-32.

Return to the Summary Table.

MCAN Endian Register

Figure 45-37. MCAN_ENDN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETV																															
R-87654321h																															

Table 45-32. MCAN_ENDN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ETV	R	87654321h	Endianess Test Value. Reading the constant value maintained in this register allows software to determine the endianess of the host CPU. Reset type: SYSRSn

45.7.4.3 MCAN_DBTP Register (Offset (x8) = Ch, Offset (x16) = 6h) [Reset = A33h]

MCAN_DBTP is shown in Figure 45-38 and described in Table 45-33.

Return to the Summary Table.

This register is only writable if bits CCCR.CCE and CCCR.INIT are set. The CAN bit time may be programed in the range of 4 to 49 time quanta. The CAN time quantum may be programmed in the range of 1 to 32 m_can_cclk periods. tq = (DBRP + 1) mtq.
DTSEG1 is the sum of Prop_Seg and Phase_Seg1. DTSEG2 is Phase_Seg2.
Therefore the length of the bit time is (programmed values) [DTSEG1 + DTSEG2 + 3] tq or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] tq.
The Information Processing Time (IPT) is zero, meaning the data for the next bit is available at the first clock edge after the sample point.

Figure 45-38. MCAN_DBTP Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
TDC	RESERVED		DBRP				
R/WQ-0h	R-0h		R/WQ-0h				
15	14	13	12	11	10	9	8
RESERVED			DTSEG1				
R-0h			R/WQ-Ah				
7	6	5	4	3	2	1	0
DTSEG2				DSJW			
R/WQ-3h				R/WQ-3h			

Table 45-33. MCAN_DBTP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23	TDC	R/WQ	0h	Transmitter Delay Compensation 0 Transmitter Delay Compensation disabled 1 Transmitter Delay Compensation enabled +I107 Reset type: SYSRSn
22-21	RESERVED	R	0h	Reserved
20-16	DBRP	R/WQ	0h	Data Bit Rate Prescaler. The value by which the oscillator frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Bit Rate Prescaler are 0 to 31. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
15-13	RESERVED	R	0h	Reserved
12-8	DTSEG1	R/WQ	Ah	Data Time Segment Before Sample Point. Valid values are 0 to 31. The actual interpretation by the hardware of this value is such that one more than the programmed value is used. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

Table 45-33. MCAN_DBTP Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-4	DTSEG2	R/WQ	3h	Data Time Segment After Sample Point. Valid values are 0 to 15. The actual interpretation by the hardware of this value is such that one more than the programmed value is used. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
3-0	DSJW	R/WQ	3h	Data Resynchronization Jump Width. Valid values are 0 to 15. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

45.7.4.4 MCAN_TEST Register (Offset (x8) = 10h, Offset (x16) = 8h) [Reset = X]

MCAN_TEST is shown in Figure 45-39 and described in Table 45-34.

Return to the Summary Table.

Write access to the Test Register has to be enabled by setting bit CCCR.TEST to '1'. All Test Register functions are set to their reset values when bit CCCR.TEST is reset.
Loop Back Mode and software control of the internal CAN TX pin are hardware test modes. Programming of TX != "00" may disturb the message transfer on the CAN bus.

Figure 45-39. MCAN_TEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RX	TX		LBCK	RESERVED			
R-X	R/WQ-0h		R/WQ-0h	R-0h			

Table 45-34. MCAN_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	RX	R	X	Receive Pin. Monitors the actual value of the CAN receive pin. 0 The CAN bus is dominant (CAN RX pin = '0') 1 The CAN bus is recessive (CAN RX pin = '1') Reset type: SYSRSn
6-5	TX	R/WQ	0h	Control of Transmit Pin 00 CAN TX pin controlled by the CAN Core, updated at the end of the CAN bit time 01 Sample Point can be monitored at CAN TX pin 10 Dominant ('0') level at CAN TX pin 11 Recessive ('1') at CAN TX pin Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
4	LBCK	R/WQ	0h	Loop Back Mode 0 Reset value, Loop Back Mode is disabled 1 Loop Back Mode is enabled Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

45.7.4.5 MCAN_RWD Register (Offset (x8) = 14h, Offset (x16) = Ah) [Reset = 0h]

MCAN_RWD is shown in Figure 45-40 and described in Table 45-35.

Return to the Summary Table.

MCAN RAM Watchdog

Figure 45-40. MCAN_RWD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																WDV								WDC							
R-0h																R-0h								R/WQ-0h							

Table 45-35. MCAN_RWD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	WDV	R	0h	Watchdog Value. Actual Message RAM Watchdog Counter Value. The RAM Watchdog monitors the READY output of the Message RAM. A Message RAM access via the MCAN's Generic Controller Interface starts the Message RAM Watchdog Counter with the value configured by the WDC field. The counter is reloaded with WDC when the Message RAM signals successful completion by activating its READY output. In case there is no response from the Message RAM until the counter has counted down to zero, the counter stops and interrupt flag MCAN_IR.WDI is set. The RAM Watchdog Counter is clocked by the host (system) clock. Reset type: SYSRSn
7-0	WDC	R/WQ	0h	Watchdog Configuration. Start value of the Message RAM Watchdog Counter. With the reset value of "00" the counter is disabled. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

45.7.4.6 MCAN_CCCR Register (Offset (x8) = 18h, Offset (x16) = Ch) [Reset = 1h]

MCAN_CCCR is shown in Figure 45-41 and described in Table 45-36.

Return to the Summary Table.

MCAN CC Control Register

Figure 45-41. MCAN_CCCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
NISO	TXP	EFBI	PXHD	RESERVED		BRSE	FDOE
R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R-0h		R/WQ-0h	R/WQ-0h
7	6	5	4	3	2	1	0
TEST	DAR	MON	CSR	CSA	ASM	CCE	INIT
R/W1SQ-0h	R/WQ-0h	R/W1SQ-0h	R/W-0h	R-0h	R/W1SQ-0h	R/WQ-0h	R/W-1h

Table 45-36. MCAN_CCCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	NISO	R/WQ	0h	Non ISO Operation. If this bit is set, the MCAN uses the CAN FD frame format as specified by the Bosch CAN FD Specification V1.0. 0 CAN FD frame format according to ISO 11898-1:2015 1 CAN FD frame format according to Bosch CAN FD Specification V1.0 Reset type: SYSRSn
14	TXP	R/WQ	0h	Transmit Pause. If this bit is set, the MCAN pauses for two CAN bit times before starting the next transmission after itself has successfully transmitted a frame. 0 Transmit pause disabled 1 Transmit pause enabled Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
13	EFBI	R/WQ	0h	Edge Filtering during Bus Integration 0 Edge filtering disabled 1 Two consecutive dominant tq required to detect an edge for hard synchronization Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
12	PXHD	R/WQ	0h	Protocol Exception Handling Disable 0 Protocol exception handling enabled 1 Protocol exception handling disabled Note: When protocol exception handling is disabled, the MCAN will transmit an error frame when it detects a protocol exception condition. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
11-10	RESERVED	R	0h	Reserved

Table 45-36. MCAN_CCCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	BRSE	R/WQ	0h	Bit Rate Switch Enable 0 Bit rate switching for transmissions disabled 1 Bit rate switching for transmissions enabled Note: When CAN FD operation is disabled FDOE = '0', BRSE is not evaluated. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
8	FDOE	R/WQ	0h	Flexible Datarate Operation Enable 0 FD operation disabled 1 FD operation enabled Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
7	TEST	R/W1SQ	0h	Test Mode Enable 0 Normal operation, register TEST holds reset values 1 Test Mode, write access to register TEST enabled Qualified Write 1 to Set is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
6	DAR	R/WQ	0h	Disable Automatic Retransmission 0 Automatic retransmission of messages not transmitted successfully enabled 1 Automatic retransmission disabled Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
5	MON	R/W1SQ	0h	Bus Monitoring Mode. Bit MON can only be set by SW when both CCE and INIT are set to '1'. The bit can be reset by SW at any time. 0 Bus Monitoring Mode is disabled 1 Bus Monitoring Mode is enabled Qualified Write 1 to Set is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
4	CSR	R/W	0h	Clock Stop Request 0 No clock stop is requested 1 Clock stop requested. When clock stop is requested, first INIT and then CSA will be set after all pending transfer requests have been completed and the CAN bus reached idle. Reset type: SYSRSn
3	CSA	R	0h	Clock Stop Acknowledge 0 No clock stop acknowledged 1 MCAN may be set in power down by stopping the Host and CAN clocks Reset type: SYSRSn
2	ASM	R/W1SQ	0h	Restricted Operation Mode. Bit ASM can only be set by SW when both CCE and INIT are set to '1'. The bit can be reset by SW at any time. 0 Normal CAN operation 1 Restricted Operation Mode active Qualified Write 1 to Set is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
1	CCE	R/WQ	0h	Configuration Change Enable 0 The CPU has no write access to the protected configuration registers 1 The CPU has write access to the protected configuration registers (while CCCR.INIT = '1') Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

Table 45-36. MCAN_CCCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INIT	R/W	1h	Initialization 0 Normal Operation 1 Initialization is started Note: Due to the synchronization mechanism between the two clock domains, there may be a delay until the value written to INIT can be read back. Therefore the programmer has to assure that the previous value written to INIT has been accepted by reading INIT before setting INIT to a new value. Reset type: SYSRSn

45.7.4.7 MCAN_NBTP Register (Offset (x8) = 1Ch, Offset (x16) = Eh) [Reset = 06000A03h]

MCAN_NBTP is shown in Figure 45-42 and described in Table 45-37.

Return to the Summary Table.

This register is only writable if bits CCCR.CCE and CCCR.INIT are set. The CAN bit time may be programed in the range of 4 to 385 time quanta. The CAN time quantum may be programmed in the range of 1 to 512 m_can_cclk periods. tq = (NBRP + 1) mtq.
NTSEG1 is the sum of Prop_Seg and Phase_Seg1. NTSEG2 is Phase_Seg2.
Therefore the length of the bit time is (programmed values) [NTSEG1 + NTSEG2 + 3] tq or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] tq.
The Information Processing Time (IPT) is zero, meaning the data for the next bit is available at the first clock edge after the sample point.
Note: With a CAN clock of 8 MHz, the reset value of 0x06000A03 configures the MCAN for a bit rate of 500 kBit/s.

Figure 45-42. MCAN_NBTP Register

31	30	29	28	27	26	25	24
NSJW							NBRP
R/WQ-3h							R/WQ-0h
23	22	21	20	19	18	17	16
NBRP							
R/WQ-0h							
15	14	13	12	11	10	9	8
NTSEG1							
R/WQ-Ah							
7	6	5	4	3	2	1	0
RESERVED	NTSEG2						
R-0h	R/WQ-3h						

Table 45-37. MCAN_NBTP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	NSJW	R/WQ	3h	Nominal (Re)Synchronization Jump Width. Valid values are 0 to 127. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
24-16	NBRP	R/WQ	0h	Nominal Bit Rate Prescaler. The value by which the oscillator frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Bit Rate Prescaler are 0 to 511. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
15-8	NTSEG1	R/WQ	Ah	Nominal Time Segment Before Sample Point. Valid values are 1 to 255. The actual interpretation by the hardware of this value is such that one more than the programmed value is used. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved

Table 45-37. MCAN_NBTP Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-0	NTSEG2	R/WQ	3h	Nominal Time Segment After Sample Point. Valid values are 1 to 127. The actual interpretation by the hardware of this value is such that one more than the programmed value is used. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

45.7.4.8 MCAN_TSCC Register (Offset (x8) = 20h, Offset (x16) = 10h) [Reset = 0h]

MCAN_TSCC is shown in Figure 45-43 and described in Table 45-38.

Return to the Summary Table.

MCAN Timestamp Counter Configuration

Figure 45-43. MCAN_TSCC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED												TCP			
R-0h												R/WQ-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														TSS	
R-0h														R/WQ-0h	

Table 45-38. MCAN_TSCC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-16	TCP	R/WQ	0h	Timestamp Counter Prescaler. Configures the timestamp and timeout counters time unit in multiples of CAN bit times. Valid values are 0 to 15. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used. Note: With CAN FD an external counter is required for timestamp generation (TSS = "10"). Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
15-2	RESERVED	R	0h	Reserved
1-0	TSS	R/WQ	0h	Timestamp Select 00 Timestamp counter value always 0x0000 01 Timestamp counter value incremented according to TCP 10 External timestamp counter value used 11 Same as "00" Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

45.7.4.9 MCAN_TSCV Register (Offset (x8) = 24h, Offset (x16) = 12h) [Reset = 0h]

MCAN_TSCV is shown in Figure 45-44 and described in Table 45-39.

Return to the Summary Table.

MCAN Timestamp Counter Value

Figure 45-44. MCAN_TSCV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																TSC															
R-0h																R/W-0h															

Table 45-39. MCAN_TSCV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	TSC	R/W	0h	Timestamp Counter. The internal/external Timestamp Counter value is captured on start of frame (both Rx and Tx). When TSCC.TSS = "01", the Timestamp Counter is incremented in multiples of CAN bit times, [1...16], depending on the configuration of TSCC.TCP. A wrap around sets interrupt flag IR.TSW. Write access resets the counter to zero. When TSCC.TSS = "10", TSC reflects the External Timestamp Counter value, and a write access has no impact. Note: A "wrap around" is a change of the Timestamp Counter value from non-zero to zero not caused by write access to MCAN_TSCV. Reset type: SYSRSn

45.7.4.10 MCAN_TOCC Register (Offset (x8) = 28h, Offset (x16) = 14h) [Reset = FFFF0000h]

MCAN_TOCC is shown in Figure 45-45 and described in Table 45-40.

Return to the Summary Table.

MCAN Timeout Counter Configuration

Figure 45-45. MCAN_TOCC Register

31	30	29	28	27	26	25	24
TOP							
R/WQ-FFFFh							
23	22	21	20	19	18	17	16
TOP							
R/WQ-FFFFh							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					TOS		ETOC
R-0h					R/WQ-0h		R/WQ-0h

Table 45-40. MCAN_TOCC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	TOP	R/WQ	FFFFh	Timeout Period. Start value of the Timeout Counter (down-counter). Configures the Timeout Period. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
15-3	RESERVED	R	0h	Reserved
2-1	TOS	R/WQ	0h	Timeout Select. When operating in Continuous mode, a write to TOCV presets the counter to the value configured by TOCC.TOP and continues down-counting. When the Timeout Counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the value configured by TOCC.TOP. Down-counting is started when the first FIFO element is stored. 00 Continuous operation 01 Timeout controlled by Tx Event FIFO 10 Timeout controlled by Rx FIFO 0 11 Timeout controlled by Rx FIFO 1 Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
0	ETOC	R/WQ	0h	Enable Timeout Counter 0 Timeout Counter disabled 1 Timeout Counter enabled Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

45.7.4.11 MCAN_TOCV Register (Offset (x8) = 2Ch, Offset (x16) = 16h) [Reset = FFFFh]

MCAN_TOCV is shown in Figure 45-46 and described in Table 45-41.

Return to the Summary Table.

MCAN Timeout Counter Value

Figure 45-46. MCAN_TOCV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																TOC															
R-0h																R/W-FFFFh															

Table 45-41. MCAN_TOCV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	TOC	R/W	FFFFh	Timeout Counter. The Timeout Counter is decremented in multiples of CAN bit times, [1...16], depending on the configuration of TSCC.TCP. When decremented to zero, interrupt flag IR.TOO is set and the Timeout Counter is stopped. Start and reset/restart conditions are configured via TOCC.TOS. Reset type: SYSRSn

45.7.4.12 MCAN_ECR Register (Offset (x8) = 40h, Offset (x16) = 20h) [Reset = 0h]

MCAN_ECR is shown in Figure 45-47 and described in Table 45-42.

Return to the Summary Table.

MCAN Error Counter Register

Figure 45-47. MCAN_ECR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								CEL							
R-0h								RC-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RP	REC							TEC							
R-0h	R-0h							R-0h							

Table 45-42. MCAN_ECR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	CEL	RC	0h	CAN Error Logging. The counter is incremented each time when a CAN protocol error causes the Transmit Error Counter or the Receive Error Counter to be incremented. It is reset by read access to CEL. The counter stops at 0xFF the next increment of TEC or REC sets interrupt flag IR.ELO. Note: When CCCR.ASM is set, the CAN protocol controller does not increment TEC and REC when a CAN protocol error is detected, but CEL is still incremented. Reset type: SYSRSn
15	RP	R	0h	Receive Error Passive 0 The Receive Error Counter is below the error passive level of 128 1 The Receive Error Counter has reached the error passive level of 128 Reset type: SYSRSn
14-8	REC	R	0h	Receive Error Counter. Actual state of the Receive Error Counter, values between 0 and 127. Note: When CCCR.ASM is set, the CAN protocol controller does not increment TEC and REC when a CAN protocol error is detected, but CEL is still incremented. Reset type: SYSRSn
7-0	TEC	R	0h	Transmit Error Counter. Actual state of the Transmit Error Counter, values between 0 and 255. Note: When CCCR.ASM is set, the CAN protocol controller does not increment TEC and REC when a CAN protocol error is detected, but CEL is still incremented. Reset type: SYSRSn

45.7.4.13 MCAN_PSR Register (Offset (x8) = 44h, Offset (x16) = 22h) [Reset = 707h]

MCAN_PSR is shown in Figure 45-48 and described in Table 45-43.

Return to the Summary Table.

MCAN Protocol Status Register

Figure 45-48. MCAN_PSR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED	TDCV						
R-0h	R-0h						
15	14	13	12	11	10	9	8
RESERVED	PXE	RFDF	RBRS	RESI	DLEC		
R-0h	RC-0h	RC-0h	RC-0h	RC-0h	RS-7h		
7	6	5	4	3	2	1	0
BO	EW	EP	ACT		LEC		
R-0h	R-0h	R-0h	R-0h		RS-7h		

Table 45-43. MCAN_PSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22-16	TDCV	R	0h	Transmitter Delay Compensation Value. Position of the secondary sample point, defined by the sum of the measured delay from the internal CAN TX signal to the internal CAN RX signal and TDCR.TDCO. The SSP position is, in the data phase, the number of mtq between the start of the transmitted bit and the secondary sample point. Valid values are 0 to 127 mtq. Reset type: SYSRSn
15	RESERVED	R	0h	Reserved
14	PXE	RC	0h	Protocol Exception Event 0 No protocol exception event occurred since last read access 1 Protocol exception event occurred Reset type: SYSRSn
13	RFDF	RC	0h	Received a CAN FD Message. This bit is set independent of acceptance filtering. 0 Since this bit was reset by the CPU, no CAN FD message has been received 1 Message in CAN FD format with FDF flag set has been received Reset type: SYSRSn
12	RBRS	RC	0h	BRS Flag of Last Received CAN FD Message. This bit is set together with RFDF, independent of acceptance filtering. 0 Last received CAN FD message did not have its BRS flag set 1 Last received CAN FD message had its BRS flag set Reset type: SYSRSn
11	RESI	RC	0h	ESI Flag of Last Received CAN FD Message. This bit is set together with RFDF, independent of acceptance filtering. 0 Last received CAN FD message did not have its ESI flag set 1 Last received CAN FD message had its ESI flag set Reset type: SYSRSn

Table 45-43. MCAN_PSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10-8	DLEC	RS	7h	Data Phase Last Error Code. Type of last error that occurred in the data phase of a CAN FD format frame with its BRS flag set. Coding is the same as for LEC. This field will be cleared to zero when a CAN FD format frame with its BRS flag set has been transferred (reception or transmission) without error. Reset type: SYSRSn
7	BO	R	0h	Bus_Off Status 0 The M_CAN is not Bus_Off 1 The M_CAN is in Bus_Off state Reset type: SYSRSn
6	EW	R	0h	Warning Status 0 Both error counters are below the Error_Warning limit of 96 1 At least one of error counter has reached the Error_Warning limit of 96 Reset type: SYSRSn
5	EP	R	0h	Error Passive 0 The M_CAN is in the Error_Active state. It normally takes part in bus communication and sends an active error flag when an error has been detected 1 The M_CAN is in the Error_Passive state Reset type: SYSRSn
4-3	ACT	R	0h	Node Activity. Monitors the module's CAN communication state. 00 Synchronizing - node is synchronizing on CAN communication 01 Idle - node is neither receiver nor transmitter 10 Receiver - node is operating as receiver 11 Transmitter - node is operating as transmitter Note: ACT is set to "00" by a Protocol Exception Event. Reset type: SYSRSn

Table 45-43. MCAN_PSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	LEC	RS	7h	Last Error Code. The LEC indicates the type of the last error to occur on the CAN bus. This field will be cleared to '0' when a message has been transferred (reception or transmission) without error. 0 No Error: No error occurred since LEC has been reset by successful reception or transmission. 1 Stuff Error: More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed. 2 Form Error: A fixed format part of a received frame has the wrong format. 3 AckError: The message transmitted by the MCAN was not acknowledged by another node. 4 Bit1Error: During the transmission of a message (with the exception of the arbitration field), the device wanted to send a recessive level (bit of logical value '1'), but the monitored bus value was dominant. 5 Bit0Error: During the transmission of a message (or acknowledge bit, or active error flag, or overload flag), the device wanted to send a dominant level (data or identifier bit logical value '0'), but the monitored bus value was recessive. During Bus_Off recovery this status is set each time a sequence of 11 recessive bits has been monitored. This enables the CPU to monitor the proceeding of the Bus_Off recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed). 6 CRCError: The CRC check sum of a received message was incorrect. The CRC of an incoming message does not match with the CRC calculated from the received data. 7 NoChange: Any read access to the Protocol Status Register re-initializes the LEC to '7'. When the LEC shows the value '7', no CAN bus event was detected since the last CPU read access to the Protocol Status Register. Note: When a frame in CAN FD format has reached the data phase with BRS flag set, the next CAN event (error or valid frame) will be shown in DLEC instead of LEC. An error in a fixed stuff bit of a CAN FD CRC sequence will be shown as a Form Error, not Stuff Error. Note: The Bus_Off recovery sequence (see ISO 11898-1:2015) cannot be shortened by setting or resetting CCCR.INIT. If the device goes Bus_Off, it will set CCCR.INIT of its own accord, stopping all bus activities. Once CCCR.INIT has been cleared by the CPU, the device will then wait for 129 occurrences of Bus Idle (129 * 11 consecutive recessive bits) before resuming normal operation. At the end of the Bus_Off recovery sequence, the Error Management Counters will be reset. During the waiting time after the resetting of CCCR.INIT, each time a sequence of 11 recessive bits has been monitored, a Bit0Error code is written to PSR.LEC, enabling the CPU to readily check up whether the CAN bus is stuck at dominant or continuously disturbed and to monitor the Bus_Off recovery sequence. ECR.REC is used to count these sequences. Reset type: SYSRSn

45.7.4.14 MCAN_TDCR Register (Offset (x8) = 48h, Offset (x16) = 24h) [Reset = 0h]

MCAN_TDCR is shown in Figure 45-49 and described in Table 45-44.

Return to the Summary Table.

MCAN Transmitter Delay Compensation Register

Figure 45-49. MCAN_TDCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	TDCO						
R-0h	R/WQ-0h						
7	6	5	4	3	2	1	0
RESERVED	TDCF						
R-0h	R/WQ-0h						

Table 45-44. MCAN_TDCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	RESERVED	R	0h	Reserved
14-8	TDCO	R/WQ	0h	Transmitter Delay Compensation Offset. Offset value defining the distance between the measured delay from the internal CAN TX signal to the internal CAN RX signal and the secondary sample point. Valid values are 0 to 127 mtq. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6-0	TDCF	R/WQ	0h	Transmitter Delay Compensation Filter Window Length. Defines the minimum value for the SSP position, dominant edges on the internal CAN RX signal that would result in an earlier SSP position are ignored for transmitter delay measurement. The feature is enabled when TDCF is configured to a value greater than TDCO. Valid values are 0 to 127 mtq. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

45.7.4.15 MCAN_IR Register (Offset (x8) = 50h, Offset (x16) = 28h) [Reset = 80000000h]

MCAN_IR is shown in Figure 45-50 and described in Table 45-45.

Return to the Summary Table.

The flags are set when one of the listed conditions is detected (edge-sensitive). The flags remain set until the Host clears them. Aflag is cleared by writing a '1' to the corresponding bit position. Writing a '0' has no effect. Ahard reset will clear the register. The configuration of IE controls whether an interrupt is generated. The configuration of ILS controls on which interrupt line an interrupt is signalled.

Figure 45-50. MCAN_IR Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	ARA	PED	PEA	WDI	BO	EW
R-1h	R-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
23	22	21	20	19	18	17	16
EP	ELO	BEU	RESERVED	DRX	TOO	MRAF	TSW
R/W1C-0h	R/W1C-0h	R/W1C-0h	R-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
15	14	13	12	11	10	9	8
TEFL	TEFF	TEFW	TEFN	TFE	TCF	TC	HPM
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
7	6	5	4	3	2	1	0
RF1L	RF1F	RF1W	RF1N	RF0L	RF0F	RF0W	RF0N
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

Table 45-45. MCAN_IR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	1h	Reserved
30	RESERVED	R	0h	Reserved
29	ARA	R/W1C	0h	Access to Reserved Address 0 No access to reserved address occurred 1 Access to reserved address occurred Reset type: SYSRSn
28	PED	R/W1C	0h	Protocol Error in Data Phase (Data Bit Time is used) 0 No protocol error in data phase 1 Protocol error in data phase detected (PSR.DLEC != 0,7) Reset type: SYSRSn
27	PEA	R/W1C	0h	Protocol Error in Arbitration Phase (Nominal Bit Time is used) 0 No protocol error in arbitration phase 1 Protocol error in arbitration phase detected (PSR.LEC != 0,7) Reset type: SYSRSn
26	WDI	R/W1C	0h	Watchdog Interrupt 0 No Message RAM Watchdog event occurred 1 Message RAM Watchdog event due to missing READY Reset type: SYSRSn
25	BO	R/W1C	0h	Bus_Off Status 0 Bus_Off status unchanged 1 Bus_Off status changed Reset type: SYSRSn
24	EW	R/W1C	0h	Warning Status 0 Error_Warning status unchanged 1 Error_Warning status changed Reset type: SYSRSn

Table 45-45. MCAN_IR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	EP	R/W1C	0h	Error Passive 0 Error_Passive status unchanged 1 Error_Passive status changed Reset type: SYSRSn
22	ELO	R/W1C	0h	Error Logging Overflow 0 CAN Error Logging Counter did not overflow 1 Overflow of CAN Error Logging Counter occurred Reset type: SYSRSn
21	BEU	R/W1C	0h	Bit Error Uncorrected. Message RAM bit error detected, uncorrected. This bit is set when a double bit error is detected by the ECC aggregator attached to the Message RAM. An uncorrected Message RAM bit error sets CCCR.INIT to '1'. This is done to avoid transmission of corrupted data. 0 No bit error detected when reading from Message RAM 1 Bit error detected, uncorrected (e.g. parity logic) Reset type: SYSRSn
20	RESERVED	R	0h	Reserved
19	DRX	R/W1C	0h	Message Stored to Dedicated Rx Buffer. The flag is set whenever a received message has been stored into a dedicated Rx Buffer. 0 No Rx Buffer updated 1 At least one received message stored into an Rx Buffer Reset type: SYSRSn
18	TOO	R/W1C	0h	Timeout Occurred 0 No timeout 1 Timeout reached Reset type: SYSRSn
17	MRAF	R/W1C	0h	Message RAM Access Failure. The flag is set, when the Rx Handler: - has not completed acceptance filtering or storage of an accepted message until the arbitration field of the following message has been received. In this case acceptance filtering or message storage is aborted and the Rx Handler starts processing of the following message. - was not able to write a message to the Message RAM. In this case message storage is aborted. In both cases the FIFO put index is not updated resp. the New Data flag for a dedicated Rx Buffer is not set, a partly stored message is overwritten when the next message is stored to this location. The flag is also set when the Tx Handler was not able to read a message from the Message RAM in time. In this case message transmission is aborted. In case of a Tx Handler access failure the MCAN is switched into Restricted Operation Mode. To leave Restricted Operation Mode, the Host CPU has to reset CCCR.ASM. 0 No Message RAM access failure occurred 1 Message RAM access failure occurred Reset type: SYSRSn
16	TSW	R/W1C	0h	Timestamp Wraparound 0 No timestamp counter wrap-around 1 Timestamp counter wrapped around Reset type: SYSRSn
15	TEFL	R/W1C	0h	Tx Event FIFO Element Lost 0 No Tx Event FIFO element lost 1 Tx Event FIFO element lost, also set after write attempt to Tx Event FIFO of size zero Reset type: SYSRSn
14	TEFF	R/W1C	0h	Tx Event FIFO Full 0 Tx Event FIFO not full 1 Tx Event FIFO full Reset type: SYSRSn

Table 45-45. MCAN_IR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	TEFW	R/W1C	0h	Tx Event FIFO Watermark Reached 0 Tx Event FIFO fill level below watermark 1 Tx Event FIFO fill level reached watermark Reset type: SYSRSn
12	TEFN	R/W1C	0h	Tx Event FIFO New Entry 0 Tx Event FIFO unchanged 1 Tx Handler wrote Tx Event FIFO element Reset type: SYSRSn
11	TFE	R/W1C	0h	Tx FIFO Empty 0 Tx FIFO non-empty 1 Tx FIFO empty Reset type: SYSRSn
10	TCF	R/W1C	0h	Transmission Cancellation Finished 0 No transmission cancellation finished 1 Transmission cancellation finished Reset type: SYSRSn
9	TC	R/W1C	0h	Transmission Completed 0 No transmission completed 1 Transmission completed Reset type: SYSRSn
8	HPM	R/W1C	0h	High Priority Message 0 No high priority message received 1 High priority message received Reset type: SYSRSn
7	RF1L	R/W1C	0h	Rx FIFO 1 Message Lost 0 No Rx FIFO 1 message lost 1 Rx FIFO 1 message lost, also set after write attempt to Rx FIFO 1 of size zero Reset type: SYSRSn
6	RF1F	R/W1C	0h	Rx FIFO 1 Full 0 Rx FIFO 1 not full 1 Rx FIFO 1 full Reset type: SYSRSn
5	RF1W	R/W1C	0h	Rx FIFO 1 Watermark Reached 0 Rx FIFO 1 fill level below watermark 1 Rx FIFO 1 fill level reached watermark Reset type: SYSRSn
4	RF1N	R/W1C	0h	Rx FIFO 1 New Message 0 No new message written to Rx FIFO 1 1 New message written to Rx FIFO 1 Reset type: SYSRSn
3	RF0L	R/W1C	0h	Rx FIFO 0 Message Lost 0 No Rx FIFO 0 message lost 1 Rx FIFO 0 message lost, also set after write attempt to Rx FIFO 0 of size zero Reset type: SYSRSn
2	RF0F	R/W1C	0h	Rx FIFO 0 Full 0 Rx FIFO 0 not full 1 Rx FIFO 0 full Reset type: SYSRSn
1	RF0W	R/W1C	0h	Rx FIFO 0 Watermark Reached 0 Rx FIFO 0 fill level below watermark 1 Rx FIFO 0 fill level reached watermark Reset type: SYSRSn
0	RF0N	R/W1C	0h	Rx FIFO 0 New Message 0 No new message written to Rx FIFO 0 1 New message written to Rx FIFO 0 Reset type: SYSRSn

45.7.4.16 MCAN_IE Register (Offset (x8) = 54h, Offset (x16) = 2Ah) [Reset = 0h]

MCAN_IE is shown in Figure 45-51 and described in Table 45-46.

Return to the Summary Table.

MCAN Interrupt Enable

Figure 45-51. MCAN_IE Register

31	30	29	28	27	26	25	24
RESERVED		ARAE	PEDE	PEAE	WDIE	BOE	EWE
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
EPE	ELOE	BEUE	BECE	DRXE	TOOE	MRAFE	TSWE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
TEFLE	TEFFE	TEFWE	TEFNE	TFEE	TCFE	TCE	HPME
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RF1LE	RF1FE	RF1WE	RF1NE	RF0LE	RF0FE	RF0WE	RF0NE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 45-46. MCAN_IE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29	ARAE	R/W	0h	Access to Reserved Address Enable Reset type: SYSRSn
28	PEDE	R/W	0h	Protocol Error in Data Phase Enable Reset type: SYSRSn
27	PEAE	R/W	0h	Protocol Error in Arbitration Phase Enable Reset type: SYSRSn
26	WDIE	R/W	0h	Watchdog Interrupt Enable Reset type: SYSRSn
25	BOE	R/W	0h	Bus_Off Status Enable Reset type: SYSRSn
24	EWE	R/W	0h	Warning Status Enable Reset type: SYSRSn
23	EPE	R/W	0h	Error Passive Enable Reset type: SYSRSn
22	ELOE	R/W	0h	Error Logging Overflow Enable Reset type: SYSRSn
21	BEUE	R/W	0h	Bit Error Uncorrected Enable Reset type: SYSRSn
20	BECE	R/W	0h	Bit Error Corrected Enable A separate interrupt line reserved for corrected bit errors is provided via the MCAN_ERROR_REGS. It advised for the user to use these registers and leave this bit cleared to '0'. Reset type: SYSRSn
19	DRXE	R/W	0h	Message Stored to Dedicated Rx Buffer Enable Reset type: SYSRSn
18	TOOE	R/W	0h	Timeout Occurred Enable Reset type: SYSRSn
17	MRAFE	R/W	0h	Message RAM Access Failure Enable Reset type: SYSRSn

Table 45-46. MCAN_IE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	TSWE	R/W	0h	Timestamp Wraparound Enable Reset type: SYSRSn
15	TEFLE	R/W	0h	Tx Event FIFO Element Lost Enable Reset type: SYSRSn
14	TEFFE	R/W	0h	Tx Event FIFO Full Enable Reset type: SYSRSn
13	TEFWE	R/W	0h	Tx Event FIFO Watermark Reached Enable Reset type: SYSRSn
12	TEFNE	R/W	0h	Tx Event FIFO New Entry Enable Reset type: SYSRSn
11	TFEE	R/W	0h	Tx FIFO Empty Enable Reset type: SYSRSn
10	TCFE	R/W	0h	Transmission Cancellation Finished Enable Reset type: SYSRSn
9	TCE	R/W	0h	Transmission Completed Enable Reset type: SYSRSn
8	HPME	R/W	0h	High Priority Message Enable Reset type: SYSRSn
7	RF1LE	R/W	0h	Rx FIFO 1 Message Lost Enable Reset type: SYSRSn
6	RF1FE	R/W	0h	Rx FIFO 1 Full Enable Reset type: SYSRSn
5	RF1WE	R/W	0h	Rx FIFO 1 Watermark Reached Enable Reset type: SYSRSn
4	RF1NE	R/W	0h	Rx FIFO 1 New Message Enable Reset type: SYSRSn
3	RF0LE	R/W	0h	Rx FIFO 0 Message Lost Enable Reset type: SYSRSn
2	RF0FE	R/W	0h	Rx FIFO 0 Full Enable Reset type: SYSRSn
1	RF0WE	R/W	0h	Rx FIFO 0 Watermark Reached Enable Reset type: SYSRSn
0	RF0NE	R/W	0h	Rx FIFO 0 New Message Enable Reset type: SYSRSn

45.7.4.17 MCAN_ILS Register (Offset (x8) = 58h, Offset (x16) = 2Ch) [Reset = 0h]

MCAN_ILS is shown in Figure 45-52 and described in Table 45-47.

Return to the Summary Table.

The Interrupt Line Select register assigns an interrupt generated by a specific interrupt flag from the Interrupt Register to one of the two module interrupt lines. For interrupt generation the respective interrupt line has to be enabled via ILE.EINT0 and ILE.EINT1.

Figure 45-52. MCAN_ILS Register

31	30	29	28	27	26	25	24
RESERVED		ARAL	PEDL	PEAL	WDIL	BOL	EWL
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
EPL	ELOL	BEUL	BECL	DRXL	TOOL	MRAFL	TSWL
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
TEFLL	TEFFL	TEFWL	TEFNL	TFEL	TCFL	TCL	HPML
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RF1LL	RF1FL	RF1WL	RF1NL	RF0LL	RF0FL	RF0WL	RF0NL
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 45-47. MCAN_ILS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29	ARAL	R/W	0h	Access to Reserved Address Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
28	PEDL	R/W	0h	Protocol Error in Data Phase Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
27	PEAL	R/W	0h	Protocol Error in Arbitration Phase Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
26	WDIL	R/W	0h	Watchdog Interrupt Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
25	BOL	R/W	0h	Bus_Off Status Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
24	EWL	R/W	0h	Warning Status Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
23	EPL	R/W	0h	Error Passive Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn

Table 45-47. MCAN_ILS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22	ELOL	R/W	0h	Error Logging Overflow Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
21	BEUL	R/W	0h	Bit Error Uncorrected Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
20	BECL	R/W	0h	Bit Error Corrected Line A separate interrupt line reserved for corrected bit errors is provided via the MCAN_ERROR_REGS. It advised for the user to use these registers and leave the MCAN_IE.BECE bit cleared to '0' (disabled), thereby relegating this bit to not applicable. Reset type: SYSRSn
19	DRXL	R/W	0h	Message Stored to Dedicated Rx Buffer Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
18	TOOL	R/W	0h	Timeout Occurred Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
17	MRAFL	R/W	0h	Message RAM Access Failure Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
16	TSWL	R/W	0h	Timestamp Wraparound Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
15	TEFLL	R/W	0h	Tx Event FIFO Element Lost Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
14	TEFFL	R/W	0h	Tx Event FIFO Full Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
13	TEFWL	R/W	0h	Tx Event FIFO Watermark Reached Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
12	TEFNL	R/W	0h	Tx Event FIFO New Entry Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
11	TFEL	R/W	0h	Tx FIFO Empty Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
10	TCFL	R/W	0h	Transmission Cancellation Finished Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
9	TCL	R/W	0h	Transmission Completed Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn

Table 45-47. MCAN_ILS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	HPML	R/W	0h	High Priority Message Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
7	RF1LL	R/W	0h	Rx FIFO 1 Message Lost Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
6	RF1FL	R/W	0h	Rx FIFO 1 Full Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
5	RF1WL	R/W	0h	Rx FIFO 1 Watermark Reached Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
4	RF1NL	R/W	0h	Rx FIFO 1 New Message Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
3	RF0LL	R/W	0h	Rx FIFO 0 Message Lost Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
2	RF0FL	R/W	0h	Rx FIFO 0 Full Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
1	RF0WL	R/W	0h	Rx FIFO 0 Watermark Reached Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
0	RF0NL	R/W	0h	Rx FIFO 0 New Message Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn

45.7.4.18 MCAN_ILE Register (Offset (x8) = 5Ch, Offset (x16) = 2Eh) [Reset = 0h]

MCAN_ILE is shown in Figure 45-53 and described in Table 45-48.

Return to the Summary Table.

MCAN Interrupt Line Enable

Figure 45-53. MCAN_ILE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						EINT1	EINT0
R-0h						R/W-0h	R/W-0h

Table 45-48. MCAN_ILE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	EINT1	R/W	0h	Enable Interrupt Line 1 0 Interrupt Line 1 is disabled 1 Interrupt Line 1 is enabled Reset type: SYSRSn
0	EINT0	R/W	0h	Enable Interrupt Line 0 0 Interrupt Line 0 is disabled 1 Interrupt Line 0 is enabled Reset type: SYSRSn

45.7.4.19 MCAN_GFC Register (Offset (x8) = 80h, Offset (x16) = 40h) [Reset = 0h]

MCAN_GFC is shown in Figure 45-54 and described in Table 45-49.

Return to the Summary Table.

MCAN Global Filter Configuration

Figure 45-54. MCAN_GFC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		ANFS		ANFE		RRFS	RRFE
R-0h		R/WQ-0h		R/WQ-0h		R/WQ-0h	R/WQ-0h

Table 45-49. MCAN_GFC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5-4	ANFS	R/WQ	0h	Accept Non-matching Frames Standard. Defines how received messages with 11-bit IDs that do not match any element of the filter list are treated. 00 Accept in Rx FIFO 0 01 Accept in Rx FIFO 1 10 Reject 11 Reject Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
3-2	ANFE	R/WQ	0h	Accept Non-matching Frames Extended. Defines how received messages with 29-bit IDs that do not match any element of the filter list are treated. 00 Accept in Rx FIFO 0 01 Accept in Rx FIFO 1 10 Reject 11 Reject Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
1	RRFS	R/WQ	0h	Reject Remote Frames Standard 0 Filter remote frames with 11-bit standard IDs 1 Reject all remote frames with 11-bit standard IDs Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
0	RRFE	R/WQ	0h	Reject Remote Frames Extended 0 Filter remote frames with 29-bit extended IDs 1 Reject all remote frames with 29-bit extended IDs Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

45.7.4.20 MCAN_SIDFC Register (Offset (x8) = 84h, Offset (x16) = 42h) [Reset = 0h]

MCAN_SIDFC is shown in Figure 45-55 and described in Table 45-50.

Return to the Summary Table.

MCAN Standard ID Filter Configuration

Figure 45-55. MCAN_SIDFC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
LSS							
R/WQ-0h							
15	14	13	12	11	10	9	8
FLSSA							
R/WQ-0h							
7	6	5	4	3	2	1	0
FLSSA						RESERVED	
R/WQ-0h						R-0h	

Table 45-50. MCAN_SIDFC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	LSS	R/WQ	0h	List Size Standard 0 No standard Message ID filter 1-128 Number of standard Message ID filter elements >128 Values greater than 128 are interpreted as 128 Reset type: SYSRSn
15-2	FLSSA	R/WQ	0h	Filter List Standard Start Address. Start address of standard Message ID filter list (32-bit word address). Reset type: SYSRSn
1-0	RESERVED	R	0h	Reserved

45.7.4.21 MCAN_XIDFC Register (Offset (x8) = 88h, Offset (x16) = 44h) [Reset = 0h]

MCAN_XIDFC is shown in Figure 45-56 and described in Table 45-51.

Return to the Summary Table.

MCAN Extended ID Filter Configuration

Figure 45-56. MCAN_XIDFC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED	LSE						
R-0h	R/WQ-0h						
15	14	13	12	11	10	9	8
FLESA							
R/WQ-0h							
7	6	5	4	3	2	1	0
FLESA						RESERVED	
R/WQ-0h						R-0h	

Table 45-51. MCAN_XIDFC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22-16	LSE	R/WQ	0h	List Size Extended 0 No extended Message ID filter 1-64 Number of extended Message ID filter elements >64 Values greater than 64 are interpreted as 64 Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
15-2	FLESA	R/WQ	0h	Filter List Extended Start Address. Start address of extended Message ID filter list (32-bit word address). Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
1-0	RESERVED	R	0h	Reserved

45.7.4.22 MCAN_XIDAM Register (Offset (x8) = 90h, Offset (x16) = 48h) [Reset = 1FFFFFFFh]

MCAN_XIDAM is shown in Figure 45-57 and described in Table 45-52.

Return to the Summary Table.

MCAN Extended ID and Mask

Figure 45-57. MCAN_XIDAM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED			EIDM												
R-0h			R/WQ-1FFFFFFFh												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EIDM															
R/WQ-1FFFFFFFh															

Table 45-52. MCAN_XIDAM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved
28-0	EIDM	R/WQ	1FFFFFFFh	Extended ID Mask. For acceptance filtering of extended frames the Extended ID AND Mask is ANDed with the Message ID of a received frame. Intended for masking of 29-bit IDs in SAE J1939. With the reset value of all bits set to one the mask is not active. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

45.7.4.23 MCAN_HPMS Register (Offset (x8) = 94h, Offset (x16) = 4Ah) [Reset = 0h]

MCAN_HPMS is shown in Figure 45-58 and described in Table 45-53.

Return to the Summary Table.

This register is updated every time a Message ID filter element configured to generate a priority event matches. This can be used to monitor the status of incoming high priority messages and to enable fast access to these messages.

Figure 45-58. MCAN_HPMS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
FLST	FIDX						
R-0h	R-0h						
7	6	5	4	3	2	1	0
MSI		BIDX					
R-0h		R-0h					

Table 45-53. MCAN_HPMS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	FLST	R	0h	Filter List. Indicates the filter list of the matching filter element. 0 Standard Filter List 1 Extended Filter List Reset type: SYSRSn
14-8	FIDX	R	0h	Filter Index. Index of matching filter element. Range is 0 to SIDFC.LSS - 1 resp. XIDFC.LSE - 1. Reset type: SYSRSn
7-6	MSI	R	0h	Message Storage Indicator 00 No FIFO selected 01 FIFO message lost 10 Message stored in FIFO 0 11 Message stored in FIFO 1 Reset type: SYSRSn
5-0	BIDX	R	0h	Buffer Index. Index of Rx FIFO element to which the message was stored. Only valid when MSI[1] = '1'. Reset type: SYSRSn

45.7.4.24 MCAN_NDAT1 Register (Offset (x8) = 98h, Offset (x16) = 4Ch) [Reset = 0h]

MCAN_NDAT1 is shown in Figure 45-59 and described in Table 45-54.

Return to the Summary Table.

MCAN New Data 1

Figure 45-59. MCAN_NDAT1 Register

31	30	29	28	27	26	25	24
ND31	ND30	ND29	ND28	ND27	ND26	ND25	ND24
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
23	22	21	20	19	18	17	16
ND23	ND22	ND21	ND20	ND19	ND18	ND17	ND16
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
15	14	13	12	11	10	9	8
ND15	ND14	ND13	ND12	ND11	ND10	ND9	ND8
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
7	6	5	4	3	2	1	0
ND7	ND6	ND5	ND4	ND3	ND2	ND1	ND0
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

Table 45-54. MCAN_NDAT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ND31	R/W1C	0h	New Data RX Buffer 31 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
30	ND30	R/W1C	0h	New Data RX Buffer 30 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
29	ND29	R/W1C	0h	New Data RX Buffer 29 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
28	ND28	R/W1C	0h	New Data RX Buffer 28 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
27	ND27	R/W1C	0h	New Data RX Buffer 27 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
26	ND26	R/W1C	0h	New Data RX Buffer 26 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
25	ND25	R/W1C	0h	New Data RX Buffer 25 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
24	ND24	R/W1C	0h	New Data RX Buffer 24 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn

Table 45-54. MCAN_NDAT1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	ND23	R/W1C	0h	New Data RX Buffer 23 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
22	ND22	R/W1C	0h	New Data RX Buffer 22 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
21	ND21	R/W1C	0h	New Data RX Buffer 21 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
20	ND20	R/W1C	0h	New Data RX Buffer 20 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
19	ND19	R/W1C	0h	New Data RX Buffer 19 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
18	ND18	R/W1C	0h	New Data RX Buffer 18 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
17	ND17	R/W1C	0h	New Data RX Buffer 17 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
16	ND16	R/W1C	0h	New Data RX Buffer 16 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
15	ND15	R/W1C	0h	New Data RX Buffer 15 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
14	ND14	R/W1C	0h	New Data RX Buffer 14 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
13	ND13	R/W1C	0h	New Data RX Buffer 13 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
12	ND12	R/W1C	0h	New Data RX Buffer 12 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
11	ND11	R/W1C	0h	New Data RX Buffer 11 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
10	ND10	R/W1C	0h	New Data RX Buffer 10 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn

Table 45-54. MCAN_NDAT1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	ND9	R/W1C	0h	New Data RX Buffer 9 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
8	ND8	R/W1C	0h	New Data RX Buffer 8 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
7	ND7	R/W1C	0h	New Data RX Buffer 7 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
6	ND6	R/W1C	0h	New Data RX Buffer 6 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
5	ND5	R/W1C	0h	New Data RX Buffer 5 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
4	ND4	R/W1C	0h	New Data RX Buffer 4 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
3	ND3	R/W1C	0h	New Data RX Buffer 3 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
2	ND2	R/W1C	0h	New Data RX Buffer 2 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
1	ND1	R/W1C	0h	New Data RX Buffer 1 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
0	ND0	R/W1C	0h	New Data RX Buffer 0 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn

45.7.4.25 MCAN_NDAT2 Register (Offset (x8) = 9Ch, Offset (x16) = 4Eh) [Reset = 0h]

MCAN_NDAT2 is shown in Figure 45-60 and described in Table 45-55.

Return to the Summary Table.

MCAN New Data 2

Figure 45-60. MCAN_NDAT2 Register

31	30	29	28	27	26	25	24
ND63	ND62	ND61	ND60	ND59	ND58	ND57	ND56
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
23	22	21	20	19	18	17	16
ND55	ND54	ND53	ND52	ND51	ND50	ND49	ND48
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
15	14	13	12	11	10	9	8
ND47	ND46	ND45	ND44	ND43	ND42	ND41	ND40
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
7	6	5	4	3	2	1	0
ND39	ND38	ND37	ND36	ND35	ND34	ND33	ND32
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

Table 45-55. MCAN_NDAT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ND63	R/W1C	0h	New Data RX Buffer 63 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
30	ND62	R/W1C	0h	New Data RX Buffer 62 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
29	ND61	R/W1C	0h	New Data RX Buffer 61 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
28	ND60	R/W1C	0h	New Data RX Buffer 60 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
27	ND59	R/W1C	0h	New Data RX Buffer 59 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
26	ND58	R/W1C	0h	New Data RX Buffer 58 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
25	ND57	R/W1C	0h	New Data RX Buffer 57 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
24	ND56	R/W1C	0h	New Data RX Buffer 56 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn

Table 45-55. MCAN_NDAT2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	ND55	R/W1C	0h	New Data RX Buffer 55 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
22	ND54	R/W1C	0h	New Data RX Buffer 54 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
21	ND53	R/W1C	0h	New Data RX Buffer 53 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
20	ND52	R/W1C	0h	New Data RX Buffer 52 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
19	ND51	R/W1C	0h	New Data RX Buffer 51 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
18	ND50	R/W1C	0h	New Data RX Buffer 50 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
17	ND49	R/W1C	0h	New Data RX Buffer 49 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
16	ND48	R/W1C	0h	New Data RX Buffer 48 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
15	ND47	R/W1C	0h	New Data RX Buffer 47 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
14	ND46	R/W1C	0h	New Data RX Buffer 46 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
13	ND45	R/W1C	0h	New Data RX Buffer 45 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
12	ND44	R/W1C	0h	New Data RX Buffer 44 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
11	ND43	R/W1C	0h	New Data RX Buffer 43 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
10	ND42	R/W1C	0h	New Data RX Buffer 42 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn

Table 45-55. MCAN_NDAT2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	ND41	R/W1C	0h	New Data RX Buffer 41 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
8	ND40	R/W1C	0h	New Data RX Buffer 40 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
7	ND39	R/W1C	0h	New Data RX Buffer 39 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
6	ND38	R/W1C	0h	New Data RX Buffer 38 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
5	ND37	R/W1C	0h	New Data RX Buffer 37 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
4	ND36	R/W1C	0h	New Data RX Buffer 36 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
3	ND35	R/W1C	0h	New Data RX Buffer 35 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
2	ND34	R/W1C	0h	New Data RX Buffer 34 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
1	ND33	R/W1C	0h	New Data RX Buffer 33 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
0	ND32	R/W1C	0h	New Data RX Buffer 32 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn

45.7.4.26 MCAN_RXF0C Register (Offset (x8) = A0h, Offset (x16) = 50h) [Reset = 0h]

MCAN_RXF0C is shown in Figure 45-61 and described in Table 45-56.

Return to the Summary Table.

MCAN Rx FIFO 0 Configuration

Figure 45-61. MCAN_RXF0C Register

31	30	29	28	27	26	25	24
F0OM	F0WM						
R/WQ-0h	R/WQ-0h						
23	22	21	20	19	18	17	16
RESERVED	F0S						
R-0h	R/WQ-0h						
15	14	13	12	11	10	9	8
F0SA							
R/WQ-0h							
7	6	5	4	3	2	1	0
F0SA						RESERVED	
R/WQ-0h						R-0h	

Table 45-56. MCAN_RXF0C Register Field Descriptions

Bit	Field	Type	Reset	Description
31	F0OM	R/WQ	0h	FIFO 0 Operation Mode. FIFO 0 can be operated in blocking or in overwrite mode. 0 FIFO 0 blocking mode 1 FIFO 0 overwrite mode Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
30-24	F0WM	R/WQ	0h	Rx FIFO 0 Watermark 0 Watermark interrupt disabled 1-64 Level for Rx FIFO 0 watermark interrupt (IR.RF0W) >64 Watermark interrupt disabled Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
23	RESERVED	R	0h	Reserved
22-16	F0S	R/WQ	0h	Rx FIFO 0 Size. The Rx FIFO 0 elements are indexed from 0 to F0S-1. 0 No Rx FIFO 0 1-64 Number of Rx FIFO 0 elements >64 Values greater than 64 are interpreted as 64 Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
15-2	F0SA	R/WQ	0h	Rx FIFO 0 Start Address. Start address of Rx FIFO 0 in Message RAM (32-bit word address). Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
1-0	RESERVED	R	0h	Reserved

45.7.4.27 MCAN_RXF0S Register (Offset (x8) = A4h, Offset (x16) = 52h) [Reset = 0h]

MCAN_RXF0S is shown in Figure 45-62 and described in Table 45-57.

Return to the Summary Table.

MCAN Rx FIFO 0 Status

Figure 45-62. MCAN_RXF0S Register

31	30	29	28	27	26	25	24
RESERVED						RF0L	F0F
R-0h						R-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED		F0PI					
R-0h		R-0h					
15	14	13	12	11	10	9	8
RESERVED		F0GI					
R-0h		R-0h					
7	6	5	4	3	2	1	0
RESERVED	F0FL						
R-0h	R-0h						

Table 45-57. MCAN_RXF0S Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved
25	RF0L	R	0h	Rx FIFO 0 Message Lost. This bit is a copy of interrupt flag IR.RF0L. When IR.RF0L is reset, this bit is also reset. 0 No Rx FIFO 0 message lost 1 Rx FIFO 0 message lost, also set after write attempt to Rx FIFO 0 of size zero Note: Overwriting the oldest message when RXF0C.F0OM = '1' will not set this flag. Reset type: SYSRSn
24	F0F	R	0h	Rx FIFO 0 Full 0 Rx FIFO 0 not full 1 Rx FIFO 0 full Reset type: SYSRSn
23-22	RESERVED	R	0h	Reserved
21-16	F0PI	R	0h	Rx FIFO 0 Put Index. Rx FIFO 0 write index pointer, range 0 to 63. Reset type: SYSRSn
15-14	RESERVED	R	0h	Reserved
13-8	F0GI	R	0h	Rx FIFO 0 Get Index. Rx FIFO 0 read index pointer, range 0 to 63. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6-0	F0FL	R	0h	Rx FIFO 0 Fill Level. Number of elements stored in Rx FIFO 0, range 0 to 64. Reset type: SYSRSn

45.7.4.28 MCAN_RXF0A Register (Offset (x8) = A8h, Offset (x16) = 54h) [Reset = 0h]

MCAN_RXF0A is shown in Figure 45-63 and described in Table 45-58.

Return to the Summary Table.

MCAN Rx FIFO 0 Acknowledge

Figure 45-63. MCAN_RXF0A Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										F0AI					
R-0h																										R/W-0h					

Table 45-58. MCAN_RXF0A Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5-0	F0AI	R/W	0h	Rx FIFO 0 Acknowledge Index. After the Host has read a message or a sequence of messages from Rx FIFO 0 it has to write the buffer index of the last element read from Rx FIFO 0 to F0AI. This will set the Rx FIFO 0 Get Index RXF0S.F0GI to F0AI + 1 and update the FIFO 0 Fill Level RXF0S.F0FL. Reset type: SYSRSn

45.7.4.29 MCAN_RXBC Register (Offset (x8) = ACh, Offset (x16) = 56h) [Reset = 0h]

MCAN_RXBC is shown in Figure 45-64 and described in Table 45-59.

Return to the Summary Table.

MCAN Rx Buffer Configuration

Figure 45-64. MCAN_RXBC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RBSA							
R/WQ-0h							
7	6	5	4	3	2	1	0
RBSA						RESERVED	
R/WQ-0h						R-0h	

Table 45-59. MCAN_RXBC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-2	RBSA	R/WQ	0h	Rx Buffer Start Address. Configures the start address of the Rx Buffers section in the Message RAM (32-bit word address). +I466 Reset type: SYSRSn
1-0	RESERVED	R	0h	Reserved

45.7.4.30 MCAN_RXF1C Register (Offset (x8) = B0h, Offset (x16) = 58h) [Reset = 0h]

MCAN_RXF1C is shown in Figure 45-65 and described in Table 45-60.

Return to the Summary Table.

MCAN Rx FIFO 1 Configuration

Figure 45-65. MCAN_RXF1C Register

31	30	29	28	27	26	25	24
F1OM	F1WM						
R/WQ-0h	R/WQ-0h						
23	22	21	20	19	18	17	16
RESERVED	F1S						
R-0h	R/WQ-0h						
15	14	13	12	11	10	9	8
F1SA							
R/WQ-0h							
7	6	5	4	3	2	1	0
F1SA						RESERVED	
R/WQ-0h						R-0h	

Table 45-60. MCAN_RXF1C Register Field Descriptions

Bit	Field	Type	Reset	Description
31	F1OM	R/WQ	0h	FIFO 1 Operation Mode. FIFO 1 can be operated in blocking or in overwrite mode. 0 FIFO 1 blocking mode 1 FIFO 1 overwrite mode Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
30-24	F1WM	R/WQ	0h	Rx FIFO 1 Watermark 0 Watermark interrupt disabled 1-64 Level for Rx FIFO 1 watermark interrupt (IR.RF1W) >64 Watermark interrupt disabled Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
23	RESERVED	R	0h	Reserved
22-16	F1S	R/WQ	0h	Rx FIFO 1 Size. The Rx FIFO 1 elements are indexed from 0 to F1S - 1. 0 No Rx FIFO 1 1-64 Number of Rx FIFO 1 elements >64 Values greater than 64 are interpreted as 64 Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
15-2	F1SA	R/WQ	0h	Rx FIFO 1 Start Address Start address of Rx FIFO 1 in Message RAM (32-bit word address). Reset type: SYSRSn
1-0	RESERVED	R	0h	Reserved

45.7.4.31 MCAN_RXF1S Register (Offset (x8) = B4h, Offset (x16) = 5Ah) [Reset = 0h]

MCAN_RXF1S is shown in Figure 45-66 and described in Table 45-61.

Return to the Summary Table.

MCAN Rx FIFO 1 Status

Figure 45-66. MCAN_RXF1S Register

31	30	29	28	27	26	25	24
DMS		RESERVED				RF1L	F1F
R-0h		R-0h				R-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED		F1PI					
R-0h		R-0h					
15	14	13	12	11	10	9	8
RESERVED		F1GI					
R-0h		R-0h					
7	6	5	4	3	2	1	0
RESERVED	F1FL						
R-0h	R-0h						

Table 45-61. MCAN_RXF1S Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	DMS	R	0h	Debug Message Status 00 Idle state, wait for reception of debug messages, DMA request is cleared 01 Debug message A received 10 Debug messages A, B received 11 Debug messages A, B, C received, DMA request is set Reset type: SYSRSn
29-26	RESERVED	R	0h	Reserved
25	RF1L	R	0h	Rx FIFO 1 Message Lost. This bit is a copy of interrupt flag IR.RF1L. When IR.RF1L is reset, this bit is also reset. 0 No Rx FIFO 1 message lost 1 Rx FIFO 1 message lost, also set after write attempt to Rx FIFO 1 of size zero Note: Overwriting the oldest message when RXF1C.F1OM = '1' will not set this flag. Reset type: SYSRSn
24	F1F	R	0h	Rx FIFO 1 Full 0 Rx FIFO 1 not full 1 Rx FIFO 1 full Reset type: SYSRSn
23-22	RESERVED	R	0h	Reserved
21-16	F1PI	R	0h	Rx FIFO 1 Put Index. Rx FIFO 1 write index pointer, range 0 to 63. Reset type: SYSRSn
15-14	RESERVED	R	0h	Reserved
13-8	F1GI	R	0h	Rx FIFO 1 Get Index. Rx FIFO 1 read index pointer, range 0 to 63. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6-0	F1FL	R	0h	Rx FIFO 1 Fill Level. Number of elements stored in Rx FIFO 1, range 0 to 64. Reset type: SYSRSn

45.7.4.32 MCAN_RXF1A Register (Offset (x8) = B8h, Offset (x16) = 5Ch) [Reset = 0h]

MCAN_RXF1A is shown in Figure 45-67 and described in Table 45-62.

Return to the Summary Table.

MCAN Rx FIFO 1 Acknowledge

Figure 45-67. MCAN_RXF1A Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										F1AI					
R-0h																										R/W-0h					

Table 45-62. MCAN_RXF1A Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5-0	F1AI	R/W	0h	Rx FIFO 1 Acknowledge Index. After the Host has read a message or a sequence of messages from Rx FIFO 1 it has to write the buffer index of the last element read from Rx FIFO 1 to F1AI. This will set the Rx FIFO 1 Get Index RXF1S.F1GI to F1AI + 1 and update the FIFO 1 Fill Level RXF1S.F1FL. Reset type: SYSRSn

45.7.4.33 MCAN_RXESC Register (Offset (x8) = BCh, Offset (x16) = 5Eh) [Reset = 0h]

MCAN_RXESC is shown in Figure 45-68 and described in Table 45-63.

Return to the Summary Table.

Configures the number of data bytes belonging to an Rx Buffer / Rx FIFO element. Data field sizes >8 bytes are intended for CAN FD operation only.

Figure 45-68. MCAN_RXESC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					RBDS		
R-0h					R/WQ-0h		
7	6	5	4	3	2	1	0
RESERVED	F1DS			RESERVED	F0DS		
R-0h	R/WQ-0h			R-0h	R/WQ-0h		

Table 45-63. MCAN_RXESC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-11	RESERVED	R	0h	Reserved
10-8	RBDS	R/WQ	0h	Rx Buffer Data Field Size 000 8 byte data field 001 12 byte data field 010 16 byte data field 011 20 byte data field 100 24 byte data field 101 32 byte data field 110 48 byte data field 111 64 byte data field Note: In case the data field size of an accepted CAN frame exceeds the data field size configured for the matching Rx Buffer or Rx FIFO, only the number of bytes as configured by RXESC are stored to the Rx Buffer resp. Rx FIFO element. The rest of the frame's data field is ignored. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved

Table 45-63. MCAN_RXESC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-4	F1DS	R/WQ	0h	Rx FIFO 1 Data Field Size 000 8 byte data field 001 12 byte data field 010 16 byte data field 011 20 byte data field 100 24 byte data field 101 32 byte data field 110 48 byte data field 111 64 byte data field Note: In case the data field size of an accepted CAN frame exceeds the data field size configured for the matching Rx Buffer or Rx FIFO, only the number of bytes as configured by RXESC are stored to the Rx Buffer resp. Rx FIFO element. The rest of the frame's data field is ignored. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
3	RESERVED	R	0h	Reserved
2-0	F0DS	R/WQ	0h	Rx FIFO 0 Data Field Size 000 8 byte data field 001 12 byte data field 010 16 byte data field 011 20 byte data field 100 24 byte data field 101 32 byte data field 110 48 byte data field 111 64 byte data field Note: In case the data field size of an accepted CAN frame exceeds the data field size configured for the matching Rx Buffer or Rx FIFO, only the number of bytes as configured by RXESC are stored to the Rx Buffer resp. Rx FIFO element. The rest of the frame's data field is ignored. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

45.7.4.34 MCAN_TXBC Register (Offset (x8) = C0h, Offset (x16) = 60h) [Reset = 0h]

MCAN_TXBC is shown in Figure 45-69 and described in Table 45-64.

Return to the Summary Table.

MCAN Tx Buffer Configuration

Figure 45-69. MCAN_TXBC Register

31	30	29	28	27	26	25	24
RESERVED	TFQM	TFQS					
R-0h	R/WQ-0h	R/WQ-0h					
23	22	21	20	19	18	17	16
RESERVED		NDTB					
R-0h		R/WQ-0h					
15	14	13	12	11	10	9	8
TBSA							
R/WQ-0h							
7	6	5	4	3	2	1	0
TBSA						RESERVED	
R/WQ-0h						R-0h	

Table 45-64. MCAN_TXBC Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30	TFQM	R/WQ	0h	Tx FIFO/Queue Mode 0 Tx FIFO operation 1 Tx Queue operation Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
29-24	TFQS	R/WQ	0h	Transmit FIFO/Queue Size 0 No Tx FIFO/Queue 1-32 Number of Tx Buffers used for Tx FIFO/Queue >32 Values greater than 32 are interpreted as 32 Note: Be aware that the sum of TFQS and NDTB may be not greater than 32. There is no check for erroneous configurations. The Tx Buffers section in the Message RAM starts with the dedicated Tx Buffers. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
23-22	RESERVED	R	0h	Reserved
21-16	NDTB	R/WQ	0h	Number of Dedicated Transmit Buffers 0 No Dedicated Tx Buffers 1-32 Number of Dedicated Tx Buffers >32 Values greater than 32 are interpreted as 32 Note: Be aware that the sum of TFQS and NDTB may be not greater than 32. There is no check for erroneous configurations. The Tx Buffers section in the Message RAM starts with the dedicated Tx Buffers. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

Table 45-64. MCAN_TXBC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-2	TBSA	R/WQ	0h	Tx Buffers Start Address. Start address of Tx Buffers section in Message RAM (32-bit word address). Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
1-0	RESERVED	R	0h	Reserved

45.7.4.35 MCAN_TXFQS Register (Offset (x8) = C4h, Offset (x16) = 62h) [Reset = 0h]

MCAN_TXFQS is shown in Figure 45-70 and described in Table 45-65.

Return to the Summary Table.

The Tx FIFO/Queue status is related to the pending Tx requests listed in register TXBRP. Therefore the effect of Add/Cancellation requests may be delayed due to a running Tx scan (TXBRP not yet updated).

Figure 45-70. MCAN_TXFQS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED		TFQF	TFQP				
R-0h		R-0h	R-0h				
15	14	13	12	11	10	9	8
RESERVED			TFGI				
R-0h			R-0h				
7	6	5	4	3	2	1	0
RESERVED		TFFL					
R-0h		R-0h					

Table 45-65. MCAN_TXFQS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reserved
21	TFQF	R	0h	Tx FIFO/Queue Full 0 Tx FIFO/Queue not full 1 Tx FIFO/Queue full Reset type: SYSRSn
20-16	TFQP	R	0h	Tx FIFO/Queue Put Index. Tx FIFO/Queue write index pointer, range 0 to 31. Note: In case of mixed configurations where dedicated Tx Buffers are combined with a Tx FIFO or a Tx Queue, the Put and Get Indices indicate the number of the Tx Buffer starting with the first dedicated Tx Buffers. Example: For a configuration of 12 dedicated Tx Buffers and a Tx FIFO of 20 Buffers a Put Index of 15 points to the fourth buffer of the Tx FIFO. Reset type: SYSRSn
15-13	RESERVED	R	0h	Reserved
12-8	TFGI	R	0h	Tx FIFO Get Index. Tx FIFO read index pointer, range 0 to 31. Read as zero when Tx Queue operation is configured (TXBC.TFQM = '1'). Note: In case of mixed configurations where dedicated Tx Buffers are combined with a Tx FIFO or a Tx Queue, the Put and Get Indices indicate the number of the Tx Buffer starting with the first dedicated Tx Buffers. Example: For a configuration of 12 dedicated Tx Buffers and a Tx FIFO of 20 Buffers a Put Index of 15 points to the fourth buffer of the Tx FIFO. Reset type: SYSRSn
7-6	RESERVED	R	0h	Reserved
5-0	TFFL	R	0h	Tx FIFO Free Level. Number of consecutive free Tx FIFO elements starting from TFGI, range 0 to 32. Read as zero when Tx Queue operation is configured (TXBC.TFQM = '1'). Reset type: SYSRSn

45.7.4.36 MCAN_TXESC Register (Offset (x8) = C8h, Offset (x16) = 64h) [Reset = 0h]

MCAN_TXESC is shown in Figure 45-71 and described in Table 45-66.

Return to the Summary Table.

Configures the number of data bytes belonging to a Tx Buffer element. Data field sizes > 8 bytes are intended for CAN FD operation only.

Figure 45-71. MCAN_TXESC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													TBDS		
R-0h													R/WQ-0h		

Table 45-66. MCAN_TXESC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2-0	TBDS	R/WQ	0h	Tx Buffer Data Field Size 000 8 byte data field 001 12 byte data field 010 16 byte data field 011 20 byte data field 100 24 byte data field 101 32 byte data field 110 48 byte data field 111 64 byte data field Note: In case the data length code DLC of a Tx Buffer element is configured to a value higher than the Tx Buffer data field size TXESC.TBDS, the bytes not defined by the Tx Buffer are transmitted as "0xCC" (padding bytes). Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

45.7.4.37 MCAN_TXBRP Register (Offset (x8) = CCh, Offset (x16) = 66h) [Reset = 0h]

MCAN_TXBRP is shown in Figure 45-72 and described in Table 45-67.

Return to the Summary Table.

MCAN Tx Buffer Request Pending

Figure 45-72. MCAN_TXBRP Register

31	30	29	28	27	26	25	24
TRP31	TRP30	TRP29	TRP28	TRP27	TRP26	TRP25	TRP24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
TRP23	TRP22	TRP21	TRP20	TRP19	TRP18	TRP17	TRP16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
TRP15	TRP14	TRP13	TRP12	TRP11	TRP10	TRP9	TRP8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
TRP7	TRP6	TRP5	TRP4	TRP3	TRP2	TRP1	TRP0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 45-67. MCAN_TXBRP Register Field Descriptions

Bit	Field	Type	Reset	Description
31	TRP31	R	0h	Transmission Request Pending 31. See description for bit 0. Reset type: SYSRSn
30	TRP30	R	0h	Transmission Request Pending 30. See description for bit 0. Reset type: SYSRSn
29	TRP29	R	0h	Transmission Request Pending 29. See description for bit 0. Reset type: SYSRSn
28	TRP28	R	0h	Transmission Request Pending 28. See description for bit 0. Reset type: SYSRSn
27	TRP27	R	0h	Transmission Request Pending 27. See description for bit 0. Reset type: SYSRSn
26	TRP26	R	0h	Transmission Request Pending 26. See description for bit 0. Reset type: SYSRSn
25	TRP25	R	0h	Transmission Request Pending 25. See description for bit 0. Reset type: SYSRSn
24	TRP24	R	0h	Transmission Request Pending 24. See description for bit 0. Reset type: SYSRSn
23	TRP23	R	0h	Transmission Request Pending 23. See description for bit 0. Reset type: SYSRSn
22	TRP22	R	0h	Transmission Request Pending 22. See description for bit 0. Reset type: SYSRSn
21	TRP21	R	0h	Transmission Request Pending 21. See description for bit 0. Reset type: SYSRSn
20	TRP20	R	0h	Transmission Request Pending 20. See description for bit 0. Reset type: SYSRSn
19	TRP19	R	0h	Transmission Request Pending 19. See description for bit 0. Reset type: SYSRSn
18	TRP18	R	0h	Transmission Request Pending 18. See description for bit 0. Reset type: SYSRSn
17	TRP17	R	0h	Transmission Request Pending 17. See description for bit 0. Reset type: SYSRSn

Table 45-67. MCAN_TXBRP Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	TRP16	R	0h	Transmission Request Pending 16. See description for bit 0. Reset type: SYSRSn
15	TRP15	R	0h	Transmission Request Pending 15. See description for bit 0. Reset type: SYSRSn
14	TRP14	R	0h	Transmission Request Pending 14. See description for bit 0. Reset type: SYSRSn
13	TRP13	R	0h	Transmission Request Pending 13. See description for bit 0. Reset type: SYSRSn
12	TRP12	R	0h	Transmission Request Pending 12. See description for bit 0. Reset type: SYSRSn
11	TRP11	R	0h	Transmission Request Pending 11. See description for bit 0. Reset type: SYSRSn
10	TRP10	R	0h	Transmission Request Pending 10. See description for bit 0. Reset type: SYSRSn
9	TRP9	R	0h	Transmission Request Pending 9. See description for bit 0. Reset type: SYSRSn
8	TRP8	R	0h	Transmission Request Pending 8. See description for bit 0. Reset type: SYSRSn
7	TRP7	R	0h	Transmission Request Pending 7. See description for bit 0. Reset type: SYSRSn
6	TRP6	R	0h	Transmission Request Pending 6. See description for bit 0. Reset type: SYSRSn
5	TRP5	R	0h	Transmission Request Pending 5. See description for bit 0. Reset type: SYSRSn
4	TRP4	R	0h	Transmission Request Pending 4. See description for bit 0. Reset type: SYSRSn
3	TRP3	R	0h	Transmission Request Pending 3. See description for bit 0. Reset type: SYSRSn
2	TRP2	R	0h	Transmission Request Pending 2. See description for bit 0. Reset type: SYSRSn
1	TRP1	R	0h	Transmission Request Pending 1. See description for bit 0. Reset type: SYSRSn

Table 45-67. MCAN_TXBRP Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	TRP0	R	0h	Transmission Request Pending 0. Each Tx Buffer has its own Transmission Request Pending bit. The bits are set via register TXBAR. The bits are reset after a requested transmission has completed or has been cancelled via register TXBCR. TXBRP bits are set only for those Tx Buffers configured via TXBC. After a TXBRP bit has been set, a Tx scan is started to check for the pending Tx request with the highest priority (Tx Buffer with lowest Message ID). A cancellation request resets the corresponding transmission request pending bit of register TXBRP. In case a transmission has already been started when a cancellation is requested, this is done at the end of the transmission, regardless whether the transmission was successful or not. The cancellation request bits are reset directly after the corresponding TXBRP bit has been reset. After a cancellation has been requested, a finished cancellation is signalled via TXBCF - after successful transmission together with the corresponding TXBTO bit - when the transmission has not yet been started at the point of cancellation - when the transmission has been aborted due to lost arbitration - when an error occurred during frame transmission In DAR mode all transmissions are automatically cancelled if they are not successful. The corresponding TXBCF bit is set for all unsuccessful transmissions. 0 No transmission request pending 1 Transmission request pending Note: TXBRP bits which are set while a Tx scan is in progress are not considered during this particular Tx scan. In case a cancellation is requested for such a Tx Buffer, this Add Request is cancelled immediately, the corresponding TXBRP bit is reset. Reset type: SYSRSn

45.7.4.38 MCAN_TXBAR Register (Offset (x8) = D0h, Offset (x16) = 68h) [Reset = 0h]

MCAN_TXBAR is shown in Figure 45-73 and described in Table 45-68.

Return to the Summary Table.

MCAN Tx Buffer Add Request

Figure 45-73. MCAN_TXBAR Register

31	30	29	28	27	26	25	24
AR31	AR30	AR29	AR28	AR27	AR26	AR25	AR24
R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h
23	22	21	20	19	18	17	16
AR23	AR22	AR21	AR20	AR19	AR18	AR17	AR16
R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h
15	14	13	12	11	10	9	8
AR15	AR14	AR13	AR12	AR11	AR10	AR9	AR8
R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h
7	6	5	4	3	2	1	0
AR7	AR6	AR5	AR4	AR3	AR2	AR1	AR0
R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h

Table 45-68. MCAN_TXBAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AR31	R/WQ	0h	Add Request 31. See description for bit 0. Reset type: SYSRSn
30	AR30	R/WQ	0h	Add Request 30. See description for bit 0. Reset type: SYSRSn
29	AR29	R/WQ	0h	Add Request 29. See description for bit 0. Reset type: SYSRSn
28	AR28	R/WQ	0h	Add Request 28. See description for bit 0. Reset type: SYSRSn
27	AR27	R/WQ	0h	Add Request 27. See description for bit 0. Reset type: SYSRSn
26	AR26	R/WQ	0h	Add Request 26. See description for bit 0. Reset type: SYSRSn
25	AR25	R/WQ	0h	Add Request 25. See description for bit 0. Reset type: SYSRSn
24	AR24	R/WQ	0h	Add Request 24. See description for bit 0. Reset type: SYSRSn
23	AR23	R/WQ	0h	Add Request 23. See description for bit 0. Reset type: SYSRSn
22	AR22	R/WQ	0h	Add Request 22. See description for bit 0. Reset type: SYSRSn
21	AR21	R/WQ	0h	Add Request 21. See description for bit 0. Reset type: SYSRSn
20	AR20	R/WQ	0h	Add Request 20. See description for bit 0. Reset type: SYSRSn
19	AR19	R/WQ	0h	Add Request 19. See description for bit 0. Reset type: SYSRSn
18	AR18	R/WQ	0h	Add Request 18. See description for bit 0. Reset type: SYSRSn
17	AR17	R/WQ	0h	Add Request 17. See description for bit 0. Reset type: SYSRSn

Table 45-68. MCAN_TXBAR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	AR16	R/WQ	0h	Add Request 16. See description for bit 0. Reset type: SYSRSn
15	AR15	R/WQ	0h	Add Request 15. See description for bit 0. Reset type: SYSRSn
14	AR14	R/WQ	0h	Add Request 14. See description for bit 0. Reset type: SYSRSn
13	AR13	R/WQ	0h	Add Request 13. See description for bit 0. Reset type: SYSRSn
12	AR12	R/WQ	0h	Add Request 12. See description for bit 0. Reset type: SYSRSn
11	AR11	R/WQ	0h	Add Request 11. See description for bit 0. Reset type: SYSRSn
10	AR10	R/WQ	0h	Add Request 10. See description for bit 0. Reset type: SYSRSn
9	AR9	R/WQ	0h	Add Request 9. See description for bit 0. Reset type: SYSRSn
8	AR8	R/WQ	0h	Add Request 8. See description for bit 0. Reset type: SYSRSn
7	AR7	R/WQ	0h	Add Request 7. See description for bit 0. Reset type: SYSRSn
6	AR6	R/WQ	0h	Add Request 6. See description for bit 0. Reset type: SYSRSn
5	AR5	R/WQ	0h	Add Request 5. See description for bit 0. Reset type: SYSRSn
4	AR4	R/WQ	0h	Add Request 4. See description for bit 0. Reset type: SYSRSn
3	AR3	R/WQ	0h	Add Request 3. See description for bit 0. Reset type: SYSRSn
2	AR2	R/WQ	0h	Add Request 2. See description for bit 0. Reset type: SYSRSn
1	AR1	R/WQ	0h	Add Request 1. See description for bit 0. Reset type: SYSRSn
0	AR0	R/WQ	0h	Add Request 0. Each Tx Buffer has its own Add Request bit. Writing a '1' will set the corresponding Add Request bit writing a '0' has no impact. This enables the Host to set transmission requests for multiple Tx Buffers with one write to TXBAR. TXBAR bits are set only for those Tx Buffers configured via TXBC. When no Tx scan is running, the bits are reset immediately, else the bits remain set until the Tx scan process has completed. 0 No transmission request added 1 Transmission requested added Note: If an add request is applied for a Tx Buffer with pending transmission request (corresponding TXBRP bit already set), this add request is ignored. Qualified Write is possible only with CCCR.CCE='0' Reset type: SYSRSn

45.7.4.39 MCAN_TXBCR Register (Offset (x8) = D4h, Offset (x16) = 6Ah) [Reset = 0h]

MCAN_TXBCR is shown in Figure 45-74 and described in Table 45-69.

Return to the Summary Table.

MCAN Tx Buffer Cancellation Request

Figure 45-74. MCAN_TXBCR Register

31	30	29	28	27	26	25	24
CR31	CR30	CR29	CR28	CR27	CR26	CR25	CR24
R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h
23	22	21	20	19	18	17	16
CR23	CR22	CR21	CR20	CR19	CR18	CR17	CR16
R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h
15	14	13	12	11	10	9	8
CR15	CR14	CR13	CR12	CR11	CR10	CR9	CR8
R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h
7	6	5	4	3	2	1	0
CR7	CR6	CR5	CR4	CR3	CR2	CR1	CR0
R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h

Table 45-69. MCAN_TXBCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CR31	R/WQ	0h	Cancellation Request 31. See description for bit 0. Reset type: SYSRSn
30	CR30	R/WQ	0h	Cancellation Request 30. See description for bit 0. Reset type: SYSRSn
29	CR29	R/WQ	0h	Cancellation Request 29. See description for bit 0. Reset type: SYSRSn
28	CR28	R/WQ	0h	Cancellation Request 28. See description for bit 0. Reset type: SYSRSn
27	CR27	R/WQ	0h	Cancellation Request 27. See description for bit 0. Reset type: SYSRSn
26	CR26	R/WQ	0h	Cancellation Request 26. See description for bit 0. Reset type: SYSRSn
25	CR25	R/WQ	0h	Cancellation Request 25. See description for bit 0. Reset type: SYSRSn
24	CR24	R/WQ	0h	Cancellation Request 24. See description for bit 0. Reset type: SYSRSn
23	CR23	R/WQ	0h	Cancellation Request 23. See description for bit 0. Reset type: SYSRSn
22	CR22	R/WQ	0h	Cancellation Request 22. See description for bit 0. Reset type: SYSRSn
21	CR21	R/WQ	0h	Cancellation Request 21. See description for bit 0. Reset type: SYSRSn
20	CR20	R/WQ	0h	Cancellation Request 20. See description for bit 0. Reset type: SYSRSn
19	CR19	R/WQ	0h	Cancellation Request 19. See description for bit 0. Reset type: SYSRSn
18	CR18	R/WQ	0h	Cancellation Request 18. See description for bit 0. Reset type: SYSRSn
17	CR17	R/WQ	0h	Cancellation Request 17. See description for bit 0. Reset type: SYSRSn

Table 45-69. MCAN_TXBCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	CR16	R/WQ	0h	Cancellation Request 16. See description for bit 0. Reset type: SYSRSn
15	CR15	R/WQ	0h	Cancellation Request 15. See description for bit 0. Reset type: SYSRSn
14	CR14	R/WQ	0h	Cancellation Request 14. See description for bit 0. Reset type: SYSRSn
13	CR13	R/WQ	0h	Cancellation Request 13. See description for bit 0. Reset type: SYSRSn
12	CR12	R/WQ	0h	Cancellation Request 12. See description for bit 0. Reset type: SYSRSn
11	CR11	R/WQ	0h	Cancellation Request 11. See description for bit 0. Reset type: SYSRSn
10	CR10	R/WQ	0h	Cancellation Request 10. See description for bit 0. Reset type: SYSRSn
9	CR9	R/WQ	0h	Cancellation Request 9. See description for bit 0. Reset type: SYSRSn
8	CR8	R/WQ	0h	Cancellation Request 8. See description for bit 0. Reset type: SYSRSn
7	CR7	R/WQ	0h	Cancellation Request 7. See description for bit 0. Reset type: SYSRSn
6	CR6	R/WQ	0h	Cancellation Request 6. See description for bit 0. Reset type: SYSRSn
5	CR5	R/WQ	0h	Cancellation Request 5. See description for bit 0. Reset type: SYSRSn
4	CR4	R/WQ	0h	Cancellation Request 4. See description for bit 0. Reset type: SYSRSn
3	CR3	R/WQ	0h	Cancellation Request 3. See description for bit 0. Reset type: SYSRSn
2	CR2	R/WQ	0h	Cancellation Request 2. See description for bit 0. Reset type: SYSRSn
1	CR1	R/WQ	0h	Cancellation Request 1. See description for bit 0. Reset type: SYSRSn
0	CR0	R/WQ	0h	Cancellation Request 0. Each Tx Buffer has its own Cancellation Request bit. Writing a '1' will set the corresponding Cancellation Request bit writing a '0' has no impact. This enables the Host to set cancellation requests for multiple Tx Buffers with one write to TXBCR. TXBCR bits are set only for those Tx Buffers configured via TXBC. The bits remain set until the corresponding bit of TXBRP is reset. 0 No cancellation pending 1 Cancellation pending Qualified Write is possible only with CCCR.CCE='0' Reset type: SYSRSn

45.7.4.40 MCAN_TXBTO Register (Offset (x8) = D8h, Offset (x16) = 6Ch) [Reset = 0h]

MCAN_TXBTO is shown in Figure 45-75 and described in Table 45-70.

Return to the Summary Table.

MCAN Tx Buffer Transmission Occurred

Figure 45-75. MCAN_TXBTO Register

31	30	29	28	27	26	25	24
TO31	TO30	TO29	TO28	TO27	TO26	TO25	TO24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
TO23	TO22	TO21	TO20	TO19	TO18	TO17	TO16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
TO15	TO14	TO13	TO12	TO11	TO10	TO9	TO8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
TO7	TO6	TO5	TO4	TO3	TO2	TO1	TO0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 45-70. MCAN_TXBTO Register Field Descriptions

Bit	Field	Type	Reset	Description
31	TO31	R	0h	Transmission Occurred 31. See description for bit 0. Reset type: SYSRSn
30	TO30	R	0h	Transmission Occurred 30. See description for bit 0. Reset type: SYSRSn
29	TO29	R	0h	Transmission Occurred 29. See description for bit 0. Reset type: SYSRSn
28	TO28	R	0h	Transmission Occurred 28. See description for bit 0. Reset type: SYSRSn
27	TO27	R	0h	Transmission Occurred 27. See description for bit 0. Reset type: SYSRSn
26	TO26	R	0h	Transmission Occurred 26. See description for bit 0. Reset type: SYSRSn
25	TO25	R	0h	Transmission Occurred 25. See description for bit 0. Reset type: SYSRSn
24	TO24	R	0h	Transmission Occurred 24. See description for bit 0. Reset type: SYSRSn
23	TO23	R	0h	Transmission Occurred 23. See description for bit 0. Reset type: SYSRSn
22	TO22	R	0h	Transmission Occurred 22. See description for bit 0. Reset type: SYSRSn
21	TO21	R	0h	Transmission Occurred 21. See description for bit 0. Reset type: SYSRSn
20	TO20	R	0h	Transmission Occurred 20. See description for bit 0. Reset type: SYSRSn
19	TO19	R	0h	Transmission Occurred 19. See description for bit 0. Reset type: SYSRSn
18	TO18	R	0h	Transmission Occurred 18. See description for bit 0. Reset type: SYSRSn
17	TO17	R	0h	Transmission Occurred 17. See description for bit 0. Reset type: SYSRSn

Table 45-70. MCAN_TXBTO Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	TO16	R	0h	Transmission Occurred 16. See description for bit 0. Reset type: SYSRSn
15	TO15	R	0h	Transmission Occurred 15. See description for bit 0. Reset type: SYSRSn
14	TO14	R	0h	Transmission Occurred 14. See description for bit 0. Reset type: SYSRSn
13	TO13	R	0h	Transmission Occurred 13. See description for bit 0. Reset type: SYSRSn
12	TO12	R	0h	Transmission Occurred 12. See description for bit 0. Reset type: SYSRSn
11	TO11	R	0h	Transmission Occurred 11. See description for bit 0. Reset type: SYSRSn
10	TO10	R	0h	Transmission Occurred 10. See description for bit 0. Reset type: SYSRSn
9	TO9	R	0h	Transmission Occurred 9. See description for bit 0. Reset type: SYSRSn
8	TO8	R	0h	Transmission Occurred 8. See description for bit 0. Reset type: SYSRSn
7	TO7	R	0h	Transmission Occurred 7. See description for bit 0. Reset type: SYSRSn
6	TO6	R	0h	Transmission Occurred 6. See description for bit 0. Reset type: SYSRSn
5	TO5	R	0h	Transmission Occurred 5. See description for bit 0. Reset type: SYSRSn
4	TO4	R	0h	Transmission Occurred 4. See description for bit 0. Reset type: SYSRSn
3	TO3	R	0h	Transmission Occurred 3. See description for bit 0. Reset type: SYSRSn
2	TO2	R	0h	Transmission Occurred 2. See description for bit 0. Reset type: SYSRSn
1	TO1	R	0h	Transmission Occurred 1. See description for bit 0. Reset type: SYSRSn
0	TO0	R	0h	Transmission Occurred 0. Each Tx Buffer has its own Transmission Occurred bit. The bits are set when the corresponding TXBRP bit is cleared after a successful transmission. The bits are reset when a new transmission is requested by writing a '1' to the corresponding bit of register TXBAR. 0 No transmission occurred 1 Transmission occurred Reset type: SYSRSn

45.7.4.41 MCAN_TXBCF Register (Offset (x8) = DCh, Offset (x16) = 6Eh) [Reset = 0h]

MCAN_TXBCF is shown in Figure 45-76 and described in Table 45-71.

Return to the Summary Table.

MCAN Tx Buffer Cancellation Finished

Figure 45-76. MCAN_TXBCF Register

31	30	29	28	27	26	25	24
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 45-71. MCAN_TXBCF Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CF31	R	0h	Cancellation Finished 31. See description for bit 0. Reset type: SYSRSn
30	CF30	R	0h	Cancellation Finished 30. See description for bit 0. Reset type: SYSRSn
29	CF29	R	0h	Cancellation Finished 29. See description for bit 0. Reset type: SYSRSn
28	CF28	R	0h	Cancellation Finished 28. See description for bit 0. Reset type: SYSRSn
27	CF27	R	0h	Cancellation Finished 27. See description for bit 0. Reset type: SYSRSn
26	CF26	R	0h	Cancellation Finished 26. See description for bit 0. Reset type: SYSRSn
25	CF25	R	0h	Cancellation Finished 25. See description for bit 0. Reset type: SYSRSn
24	CF24	R	0h	Cancellation Finished 24. See description for bit 0. Reset type: SYSRSn
23	CF23	R	0h	Cancellation Finished 23. See description for bit 0. Reset type: SYSRSn
22	CF22	R	0h	Cancellation Finished 22. See description for bit 0. Reset type: SYSRSn
21	CF21	R	0h	Cancellation Finished 21. See description for bit 0. Reset type: SYSRSn
20	CF20	R	0h	Cancellation Finished 20. See description for bit 0. Reset type: SYSRSn
19	CF19	R	0h	Cancellation Finished 19. See description for bit 0. Reset type: SYSRSn
18	CF18	R	0h	Cancellation Finished 18. See description for bit 0. Reset type: SYSRSn
17	CF17	R	0h	Cancellation Finished 17. See description for bit 0. Reset type: SYSRSn

Table 45-71. MCAN_TXBCF Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	CF16	R	0h	Cancellation Finished 16. See description for bit 0. Reset type: SYSRSn
15	CF15	R	0h	Cancellation Finished 15. See description for bit 0. Reset type: SYSRSn
14	CF14	R	0h	Cancellation Finished 14. See description for bit 0. Reset type: SYSRSn
13	CF13	R	0h	Cancellation Finished 13. See description for bit 0. Reset type: SYSRSn
12	CF12	R	0h	Cancellation Finished 12. See description for bit 0. Reset type: SYSRSn
11	CF11	R	0h	Cancellation Finished 11. See description for bit 0. Reset type: SYSRSn
10	CF10	R	0h	Cancellation Finished 10. See description for bit 0. Reset type: SYSRSn
9	CF9	R	0h	Cancellation Finished 9. See description for bit 0. Reset type: SYSRSn
8	CF8	R	0h	Cancellation Finished 8. See description for bit 0. Reset type: SYSRSn
7	CF7	R	0h	Cancellation Finished 7. See description for bit 0. Reset type: SYSRSn
6	CF6	R	0h	Cancellation Finished 6. See description for bit 0. Reset type: SYSRSn
5	CF5	R	0h	Cancellation Finished 5. See description for bit 0. Reset type: SYSRSn
4	CF4	R	0h	Cancellation Finished 4. See description for bit 0. Reset type: SYSRSn
3	CF3	R	0h	Cancellation Finished 3. See description for bit 0. Reset type: SYSRSn
2	CF2	R	0h	Cancellation Finished 2. See description for bit 0. Reset type: SYSRSn
1	CF1	R	0h	Cancellation Finished 1. See description for bit 0. Reset type: SYSRSn
0	CF0	R	0h	Cancellation Finished 0. Each Tx Buffer has its own Cancellation Finished bit. The bits are set when the corresponding TXBRP bit is cleared after a cancellation was requested via TXBCR. In case the corresponding TXBRP bit was not set at the point of cancellation, CF is set immediately. The bits are reset when a new transmission is requested by writing a '1' to the corresponding bit of register TXBAR. 0 No transmit buffer cancellation 1 Transmit buffer cancellation finished Reset type: SYSRSn

45.7.4.42 MCAN_TXBTIE Register (Offset (x8) = E0h, Offset (x16) = 70h) [Reset = 0h]

MCAN_TXBTIE is shown in Figure 45-77 and described in Table 45-72.

Return to the Summary Table.

Each Tx Buffer has its own Transmission Interrupt Enable bit.

Figure 45-77. MCAN_TXBTIE Register

31	30	29	28	27	26	25	24
TIE31	TIE30	TIE29	TIE28	TIE27	TIE26	TIE25	TIE24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
TIE23	TIE22	TIE21	TIE20	TIE19	TIE18	TIE17	TIE16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
TIE15	TIE14	TIE13	TIE12	TIE11	TIE10	TIE9	TIE8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TIE7	TIE6	TIE5	TIE4	TIE3	TIE2	TIE1	TIE0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 45-72. MCAN_TXBTIE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	TIE31	R/W	0h	Transmission Interrupt Enable 31. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
30	TIE30	R/W	0h	Transmission Interrupt Enable 30. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
29	TIE29	R/W	0h	Transmission Interrupt Enable 29. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
28	TIE28	R/W	0h	Transmission Interrupt Enable 28. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
27	TIE27	R/W	0h	Transmission Interrupt Enable 27. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
26	TIE26	R/W	0h	Transmission Interrupt Enable 26. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn

Table 45-72. MCAN_TXBTIE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	TIE25	R/W	0h	Transmission Interrupt Enable 25. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
24	TIE24	R/W	0h	Transmission Interrupt Enable 24. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
23	TIE23	R/W	0h	Transmission Interrupt Enable 23. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
22	TIE22	R/W	0h	Transmission Interrupt Enable 22. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
21	TIE21	R/W	0h	Transmission Interrupt Enable 21. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
20	TIE20	R/W	0h	Transmission Interrupt Enable 20. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
19	TIE19	R/W	0h	Transmission Interrupt Enable 19. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
18	TIE18	R/W	0h	Transmission Interrupt Enable 18. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
17	TIE17	R/W	0h	Transmission Interrupt Enable 17. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
16	TIE16	R/W	0h	Transmission Interrupt Enable 16. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
15	TIE15	R/W	0h	Transmission Interrupt Enable 15. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
14	TIE14	R/W	0h	Transmission Interrupt Enable 14. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn

Table 45-72. MCAN_TXBTIE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	TIE13	R/W	0h	Transmission Interrupt Enable 13. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
12	TIE12	R/W	0h	Transmission Interrupt Enable 12. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
11	TIE11	R/W	0h	Transmission Interrupt Enable 11. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
10	TIE10	R/W	0h	Transmission Interrupt Enable 10. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
9	TIE9	R/W	0h	Transmission Interrupt Enable 9. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
8	TIE8	R/W	0h	Transmission Interrupt Enable 8. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
7	TIE7	R/W	0h	Transmission Interrupt Enable 7. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
6	TIE6	R/W	0h	Transmission Interrupt Enable 6. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
5	TIE5	R/W	0h	Transmission Interrupt Enable 5. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
4	TIE4	R/W	0h	Transmission Interrupt Enable 4. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
3	TIE3	R/W	0h	Transmission Interrupt Enable 3. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
2	TIE2	R/W	0h	Transmission Interrupt Enable 2. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn

Table 45-72. MCAN_TXBTIE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	TIE1	R/W	0h	Transmission Interrupt Enable 1. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
0	TIE0	R/W	0h	Transmission Interrupt Enable 0. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn

45.7.4.43 MCAN_TXBCIE Register (Offset (x8) = E4h, Offset (x16) = 72h) [Reset = 0h]

MCAN_TXBCIE is shown in Figure 45-78 and described in Table 45-73.

Return to the Summary Table.

MCAN Tx Buffer Cancellation Finished Interrupt Enable

Figure 45-78. MCAN_TXBCIE Register

31	30	29	28	27	26	25	24
CFIE31	CFIE30	CFIE29	CFIE28	CFIE27	CFIE26	CFIE25	CFIE24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
CFIE23	CFIE22	CFIE21	CFIE20	CFIE19	CFIE18	CFIE17	CFIE16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
CFIE15	CFIE14	CFIE13	CFIE12	CFIE11	CFIE10	CFIE9	CFIE8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CFIE7	CFIE6	CFIE5	CFIE4	CFIE3	CFIE2	CFIE1	CFIE0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 45-73. MCAN_TXBCIE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CFIE31	R/W	0h	Cancellation Finished Interrupt Enable 31. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
30	CFIE30	R/W	0h	Cancellation Finished Interrupt Enable 30. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
29	CFIE29	R/W	0h	Cancellation Finished Interrupt Enable 29. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
28	CFIE28	R/W	0h	Cancellation Finished Interrupt Enable 28. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
27	CFIE27	R/W	0h	Cancellation Finished Interrupt Enable 27. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
26	CFIE26	R/W	0h	Cancellation Finished Interrupt Enable 26. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn

Table 45-73. MCAN_TXBCIE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	CFIE25	R/W	0h	Cancellation Finished Interrupt Enable 25. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
24	CFIE24	R/W	0h	Cancellation Finished Interrupt Enable 24. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
23	CFIE23	R/W	0h	Cancellation Finished Interrupt Enable 23. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
22	CFIE22	R/W	0h	Cancellation Finished Interrupt Enable 22. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
21	CFIE21	R/W	0h	Cancellation Finished Interrupt Enable 21. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
20	CFIE20	R/W	0h	Cancellation Finished Interrupt Enable 20. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
19	CFIE19	R/W	0h	Cancellation Finished Interrupt Enable 19. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
18	CFIE18	R/W	0h	Cancellation Finished Interrupt Enable 18. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
17	CFIE17	R/W	0h	Cancellation Finished Interrupt Enable 17. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
16	CFIE16	R/W	0h	Cancellation Finished Interrupt Enable 16. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
15	CFIE15	R/W	0h	Cancellation Finished Interrupt Enable 15. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
14	CFIE14	R/W	0h	Cancellation Finished Interrupt Enable 14. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn

Table 45-73. MCAN_TXBCIE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	CFIE13	R/W	0h	Cancellation Finished Interrupt Enable 13. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
12	CFIE12	R/W	0h	Cancellation Finished Interrupt Enable 12. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
11	CFIE11	R/W	0h	Cancellation Finished Interrupt Enable 11. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
10	CFIE10	R/W	0h	Cancellation Finished Interrupt Enable 10. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
9	CFIE9	R/W	0h	Cancellation Finished Interrupt Enable 9. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
8	CFIE8	R/W	0h	Cancellation Finished Interrupt Enable 8. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
7	CFIE7	R/W	0h	Cancellation Finished Interrupt Enable 7. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
6	CFIE6	R/W	0h	Cancellation Finished Interrupt Enable 6. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
5	CFIE5	R/W	0h	Cancellation Finished Interrupt Enable 5. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
4	CFIE4	R/W	0h	Cancellation Finished Interrupt Enable 4. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
3	CFIE3	R/W	0h	Cancellation Finished Interrupt Enable 3. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
2	CFIE2	R/W	0h	Cancellation Finished Interrupt Enable 2. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn

Table 45-73. MCAN_TXBCIE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	CFIE1	R/W	0h	Cancellation Finished Interrupt Enable 1. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
0	CFIE0	R/W	0h	Cancellation Finished Interrupt Enable 0. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn

45.7.4.44 MCAN_TXEFC Register (Offset (x8) = F0h, Offset (x16) = 78h) [Reset = 0h]

MCAN_TXEFC is shown in Figure 45-79 and described in Table 45-74.

Return to the Summary Table.

MCAN Tx Event FIFO Configuration

Figure 45-79. MCAN_TXEFC Register

31	30	29	28	27	26	25	24
RESERVED		EFWM					
R-0h		R/WQ-0h					
23	22	21	20	19	18	17	16
RESERVED		EFS					
R-0h		R/WQ-0h					
15	14	13	12	11	10	9	8
EFSA							
R/WQ-0h							
7	6	5	4	3	2	1	0
EFSA						RESERVED	
R/WQ-0h						R-0h	

Table 45-74. MCAN_TXEFC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29-24	EFWM	R/WQ	0h	Event FIFO Watermark 0 Watermark interrupt disabled 1-32 Level for Tx Event FIFO watermark interrupt (IR.TEFW) >32 Watermark interrupt disabled Reset type: SYSRSn
23-22	RESERVED	R	0h	Reserved
21-16	EFS	R/WQ	0h	Event FIFO Size. The Tx Event FIFO elements are indexed from 0 to EFS - 1. 0 Tx Event FIFO disabled 1-32 Number of Tx Event FIFO elements >32 Values greater than 32 are interpreted as 32 Reset type: SYSRSn
15-2	EFSA	R/WQ	0h	Event FIFO Start Address. Start address of Tx Event FIFO in Message RAM (32-bit word address). Reset type: SYSRSn
1-0	RESERVED	R	0h	Reserved

45.7.4.45 MCAN_TXEFS Register (Offset (x8) = F4h, Offset (x16) = 7Ah) [Reset = 0h]

MCAN_TXEFS is shown in Figure 45-80 and described in Table 45-75.

Return to the Summary Table.

MCAN Tx Event FIFO Status

Figure 45-80. MCAN_TXEFS Register

31	30	29	28	27	26	25	24
RESERVED						TEFL	EFF
R-0h						R-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED			EFPI				
R-0h			R-0h				
15	14	13	12	11	10	9	8
RESERVED			EFGI				
R-0h			R-0h				
7	6	5	4	3	2	1	0
RESERVED		EFFL					
R-0h		R-0h					

Table 45-75. MCAN_TXEFS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved
25	TEFL	R	0h	Tx Event FIFO Element Lost. This bit is a copy of interrupt flag IR.TEFL. When IR.TEFL is reset, this bit is also reset. 0 No Tx Event FIFO element lost 1 Tx Event FIFO element lost, also set after write attempt to Tx Event FIFO of size zero. Reset type: SYSRSn
24	EFF	R	0h	Event FIFO Full 0 Tx Event FIFO not full 1 Tx Event FIFO full Reset type: SYSRSn
23-21	RESERVED	R	0h	Reserved
20-16	EFPI	R	0h	Event FIFO Put Index.Tx Event FIFO write index pointer, range 0 to 31. Reset type: SYSRSn
15-13	RESERVED	R	0h	Reserved
12-8	EFGI	R	0h	Event FIFO Get Index. Tx Event FIFO read index pointer, range 0 to 31. Reset type: SYSRSn
7-6	RESERVED	R	0h	Reserved
5-0	EFFL	R	0h	Event FIFO Fill Level. Number of elements stored in Tx Event FIFO, range 0 to 32. Reset type: SYSRSn

45.7.4.46 MCAN_TXEFA Register (Offset (x8) = F8h, Offset (x16) = 7Ch) [Reset = 0h]

MCAN_TXEFA is shown in Figure 45-81 and described in Table 45-76.

Return to the Summary Table.

MCAN Tx Event FIFO Acknowledge

Figure 45-81. MCAN_TXEFA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											EFAI				
R-0h																											R/W-0h				

Table 45-76. MCAN_TXEFA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4-0	EFAI	R/W	0h	Event FIFO Acknowledge Index. After the Host has read an element or a sequence of elements from the Tx Event FIFO it has to write the index of the last element read from Tx Event FIFO to EFAI. This will set the Tx Event FIFO Get Index TXEFS.EFGI to EFAI + 1 and update the Event FIFO Fill Level TXEFS.EFFL. Reset type: SYSRSn

45.7.5 MCAN_ERROR_REGS Registers

Table 45-77 lists the memory-mapped registers for the MCAN_ERROR_REGS registers. All register offset addresses not listed in Table 45-77 should be considered as reserved locations and the register contents should not be modified.

Table 45-77. MCAN_ERROR_REGS Registers

Offset (x8)	Offset (x16)	Acronym	Register Name	Write Protection	Section
0h	0h	MCANERR_REV	MCAN Error Aggregator Revision Register		Go
8h	4h	MCANERR_VECTOR	MCAN ECC Vector Register		Go
Ch	6h	MCANERR_STAT	MCAN Error Misc Status		Go
10h	8h	MCANERR_WRAP_REV	MCAN ECC Wrapper Revision Register		Go
14h	Ah	MCANERR_CTRL	MCAN ECC Control		Go
18h	Ch	MCANERR_ERR_CTRL1	MCAN ECC Error Control 1 Register		Go
1Ch	Eh	MCANERR_ERR_CTRL2	MCAN ECC Error Control 2 Register		Go
20h	10h	MCANERR_ERR_STAT1	MCAN ECC Error Status 1 Register		Go
24h	12h	MCANERR_ERR_STAT2	MCAN ECC Error Status 2 Register		Go
28h	14h	MCANERR_ERR_STAT3	MCAN ECC Error Status 3 Register		Go
3Ch	1Eh	MCANERR_SEC_EOI	MCAN Single Error Corrected End of Interrupt Register		Go
40h	20h	MCANERR_SEC_STATUS	MCAN Single Error Corrected Interrupt Status Register		Go
80h	40h	MCANERR_SEC_ENABLE_SET	MCAN Single Error Corrected Interrupt Enable Set Register		Go
C0h	60h	MCANERR_SEC_ENABLE_CLR	MCAN Single Error Corrected Interrupt Enable Clear Register		Go
13Ch	9Eh	MCANERR_DED_EOI	MCAN Double Error Detected End of Interrupt Register		Go
140h	A0h	MCANERR_DED_STATUS	MCAN Double Error Detected Interrupt Status Register		Go
180h	C0h	MCANERR_DED_ENABLE_SET	MCAN Double Error Detected Interrupt Enable Set Register		Go
1C0h	E0h	MCANERR_DED_ENABLE_CLR	MCAN Double Error Detected Interrupt Enable Clear Register		Go
200h	100h	MCANERR_AGGR_ENABLE_SET	MCAN Error Aggregator Enable Set Register		Go
204h	102h	MCANERR_AGGR_ENABLE_CLR	MCAN Error Aggregator Enable Clear Register		Go
208h	104h	MCANERR_AGGR_STATUS_SET	MCAN Error Aggregator Status Set Register		Go
20Ch	106h	MCANERR_AGGR_STATUS_CLR	MCAN Error Aggregator Status Clear Register		Go

Complex bit access types are encoded to fit into small table cells. Table 45-78 shows the codes that are used for access types in this section.

Table 45-78. MCAN_ERROR_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s

Table 45-78. MCAN_ERROR_REGS Access Type Codes (continued)

Access Type	Code	Description
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
WD	W D	Write Decrement. Decrements the specified bit field by the amount written.
WI	W I	Write Increment. Increments the specified bit field by the amount written.
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

45.7.5.1 MCANERR_REV Register (Offset (x8) = 0h, Offset (x16) = 0h) [Reset = 66A0E200h]

MCANERR_REV is shown in Figure 45-82 and described in Table 45-79.

Return to the Summary Table.

MCAN Error Aggregator Revision Register

Figure 45-82. MCANERR_REV Register

31	30	29	28	27	26	25	24
SCHEME		RESERVED		MODULE_ID			
R-1h		R-2h		R-6A0h			
23	22	21	20	19	18	17	16
MODULE_ID							
R-6A0h							
15	14	13	12	11	10	9	8
RESERVED						REVMAJ	
R-1Ch						R-2h	
7	6	5	4	3	2	1	0
RESERVED		REVMIN					
R-0h		R-0h					

Table 45-79. MCANERR_REV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	SCHEME	R	1h	PID Register Scheme Reset type: SYSRSn
29-28	RESERVED	R	2h	Reserved
27-16	MODULE_ID	R	6A0h	Module Identification Number Reset type: SYSRSn
15-11	RESERVED	R	1Ch	Reserved
10-8	REVMAJ	R	2h	Major Revision of the Error Aggregator Reset type: SYSRSn
7-6	RESERVED	R	0h	Reserved
5-0	REVMIN	R	0h	Minor Revision of the Error Aggregator Reset type: SYSRSn

45.7.5.2 MCANERR_VECTOR Register (Offset (x8) = 8h, Offset (x16) = 4h) [Reset = 0h]

MCANERR_VECTOR is shown in Figure 45-83 and described in Table 45-80.

Return to the Summary Table.

Each error detection and correction (EDC) controller has a bank of error registers (offsets 0x10 - 0x3B) associated with it. These registers are accessed via an internal serial bus (SVBUS). To access them through the ECC aggregator the controller ID desired must be written to the ECC_VECTOR field, together with the RD_SVBUS trigger and RD_SVBUS_ADDRESS bit field. This initiates the serial read which consummates by setting the RD_SVBUS_DONE bit. At this point the addressed register may be read by a normal CPU read of the appropriate offset address.

Figure 45-83. MCANERR_VECTOR Register

31	30	29	28	27	26	25	24
RESERVED							RD_SVBUS_DONE
R-0h							R-0h
23	22	21	20	19	18	17	16
RD_SVBUS_ADDRESS							
R/W-0h							
15	14	13	12	11	10	9	8
RD_SVBUS	RESERVED				ECC_VECTOR		
R-0/W1S-0h	R-0h				R/W-0h		
7	6	5	4	3	2	1	0
ECC_VECTOR							
R/W-0h							

Table 45-80. MCANERR_VECTOR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24	RD_SVBUS_DONE	R	0h	Read Completion Flag Reset type: SYSRSn
23-16	RD_SVBUS_ADDRESS	R/W	0h	Read Address Offset Reset type: SYSRSn
15	RD_SVBUS	R-0/W1S	0h	Read Trigger Reset type: SYSRSn
14-11	RESERVED	R	0h	Reserved
10-0	ECC_VECTOR	R/W	0h	ECC RAM ID. Each error detection and correction (EDC) controller has a bank of error registers (offsets 0x10 - 0x3B) associated with it. These registers are accessed via an internal serial bus (SVBUS). To access them through the ECC aggregator the controller ID desired must be written to the ECC_VECTOR field, together with the RD_SVBUS trigger and RD_SVBUS_ADDRESS bit field. This initiates the serial read which consummates by setting the RD_SVBUS_DONE bit. At this point the addressed register may be read by a normal CPU read of the appropriate offset address. 0x000 Message RAM ECC controller is selected Others Reserved (do not use) Subsequent writes through the SVBUS (offsets 0x10 - 0x3B) have a delayed completion. To avoid conflicts, perform a read back of a register within this range after writing. Reset type: SYSRSn

45.7.5.3 MCANERR_STAT Register (Offset (x8) = Ch, Offset (x16) = 6h) [Reset = 2h]

MCANERR_STAT is shown in Figure 45-84 and described in Table 45-81.

Return to the Summary Table.

MCAN Error Misc Status

Figure 45-84. MCANERR_STAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					NUM_RAMS										
R-0h																					R-2h										

Table 45-81. MCANERR_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Reserved
10-0	NUM_RAMS	R	2h	Number of RAMs. Number of ECC RAMs serviced by the aggregator. Reset type: SYSRSn

45.7.5.4 MCANERR_WRAP_REV Register (Offset (x8) = 10h, Offset (x16) = 8h) [Reset = 66A42A02h]

MCANERR_WRAP_REV is shown in Figure 45-85 and described in Table 45-82.

Return to the Summary Table.

This register is accessed through the ECC aggregator via an internal serial bus. To access, the appropriate procedure must be first followed in the MCAN ECC Vector Register.

Figure 45-85. MCANERR_WRAP_REV Register

31	30	29	28	27	26	25	24
SCHEME		RESERVED		MODULE_ID			
R-1h		R-2h		R-6A4h			
23	22	21	20	19	18	17	16
MODULE_ID							
R-6A4h							
15	14	13	12	11	10	9	8
RESERVED					REVMAJ		
R-5h					R-2h		
7	6	5	4	3	2	1	0
RESERVED		REVMIN					
R-0h		R-2h					

Table 45-82. MCANERR_WRAP_REV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	SCHEME	R	1h	PID Register Scheme Reset type: SYSRSn
29-28	RESERVED	R	2h	Reserved
27-16	MODULE_ID	R	6A4h	Module Identification Number Reset type: SYSRSn
15-11	RESERVED	R	5h	Reserved
10-8	REVMAJ	R	2h	Major Revision of the Error Aggregator Reset type: SYSRSn
7-6	RESERVED	R	0h	Reserved
5-0	REVMIN	R	2h	Minor Revision of the Error Aggregator Reset type: SYSRSn

45.7.5.5 MCANERR_CTRL Register (Offset (x8) = 14h, Offset (x16) = Ah) [Reset = 187h]

MCANERR_CTRL is shown in Figure 45-86 and described in Table 45-83.

Return to the Summary Table.

This register is accessed through the ECC aggregator via an internal serial bus. To access, the appropriate procedure must be first followed in the MCAN ECC Vector Register.

Figure 45-86. MCANERR_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							CHECK_SVBUS_TIMEOUT
R-0h							R/W-1h
7	6	5	4	3	2	1	0
RESERVED	ERROR_ONCE	FORCE_N_ROW	FORCE_DED	FORCE_SEC	ENABLE_RMW	ECC_CHECK	ECC_ENABLE
R/W-1h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h	R/W-1h	R/W-1h

Table 45-83. MCANERR_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	CHECK_SVBUS_TIMEOUT	R/W	1h	Enables Serial VBUS timeout mechanism Reset type: SYSRSn
7	RESERVED	R/W	1h	Reserved
6	ERROR_ONCE	R/W	0h	If this bit is set, the FORCE_SEC/FORCE_DED will inject an error to the specified row only once. The FORCE_SEC bit will be cleared once a writeback happens. If writeback is not enabled, this error will be cleared the cycle following the read when the data is corrected. For double-bit errors, the FORCE_DED bit will be cleared the cycle following the double-bit error. Any subsequent reads will not force an error. Reset type: SYSRSn
5	FORCE_N_ROW	R/W	0h	Enable single/double-bit error on the next RAM read, regardless of the MCANERR_ERR_CTRL1.ECC_ROW setting. For write through mode, this applies to writes as well as reads. Reset type: SYSRSn
4	FORCE_DED	R/W	0h	Force double-bit error. Cleared the cycle following the error if ERROR_ONCE is asserted. For write through mode, this applies to writes as well as reads. MCANERR_ERR_CTRL1 and MCANERR_ERR_CTRL2 should be configured prior to setting this bit. Reset type: SYSRSn
3	FORCE_SEC	R/W	0h	Force single-bit error. Cleared on a writeback or the cycle following the error if ERROR_ONCE is asserted. For write through mode, this applies to writes as well as reads. MCANERR_ERR_CTRL1 and MCANERR_ERR_CTRL2 should be configured prior to setting this bit. Reset type: SYSRSn
2	ENABLE_RMW	R/W	1h	Enable read-modify-write on partial word writes Reset type: SYSRSn

Table 45-83. MCANERR_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	ECC_CHECK	R/W	1h	Enable ECC Check. ECC is completely bypassed if both ECC_ENABLE and ECC_CHECK are '0'. Reset type: SYSRSn
0	ECC_ENABLE	R/W	1h	Enable ECC Generation Reset type: SYSRSn

45.7.5.6 MCANERR_ERR_CTRL1 Register (Offset (x8) = 18h, Offset (x16) = Ch) [Reset = 0h]

MCANERR_ERR_CTRL1 is shown in Figure 45-87 and described in Table 45-84.

Return to the Summary Table.

This register is accessed through the ECC aggregator via an internal serial bus. To access, the appropriate procedure must be first followed in the MCAN ECC Vector Register.

Figure 45-87. MCANERR_ERR_CTRL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_ROW																															
R/W-0h																															

Table 45-84. MCANERR_ERR_CTRL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ECC_ROW	R/W	0h	Row address where FORCE_SEC or FORCE_DED needs to be applied. This is ignored if FORCE_N_ROW is set. Reset type: SYSRSn

45.7.5.7 MCANERR_ERR_CTRL2 Register (Offset (x8) = 1Ch, Offset (x16) = Eh) [Reset = 0h]

MCANERR_ERR_CTRL2 is shown in Figure 45-88 and described in Table 45-85.

Return to the Summary Table.

This register is accessed through the ECC aggregator via an internal serial bus. To access, the appropriate procedure must be first followed in the MCAN ECC Vector Register.

Figure 45-88. MCANERR_ERR_CTRL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_BIT2																ECC_BIT1															
R/W-0h																R/W-0h															

Table 45-85. MCANERR_ERR_CTRL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	ECC_BIT2	R/W	0h	Second column/data bit that needs to be flipped when FORCE_DED is set Reset type: SYSRSn
15-0	ECC_BIT1	R/W	0h	Column/Data bit that needs to be flipped when FORCE_SEC or FORCE_DED is set Reset type: SYSRSn

45.7.5.8 MCANERR_ERR_STAT1 Register (Offset (x8) = 20h, Offset (x16) = 10h) [Reset = 0h]

MCANERR_ERR_STAT1 is shown in Figure 45-89 and described in Table 45-86.

Return to the Summary Table.

This register is accessed through the ECC aggregator via an internal serial bus. To access, the appropriate procedure must be first followed in the MCAN ECC Vector Register.

Figure 45-89. MCANERR_ERR_STAT1 Register

31	30	29	28	27	26	25	24
ECC_BIT1							
R-0h							
23	22	21	20	19	18	17	16
ECC_BIT1							
R-0h							
15	14	13	12	11	10	9	8
CLR_CTRL_REG_ERROR	RESERVED		CLR_ECC_OTHER	CLR_ECC_DED		CLR_ECC_SEC	
R/W1S-0h	R/WD-0h		R/W1C-0h	R/WD-0h		R/WD-0h	
7	6	5	4	3	2	1	0
CTRL_REG_ERROR	RESERVED		ECC_OTHER	ECC_DED		ECC_SEC	
R/W1S-0h	R/WI-0h		R/W1S-0h	R/WI-0h		R/WI-0h	

Table 45-86. MCANERR_ERR_STAT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	ECC_BIT1	R	0h	ECC Error Bit Position. Indicates the bit position in the RAM data that is in error on an SEC error. Only valid on an SEC error. 0 Bit 0 is in error 1 Bit 1 is in error 2 Bit 2 is in error 3 Bit 3 is in error ... 31 Bit 31 is in error >32 Invalid Reset type: SYSRSn
15	CLR_CTRL_REG_ERROR	R/W1S	0h	Writing a '1' clears the CTRL_REG_ERROR bit Reset type: SYSRSn
14-13	RESERVED	R/WD	0h	Reserved
12	CLR_ECC_OTHER	R/W1C	0h	Writing a '1' clears the ECC_OTHER bit. Reset type: SYSRSn
11-10	CLR_ECC_DED	R/WD	0h	Clear ECC_DED. A write of a non-zero value to this bit field decrements the ECC_DED bit field by the value provided. Reset type: SYSRSn
9-8	CLR_ECC_SEC	R/WD	0h	Clear ECC_SEC. A write of a non-zero value to this bit field decrements the ECC_SEC bit field by the value provided. Reset type: SYSRSn
7	CTRL_REG_ERROR	R/W1S	0h	Control Register Error. A bit field in the control register is in an ambiguous state. This means that the redundancy registers have detected a state where not all values are the same and has defaulted to the reset state. S/W needs to re-write these registers to a known state. A write of 1 will set this interrupt flag. Reset type: SYSRSn
6-5	RESERVED	R/WI	0h	Reserved

Table 45-86. MCANERR_ERR_STAT1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	ECC_OTHER	R/W1S	0h	SEC While Writeback Error Status 0 No SEC error while writeback pending 1 Indicates that successive single-bit errors have occurred while a writeback is still pending Reset type: SYSRSn
3-2	ECC_DED	R/WI	0h	Double Bit Error Detected Status. A 2-bit saturating counter of the number of DED errors that have occurred since last cleared. 0 No double-bit error detected 1 One double-bit error was detected 2 Two double-bit errors were detected 3 Three double-bit errors were detected A write of a non-zero value to this bit field increments it by the value provided. Reset type: SYSRSn
1-0	ECC_SEC	R/WI	0h	Single Bit Error Corrected Status. A 2-bit saturating counter of the number of SEC errors that have occurred since last cleared. 0 No single-bit error detected 1 One single-bit error was detected and corrected 2 Two single-bit errors were detected and corrected 3 Three single-bit errors were detected and corrected A write of a non-zero value to this bit field increments it by the value provided. Reset type: SYSRSn

45.7.5.9 MCANERR_ERR_STAT2 Register (Offset (x8) = 24h, Offset (x16) = 12h) [Reset = 0h]

MCANERR_ERR_STAT2 is shown in Figure 45-90 and described in Table 45-87.

Return to the Summary Table.

This register is accessed through the ECC aggregator via an internal serial bus. To access, the appropriate procedure must be first followed in the MCAN ECC Vector Register.

Figure 45-90. MCANERR_ERR_STAT2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_ROW																															
R-0h																															

Table 45-87. MCANERR_ERR_STAT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ECC_ROW	R	0h	Indicates the row address where the single or double-bit error occurred. This value is address offset/4. Reset type: SYSRSn

45.7.5.10 MCANERR_ERR_STAT3 Register (Offset (x8) = 28h, Offset (x16) = 14h) [Reset = 0h]

MCANERR_ERR_STAT3 is shown in Figure 45-91 and described in Table 45-88.

Return to the Summary Table.

This register is accessed through the ECC aggregator via an internal serial bus. To access, the appropriate procedure must be first followed in the MCAN ECC Vector Register.

Figure 45-91. MCANERR_ERR_STAT3 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						CLR_SVBUS_TIMEOUT	RESERVED
R-0h						R-0/W1C-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED						SVBUS_TIMEOUT	WB_PEND
R-0h						R-0/W1S-0h	R-0h

Table 45-88. MCANERR_ERR_STAT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	CLR_SVBUS_TIMEOUT	R-0/W1C	0h	Write 1 to clear the Serial VBUS Timeout Flag Reset type: SYSRSn
8-2	RESERVED	R	0h	Reserved
1	SVBUS_TIMEOUT	R-0/W1S	0h	Serial VBUS Timeout Flag. Write 1 to set. Reset type: SYSRSn
0	WB_PEND	R	0h	Delayed Write Back Pending Status 0 No write back pending 1 An ECC data correction write back is pending Reset type: SYSRSn

45.7.5.11 MCANERR_SEC_EOI Register (Offset (x8) = 3Ch, Offset (x16) = 1Eh) [Reset = 0h]

MCANERR_SEC_EOI is shown in Figure 45-92 and described in Table 45-89.

Return to the Summary Table.

MCAN Single Error Corrected End of Interrupt Register

Figure 45-92. MCANERR_SEC_EOI Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EOI_WR
R-0h							R-0/W1S-0h

Table 45-89. MCANERR_SEC_EOI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EOI_WR	R-0/W1S	0h	Write to this register indicates that software has acknowledged the pending interrupt and the next interrupt can be sent to the host. Note that a write to the MCANERR_ERR_STAT1.CLR_ECC_SEC goes through the SVBUS and has a delayed completion. To avoid an additional interrupt, read the MCANERR_ERR_STAT1 register back prior to writing to this bit field. Reset type: SYSRSn

45.7.5.12 MCANERR_SEC_STATUS Register (Offset (x8) = 40h, Offset (x16) = 20h) [Reset = 0h]

MCANERR_SEC_STATUS is shown in Figure 45-93 and described in Table 45-90.

Return to the Summary Table.

MCAN Single Error Corrected Interrupt Status Register

Figure 45-93. MCANERR_SEC_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	MSGMEM_PEND
R-0h						R-0/W1S-0h	R-0/W1S-0h

Table 45-90. MCANERR_SEC_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	RESERVED	R-0/W1S	0h	Reserved
0	MSGMEM_PEND	R-0/W1S	0h	Message RAM SEC Interrupt Pending 0 No SEC interrupt is pending 1 SEC interrupt is pending Reset type: SYSRSn

45.7.5.13 MCANERR_SEC_ENABLE_SET Register (Offset (x8) = 80h, Offset (x16) = 40h) [Reset = 0h]

MCANERR_SEC_ENABLE_SET is shown in Figure 45-94 and described in Table 45-91.

Return to the Summary Table.

MCAN Single Error Corrected Interrupt Enable Set Register

Figure 45-94. MCANERR_SEC_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	MSGMEM_ENABLE_SET
R-0h						R/W1S-0h	R/W1S-0h

Table 45-91. MCANERR_SEC_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	RESERVED	R/W1S	0h	Reserved
0	MSGMEM_ENABLE_SET	R/W1S	0h	Message RAM SEC Interrupt Pending Enable Set. Writing a 1 to this bit enables the Message RAM SEC error interrupts. Writing a 0 has no effect. Reads return the corresponding enable bit's current value. Reset type: SYSRSn

45.7.5.14 MCANERR_SEC_ENABLE_CLR Register (Offset (x8) = C0h, Offset (x16) = 60h) [Reset = 0h]

MCANERR_SEC_ENABLE_CLR is shown in Figure 45-95 and described in Table 45-92.

Return to the Summary Table.

MCAN Single Error Corrected Interrupt Enable Clear Register

Figure 45-95. MCANERR_SEC_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	MSGMEM_ENABLE_CLR
R-0h						R/W1C-0h	R/W1C-0h

Table 45-92. MCANERR_SEC_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	RESERVED	R/W1C	0h	Reserved
0	MSGMEM_ENABLE_CLR	R/W1C	0h	Message RAM SEC Interrupt Pending Enable Clear. Writing a 1 to this bit disables the Message RAM SEC error interrupts. Writing a 0 has no effect. Reads return the corresponding enable bit's current value. Reset type: SYSRSn

45.7.5.15 MCANERR_DED_EOI Register (Offset (x8) = 13Ch, Offset (x16) = 9Eh) [Reset = 0h]

MCANERR_DED_EOI is shown in Figure 45-96 and described in Table 45-93.

Return to the Summary Table.

MCAN Double Error Detected End of Interrupt Register

Figure 45-96. MCANERR_DED_EOI Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EOI_WR
R-0h							R-0/W1S-0h

Table 45-93. MCANERR_DED_EOI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EOI_WR	R-0/W1S	0h	Write to this register indicates that software has acknowledged the pending interrupt and the next interrupt can be sent to the host. Note that a write to the MCANERR_ERR_STAT1.CLR_ECC_DED goes through the SVBUS and has a delayed completion. To avoid an additional interrupt, read the MCANERR_ERR_STAT1 register back prior to writing to this bit field. Reset type: SYSRSn

45.7.5.16 MCANERR_DED_STATUS Register (Offset (x8) = 140h, Offset (x16) = A0h) [Reset = 0h]

MCANERR_DED_STATUS is shown in Figure 45-97 and described in Table 45-94.

Return to the Summary Table.

MCAN Double Error Detected Interrupt Status Register

Figure 45-97. MCANERR_DED_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	MSGMEM_PEND
R-0h						R-0/W1S-0h	R-0/W1S-0h

Table 45-94. MCANERR_DED_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	RESERVED	R-0/W1S	0h	Reserved
0	MSGMEM_PEND	R-0/W1S	0h	Message RAM DED Interrupt Pending 0 No DED interrupt is pending 1 DED interrupt is pending Reset type: SYSRSn

45.7.5.17 MCANERR_DED_ENABLE_SET Register (Offset (x8) = 180h, Offset (x16) = C0h) [Reset = 0h]

MCANERR_DED_ENABLE_SET is shown in Figure 45-98 and described in Table 45-95.

Return to the Summary Table.

MCAN Double Error Detected Interrupt Enable Set Register

Figure 45-98. MCANERR_DED_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	MSGMEM_ENABLE_SET
R-0h						R/W1S-0h	R/W1S-0h

Table 45-95. MCANERR_DED_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	RESERVED	R/W1S	0h	Reserved
0	MSGMEM_ENABLE_SET	R/W1S	0h	Message RAM DED Interrupt Pending Enable Set. Writing a 1 to this bit enables the Message RAM DED error interrupts. Writing a 0 has no effect. Reads return the corresponding enable bit's current value. Reset type: SYSRSn

45.7.5.18 MCANERR_DED_ENABLE_CLR Register (Offset (x8) = 1C0h, Offset (x16) = E0h) [Reset = 0h]

MCANERR_DED_ENABLE_CLR is shown in Figure 45-99 and described in Table 45-96.

Return to the Summary Table.

MCAN Double Error Detected Interrupt Enable Clear Register

Figure 45-99. MCANERR_DED_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	MSGMEM_ENABLE_CLR
R-0h						R/W1C-0h	R/W1C-0h

Table 45-96. MCANERR_DED_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	RESERVED	R/W1C	0h	Reserved
0	MSGMEM_ENABLE_CLR	R/W1C	0h	Message RAM DED Interrupt Pending Enable Clear. Writing a 1 to this bit disables the Message RAM DED error interrupts. Writing a 0 has no effect. Reads return the corresponding enable bit's current value. Reset type: SYSRSn

45.7.5.19 MCANERR_AGGR_ENABLE_SET Register (Offset (x8) = 200h, Offset (x16) = 100h) [Reset = 0h]

MCANERR_AGGR_ENABLE_SET is shown in Figure 45-100 and described in Table 45-97.

Return to the Summary Table.

MCAN Error Aggregator Enable Set Register

Figure 45-100. MCANERR_AGGR_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ENABLE_TIMEOUT_SET	ENABLE_PARITY_SET
R-0h						R/W1S-0h	R/W1S-0h

Table 45-97. MCANERR_AGGR_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	ENABLE_TIMEOUT_SET	R/W1S	0h	Write 1 to enable timeout errors. Reads return the corresponding enable bit's current value. Reset type: SYSRSn
0	ENABLE_PARITY_SET	R/W1S	0h	Write 1 to enable parity errors. Reads return the corresponding enable bit's current value. Reset type: SYSRSn

45.7.5.20 MCANERR_AGGR_ENABLE_CLR Register (Offset (x8) = 204h, Offset (x16) = 102h) [Reset = 0h]

MCANERR_AGGR_ENABLE_CLR is shown in Figure 45-101 and described in Table 45-98.

Return to the Summary Table.

MCAN Error Aggregator Enable Clear Register

Figure 45-101. MCANERR_AGGR_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ENABLE_TIMEOUT_CLR	ENABLE_PARITY_CLR
R-0h						R/W1C-0h	R/W1C-0h

Table 45-98. MCANERR_AGGR_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	ENABLE_TIMEOUT_CLR	R/W1C	0h	Write 1 to disable timeout errors. Reads return the corresponding enable bit's current value. Reset type: SYSRSn
0	ENABLE_PARITY_CLR	R/W1C	0h	Write 1 to disable parity errors. Reads return the corresponding enable bit's current value. Reset type: SYSRSn

45.7.5.21 MCANERR_AGGR_STATUS_SET Register (Offset (x8) = 208h, Offset (x16) = 104h) [Reset = 0h]

MCANERR_AGGR_STATUS_SET is shown in Figure 45-102 and described in Table 45-99.

Return to the Summary Table.

MCAN Error Aggregator Status Set Register

Figure 45-102. MCANERR_AGGR_STATUS_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				SVBUS_TIMEOUT		AGGR_PARITY_ERR	
R-0h				R/WI-0h		R/WI-0h	

Table 45-99. MCANERR_AGGR_STATUS_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-2	SVBUS_TIMEOUT	R/WI	0h	Aggregator Serial VBUS Timeout Error Status 2-bit saturating counter of the number of SVBUS timeout errors that have occurred since last cleared. 0 No timeout errors have occurred 1 One timeout error has occurred 2 Two timeout errors have occurred 3 Three timeout errors have occurred A write of a non-zero value to this bit field increments it by the value provided. Reset type: SYSRSn
1-0	AGGR_PARITY_ERR	R/WI	0h	Aggregator Parity Error Status 2-bit saturating counter of the number of parity errors that have occurred since last cleared. 0 No parity errors have occurred 1 One parity error has occurred 2 Two parity errors have occurred 3 Three parity errors have occurred A write of a non-zero value to this bit field increments it by the value provided. Reset type: SYSRSn

45.7.5.22 MCANERR_AGGR_STATUS_CLR Register (Offset (x8) = 20Ch, Offset (x16) = 106h) [Reset = 0h]

MCANERR_AGGR_STATUS_CLR is shown in Figure 45-103 and described in Table 45-100.

Return to the Summary Table.

MCAN Error Aggregator Status Clear Register

Figure 45-103. MCANERR_AGGR_STATUS_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				SVBUS_TIMEOUT		AGGR_PARITY_ERR	
R-0h				R/WD-0h		R/WD-0h	

Table 45-100. MCANERR_AGGR_STATUS_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-2	SVBUS_TIMEOUT	R/WD	0h	Aggregator Serial VBUS Timeout Error Status 2-bit saturating counter of the number of SVBUS timeout errors that have occurred since last cleared. 0 No timeout errors have occurred 1 One timeout error has occurred 2 Two timeout errors have occurred 3 Three timeout errors have occurred A write of a non-zero value to this bit field decrements it by the value provided. Reset type: SYSRSn
1-0	AGGR_PARITY_ERR	R/WD	0h	Aggregator Parity Error Status 2-bit saturating counter of the number of parity errors that have occurred since last cleared. 0 No parity errors have occurred 1 One parity error has occurred 2 Two parity errors have occurred 3 Three parity errors have occurred A write of a non-zero value to this bit field decrements it by the value provided. Reset type: SYSRSn

This page intentionally left blank.

Chapter 46

Connectivity Manager Inter-Integrated Circuit (I2C) Module

This chapter describes the features and operation of the inter-integrated circuit (I2C) module in the Connectivity Manager subsystem.

46.1 Introduction

The Inter-Integrated Circuit (I2C) bus provides bidirectional data transfer through a two-wire design; a serial data line (SDA) and a serial clock line (SCL), and interfaces to external I2C devices such as serial memory (RAMs and ROMs), networking devices, LCDs, tone generators, and so on. The I2C bus may also be used for system testing and diagnostic purposes in product development and manufacturing.

46.1.1 Features

The I2C modules support the following features:

- Devices on the I2C bus can be designated as either a master or a slave.
  - Supports both transmitting and receiving data as either a master or a slave
  - Supports simultaneous master and slave operation
- Four I2C modes:
  - Master transmit
  - Master receive
  - Slave transmit
  - Slave receive
- Receive FIFO and Transmitter FIFO (8 deep x 8 bits FIFO)
  - FIFOs can be independently assigned to master or slave
- Four transmission speeds:
  - Standard (100 kbps)
  - Fast mode (400 kbps)
  - Fast-mode plus (1 Mbps)
  - High-speed mode (3.33 Mbps)
- Glitch suppression
- SMBus support through software
  - Clock low time-out interrupt
  - Dual slave address capability
  - Quick command capability
- Master and slave interrupt generation
  - Master generates interrupts when a transmit or receive operation completes (or aborts because of an error)
  - Slave generates interrupts when data has been transferred or requested by a master or when a START or STOP condition is detected
- Master with arbitration and clock synchronization, multiple-master support, and 7-bit addressing mode
- Efficient transfers using a Micro Direct Memory Access Controller (µDMA)
  - Separate channels for transmit and receive
  - Ability to execute single data transfers or burst data transfers using the RX and TX FIFOs in the I2C

46.1.2 Block Diagram

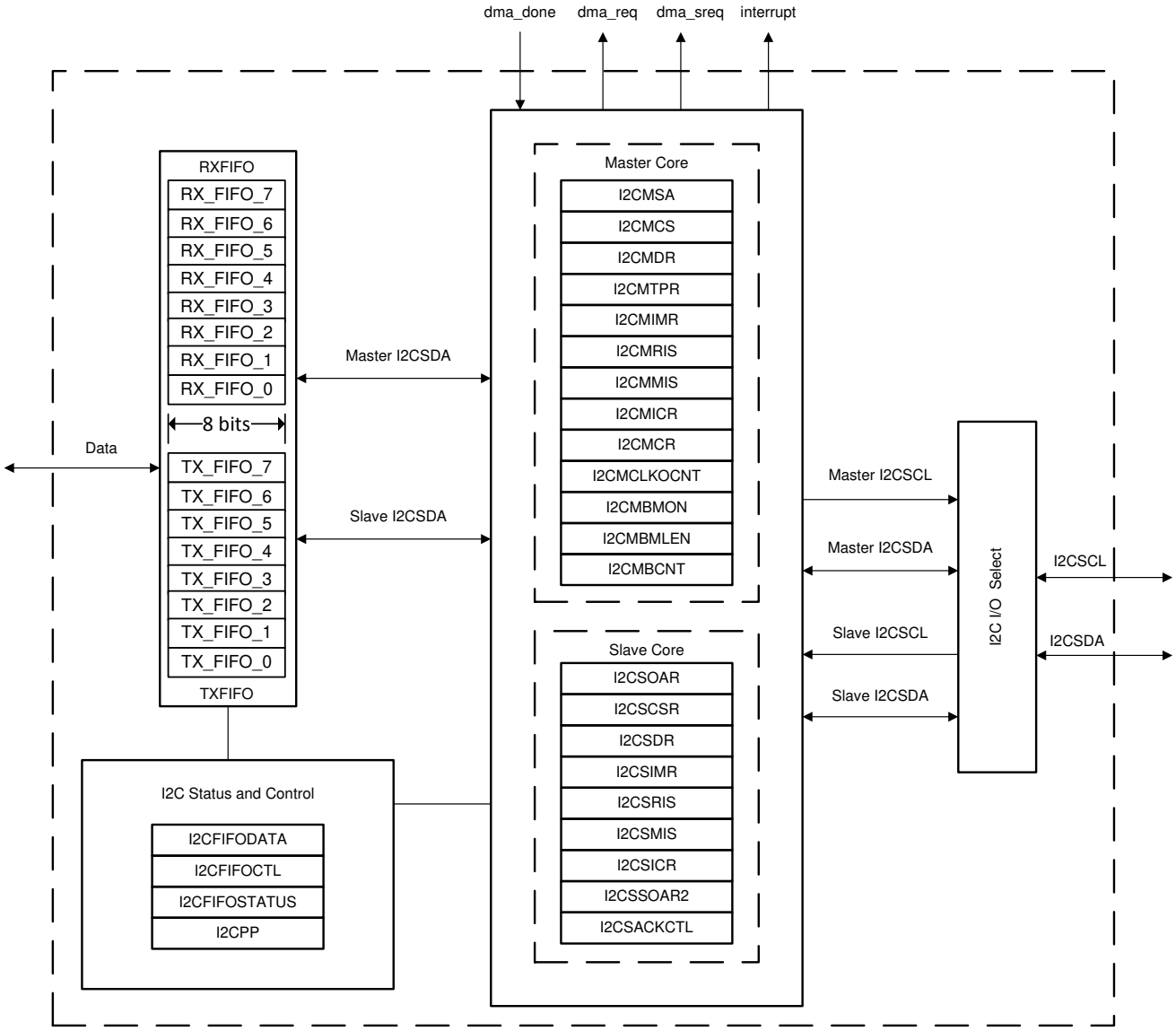


Figure 46-1. I2C Block Diagram

46.2 Functional Description

Each I2C module is comprised of both master and slave functions and is identified by a unique address. A master-initiated communication generates the clock signal, SCL. For proper operation, the SDA pin must be configured as an open-drain signal. Due to the internal circuitry that supports high-speed operation, the SCL pin must not be configured as an open-drain signal, although the internal circuitry causes it to act as if it were an open-drain signal. Both SDA and SCL signals must be connected to a positive supply voltage using a pullup resistor. Figure 46-2 shows a typical I2C bus configuration. See the I2C bus specification and user manual to determine the size of the pullups required for proper operation.

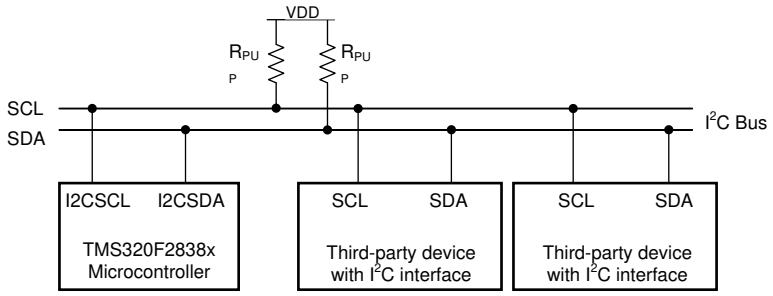


Figure 46-2. I2C Bus Configuration

46.2.1 I2C Bus Functional Overview

The I2C bus uses only two signals: SDA and SCL (named CM-I2CA_SCL and CM-I2CA_SDA in TMS320F2838x microcontrollers). SDA is the bidirectional SDA and SCL is the bidirectional serial clock line. The bus is considered idle when both lines are high.

Every transaction on the I2C bus is nine bits long, consisting of eight data bits and a single acknowledge bit. The number of bytes per transfer (defined as the time between a valid START and STOP condition, described in Figure 46-3) is unrestricted, but each data byte must be followed by an acknowledge bit, and data must be transferred MSB first. When a receiver cannot receive another complete byte, it can hold the clock line SCL low and force the transmitter into a wait state. The data transfer continues when the receiver releases the clock SCL.

46.2.1.1 START and STOP Conditions

The protocol of the I2C bus defines two states to begin and end a transaction: START and STOP. A high-to-low transition on the SDA line while the SCL is high is defined as a START condition, and a low-to-high transition on the SDA line while SCL is high is defined as a STOP condition. The bus is considered busy after a START condition and free after a STOP condition. See Figure 46-3.

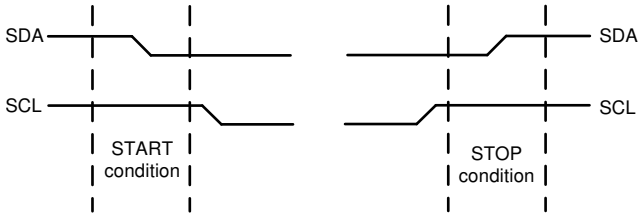


Figure 46-3. START and STOP Conditions

The STOP bit determines if the cycle stops at the end of the data cycle or continues on to a repeated START condition. To generate a single transmit cycle, the I2C Master Slave Address (I2CMSA) register is written with the desired address, the R/S bit is cleared, and the Control register is written with ACK = X (0 or 1), STOP = 1, START = 1, and RUN = 1 to perform the operation and stop. When the operation is completed (or aborted due an error), the interrupt pin becomes active and the data may be read from the I2C Master Data (I2CMDR) register. When the I2C module operates in master receiver mode, the ACK bit is normally set causing the I2C bus controller to transmit an acknowledge automatically after each byte. This bit must be cleared when the I2C bus controller requires no further data to be transmitted from the slave transmitter.

When operating in slave mode, the STARTRIS and STOPRIS bits in the I2C Slave Raw Interrupt Status (I2CSRIS) register indicate detection of start and stop conditions on the bus and the I2C Slave Masked Interrupt Status (I2CSMIS) register can be configured to allow STARTRIS and STOPRIS to be promoted to controller interrupts (when interrupts are enabled).

46.2.1.2 Data Format With 7-Bit Address

Data transfers follow the format in Figure 46-4. After the START condition, a slave address is transmitted. This address is 7 bits long followed by an eighth bit, which is a data direction bit (R/S bit in the I2CMSA register). If the R/S bit is clear, it indicates a transmit operation (send), and if it is set, it indicates a request for data (receive). A data transfer is always terminated by a STOP condition generated by the master; however, a master can initiate communications with another device on the bus by generating a repeated START condition and addressing another slave without first generating a STOP condition. Various combinations of receive and transmit formats are then possible within a single transfer.

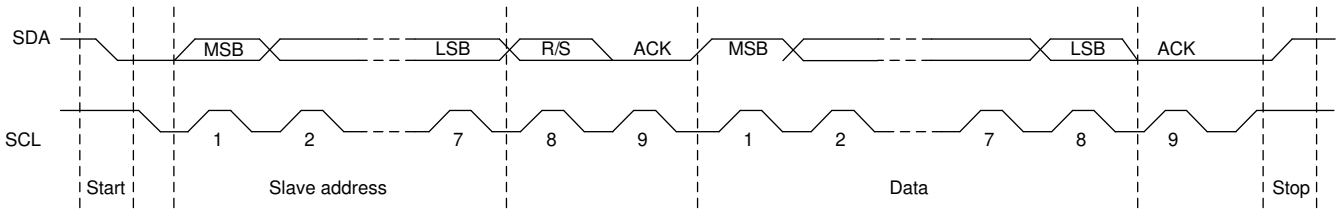


Figure 46-4. Complete Data Transfer With a 7-Bit Address

The first seven bits of the first byte make up the slave address (see Figure 46-5). The eighth bit determines the direction of the message. A zero in the R/S position of the first byte means that the master transmits (sends) data to the selected slave, and a one in this position means that the master receives data from the slave.

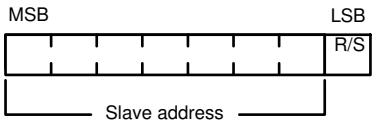


Figure 46-5. R/S Bit in First Byte

46.2.1.3 Data Validity

The data on the SDA line must be stable during the high period of the clock, and the data line can change only when SCL is low (see Figure 46-6).

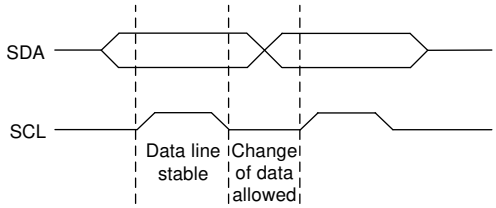


Figure 46-6. Data Validity During Bit Transfer on the I2C Bus

46.2.1.4 Acknowledge

All bus transactions have a required acknowledge clock cycle that is generated by the master. During the acknowledge cycle, the transmitter (which can be the master or slave) releases the SDA line. To acknowledge the transaction, the receiver must pull down SDA during the acknowledge clock cycle. The data transmitted out by the receiver during the acknowledge cycle must comply with the data validity requirements described in Section 46.2.1.3.

When a slave receiver does not acknowledge the slave address, SDA must be left high by the slave so that the master can generate a STOP condition and abort the current transfer. If the master device is acting as a receiver during a transfer, it is responsible for acknowledging each transfer made by the slave. Because the master controls the number of bytes in the transfer, it signals the end of data to the slave transmitter by not

generating an acknowledge on the last data byte. The slave transmitter must then release SDA to allow the master to generate the STOP or a repeated START condition.

If the slave is required to provide a manual ACK or NACK, the I2C Slave ACK Control (I2CSACKCTL) register allows the slave to NACK for invalid data or command or ACK for valid data or command. When this operation is enabled, the MCU slave module I2C clock is pulled low after the last data bit until this register is written with the indicated response.

46.2.1.5 Repeated START

The I2C master module can execute a repeated START (transmit or receive) after an initial transfer has occurred.

Note

When reading the I2CMCS register to check the BUSY bit, also read the ADRACK and DATACK bits, because these are cleared on register read, and status may be lost if they are not checked on every read of the register.

Alternatively, the NACKRIS bit of the I2CMRIS register can be used to monitor NACK status.

For more information on repeated START, see Figure 46-12 and Figure 46-13.

46.2.1.5.1 Repeated Start for Master Transmit

A repeated start sequence for a master transmit is as follows:

1. When the device is in the idle state, the master writes the slave address to the I2CMSA register and configures the R/S bit for the desired transfer type.
2. Data is written to the I2CMDR register.
3. When the BUSY bit in the I2CMCS register is 0, the master writes 0x3 to the I2CMCS register to initiate a transfer.
4. The master does not generate a STOP condition but instead writes another slave address to the I2CMSA register and then writes 0x3 to initiate the repeated START.

46.2.1.5.2 Repeated Start for Master Receive

A repeated start sequence for a master receive is similar:

1. When the device is in idle, the master writes the slave address to the I2CMSA register and configures the R/S bit for the desired transfer type.
2. The master reads data from the I2CMDR register.
3. When the BUSY bit in the I2CMCS register is 0, the master writes 0x3 to the I2CMCS_WRITE register to initiate a transfer.
4. The master does not generate a STOP condition but instead writes another slave address to the I2CMSA register and then writes 0x3 to initiate the repeated START.

46.2.1.6 Clock Low Time-out (CLTO)

The I2C slave can extend the transaction by pulling the clock low periodically to create a slow bit transfer rate. The I2C module has a 12-bit programmable counter that is used to track how long the clock has been held low. The upper 8 bits of the count value are software programmable through the I2C Master Clock Low Time-out Count (I2CMCLKOCNT) register. The lower four bits are not user visible and are 0x0. The CNTL value programmed in the I2CMCLKOCNT register has to be greater than 0x01. The application can program the eight most-significant bits of the counter to reflect the acceptable cumulative low period in transaction. The count is loaded at the START condition and counts down on each falling edge of the internal bus clock of the master. Note that the internal bus clock generated for this counter keeps running at the programmed I2C speed even if SCL is held low on the bus. Upon reaching terminal count, the masterstate machine forces ABORT on the bus by issuing a STOP condition at the instance of SCL and SDA release.

As an example, if an I2C module is operating at 100-kHz speed, programming the I2CMCLKOCNT register to 0xDA translates to the value 0xDA0 since the lower four bits are set to 0x0. This translates to a decimal value of 3488 clocks or a cumulative clock low period of 34.88 ms at 100 kHz.

The CLKRIS bit in the I2C Master Raw Interrupt Status (I2CMRIS) register is set when the clock time-out period is reached, allowing the master to start corrective action to resolve the remote slave state. In addition, the CLKTO bit in the I2C Master Control/Status (I2CMCS) register is set; this bit is cleared when a STOP condition is sent or during the I2C master reset. The status of the raw SDA and SCL signals are readable by software through the SDA and SCL bits in the I2C Master Bus Monitor (I2CMBMON) register to help determine the state of the remote slave.

In the event of a CLTO condition, application software must choose how the application intends to attempt bus recovery. Most applications attempt to manually toggle the I2C pins to force the slave to let go of the clock signal (a common solution is to attempt to force a STOP on the bus). If a CLTO is detected before the end of a burst transfer, and the bus is successfully recovered by the master, the master hardware attempts to finish the pending burst operation. The behavior of the bus varies depending on the state of the slave after bus recovery. If the slave resumes in a state where the slave can acknowledge the master (essentially, where it was before the bus hang), it continues where it left off. However, if the slave resumes in a reset state (or if a forced STOP by the master causes the slave to enter the idle state), it may ignore the master attempt to complete the burst operation and NAK the first data byte that the master sends or requests.

Since the behavior of slaves cannot always be predicted, it is suggested that the application software always write the STOP bit in the I2C Master Configuration (I2CMCR) register during the CLTO interrupt service routine. This limits the amount of data the master attempts to send or receive upon bus recovery to a single byte, and after the single byte is on the wire, the master issues a STOP. An alternative solution is to have the application software reset the I2C peripheral before attempting to manually recover the bus. This solution allows the I2C master hardware to be returned to a known good (and idle) state before attempting to recover a stuck bus and prevents any unwanted data from appearing on the wire.

Note

The Master Clock Low Time-out counter counts for the entire time SCL is held low continuously. If SCL is deasserted at any point, the Master Clock Low Time-out Counter is reloaded with the value in the I2CMCLKOCNT register and begins counting down from this value.

46.2.1.7 Dual Address

The I2C interface supports dual address capability for the slave. The additional programmable address is provided and can be matched if enabled. In legacy mode with dual address disabled, the I2C slave provides an ACK on the bus if the address matches the OAR field in the I2CSOAR register. In dual address mode, the I2C slave provides an ACK on the bus if either the OAR field in the I2CSOAR register or the OAR2 field in the I2CSOAR2 register is matched. The enable for dual address is programmable through the OAR2EN bit in the I2CSOAR2 register and there is no disable on the legacy address.

The OAR2SEL bit in the I2CSCSR register indicates if the address that was ACKed is the alternate address or not. When this bit is clear, the bit indicates either legacy operation or no address match.

46.2.1.8 Arbitration

A master may start a transfer only if the bus is idle. It is possible for two or more master to generate a START condition within the minimum hold time of the START condition. In these situations, an arbitration scheme takes place on the SDA line, while SCL is high. During arbitration, the first of the competing master devices to place a 1 (high) on SDA, while another master transmits a 0 (low), switches off its data output stage and retires until the bus is idle again.

Arbitration can take place over several bits. The first stage is a comparison of address bits, and if both master are trying to address the same device, arbitration continues on to the comparison of data bits.

If arbitration is lost when the I2C master is initiating a BURST with the TX FIFO enabled, the application should execute the following steps to correctly handle the arbitration loss:

1. Flush and disable the TX FIFO.
2. Clear and mask the TXFE interrupt by clearing the TXFEIM bit in the I2CMIMR register.

Once the bus is IDLE, the TXFIFO can be filled and enabled, the TXFE bit can be unmasked and a new BURST transaction can be initiated.

46.2.1.9 Glitch Suppression in Multi-Master Configuration

When a multi-master configuration is being used, the PULSEL bit in the I2CMTPR register can be programmed to provide glitch suppression on the SCL and SDA lines and assure proper signal values. The glitch suppression value is in terms of buffered system clocks. Note that all signals will be delayed internally when glitch suppression is nonzero. For example, if PULSEL is set to 0x7, 31 clocks should be added onto the calculation for the expected transaction time.

46.2.1.10 SMBus Operation

The SMBus interface is based on the I2C protocol; however, some differences exist between the two. These differences must be handled through software to make sure the SMBus protocol, including timing specifications, is met. Note that the SMBus 2.0 specification limits the maximum frequency of the interface to 100 KHz; as a result, I2C standard speed operation is used for SMBus.

The SMBus and I2C slave can extend the transaction, if it is not ready by pulling the clock low. The SMbus specification allows the maximum time-out for such elongated transaction to be 25 to 35 ms. The I2C specification does not have this requirement. The I2C module supports a programmable count to support clock-low time-out for the master to error out and take action as required; this feature is explained in Section 46.2.1.6. Note that if transactions are extended, a time-out period should be programmed in the I2CMCLKOCNT register, and the CLKRIS bit in the I2CMRIS register should not be masked.

Unlike the I2C slave, the SMBus slave must respond with an ACK response to its address regardless of whether it is ready or not. As a result, the I2C slave sends an ACK response to its address and a NACK response on the data byte if it is not ready. The ARBLST bit in the I2CMCS register is set if there were any issues with the transfer. In addition, the slave can send a NACK at any time to force the master to stop sending additional bytes.

The I2C interface supports µDMA for efficient data handling. The µDMA operation needs FIFOs to be enabled for appropriate transfer type to perform I2C master for burst transfers and all types of slave transfers. The I2C interface is supported by two channels: one for Rx (I2C-to-memory) and one for Tx (memory-to-I2C) transfers. See Section 46.2.5 for more information.

46.2.1.10.1 Quick Command

A quick command is a simple, compact SMBus protocol that sends an address and one bit of data in the R/S bit of the I2C header byte to communicate a command to the slave, typically a turn off or turn on. The I2C master can send a quick command by writing the target address and R/S value into the I2CMSA register followed by a write to I2CMCS with a value of 0x27. SMBus requires the slaveto be able to accept and process commands and the master to generate the quick command transactions. The master also has the capability to stop the transaction after acknowledgment from a slave.

The I2C slave requires special handling when a quick command is sent. In the case where a master sends a quick command with the R/S (data) bit cleared, the QCMDST bit in I2CSCSR is set, and the QCMDRW bit shows the data value (which, in this case, is 0) when the STOPRIS bit is set in I2CSRIS and the STOP interrupt is asserted. In this scenario, a DATARIS interrupt bit is not set. When the master sends a quick command with the R/S (data) bit set, the DATARIS bit is set to notify the slave to write a data byte to I2CSDR in which bit 7 is set. A dummy write of 0xFF to the I2CSDR register is recommended. After the write to I2CSDR, the STOP interrupt is asserted and the QCMDST and QCMDRW bits are set in the I2CSCSR register to indicate that a quick command read occurred and the last transaction was a quick command. Therefore, when the slave must receive a quick command, it expects such a command because it must write the I2CSDR with a specific value when R/S is set.

46.2.2 Available Speed Modes

The I2C bus can run in standard mode (100 kbps), fast mode (400 kbps), fast mode plus (1 Mbps) or high-speed mode (3.4 Mbps, if the correct system clock frequency is set and there is appropriate pull strength on SCL and SDA). The selected mode should match the speed of the other I2C devices on the bus.

46.2.2.1 Standard, Fast, and Fast Plus Modes

Standard, fast, and fast plus modes are selected using a value in the I2C Master Timer Period (I2CMTPR) register that results in an SCL frequency of 100 kbps for standard mode, 400 kbps for fast mode, or 1 Mbps for fast mode plus.

The I2C clock rate is determined by the parameters CLK_PRD, TIMER_PRD, SCL_LP, and SCL_HP, where:

- CLK_PRD is the system clock period
- SCL_LP is the low phase of SCL (fixed at 6)
- SCL_HP is the high phase of SCL (fixed at 4)

TIMER_PRD is the programmed value in the I2CMTPR register. This value is determined by replacing the known variables in Equation 25 and solving for TIMER_PRD. The I2C clock period is calculated as:

$$SCL\_PERIOD = 2 \times (1 + TIMER\_PRD) \times (SCL\_LP + SCL\_HP) \times CLK\_PRD \quad (25)$$

For example:

- CLK_PRD = 50 ns
- TIMER_PRD = 2
- SCL_LP = 6
- SCL_HP = 4

Yields an SCL frequency of: 1/SCL_PERIOD = 338 KHz

Table 46-1 lists examples of the timer periods that must be used to generate standard, fast mode, and fast mode plus SCL frequencies based on various system clock frequencies.

Table 46-1. Examples of I2C Master Timer Period Versus Speed Mode

System Clock	Standard Mode		Fast Mode		Fast Mode Plus	
	Timer Period	Data Rate	Timer Period	Data Rate	Timer Period	Data Rate
200 MHz	99	100 kbps	24	400 kbps	9	1 Mbps

46.2.2.2 High-Speed Mode

The I2C peripheral has support for high-speed operation as both a master and slave. High-speed mode is configured by setting the HS bit in the I2C Master Control/Status (I2CMCS) register. High-speed mode transmits data at a high bit rate with a 66.6%/33.3% duty cycle, but communication and arbitration are done at standard, fast mode, or fast-mode plus speed, depending on which is selected by the user. When the HS bit in the I2CMCS register is set, current mode pullups are enabled.

The clock period can be selected using Equation 26, but in this case, SCL_LP = 2 and SCL_HP = 1.

$$SCL\_PERIOD = 2 \times (1 + TIMER\_PRD) \times (SCL\_LP + SCL\_HP) \times CLK\_PRD \quad (26)$$

For example:

- CLK_PRD = 25 ns
- TIMER_PRD = 1
- SCL_LP = 2
- SCL_HP = 1

Yields a SCL frequency of: 1/T = 3.33 MHz

Table 46-2 gives examples of timer period and system clock in high-speed mode. Note that the HS bit in the I2CMTPR register must be set for the TPR value to be used in high-speed mode.

Table 46-2. Examples of I2C Master Timer Period in High-Speed Mode

System Clock	Timer Period	Transmission Mode
200 MHz	9	3.33 Mbps

When operating as a master, the protocol is shown in Figure 46-7. The master is responsible for sending a master code byte in either standard (100 kbps) or fast mode (400 kbps) before the code byte begins transferring in high-speed mode. The master code byte must contain data in the form of 0000.1XXX and is used to tell the slave devices to prepare for a high-speed transfer. The master code byte must never be acknowledged by a slave, since the code byte is only used to indicate that the upcoming data is going to be transferred at a higher data rate. To send the master code byte for a standard high-speed transfer, software must place the value of the master code byte into the I2CMSA register and write the I2CMCS register with a value of 0x13. If a high-speed burst transfer is required, then to send the master code byte, software must place the value of the master code byte into the I2CMSA register and write the I2CMCS register with 0x50. Either configuration places the I2C master peripheral in high-speed mode, and all subsequent transfers (until STOP) are carried out at high-speed data rate using the normal I2CMCS command bits, without setting the HS bit in the I2CMCS register. Again, setting the HS bit in the I2CMCS register is only necessary during the master code byte.

When operating as a high-speed slave, no additional software is required.

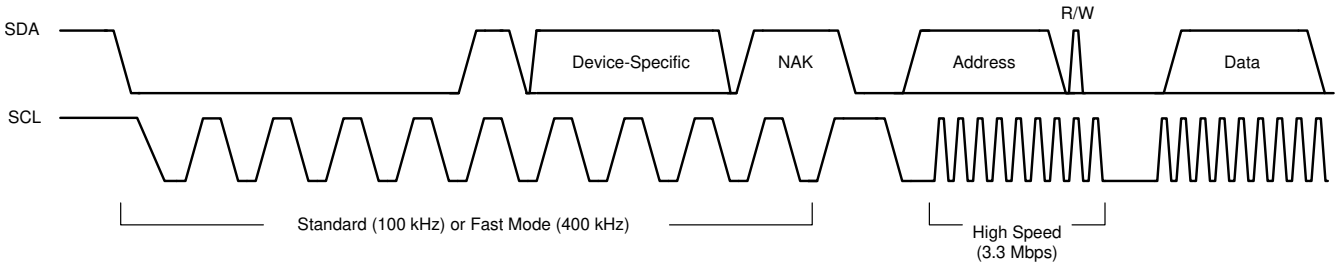


Figure 46-7. High-Speed Data Format

Note

High-speed mode is 3.4 Mbps, if the correct system clock frequency is set and there is appropriate pull strength on SCL and SDA lines.

46.2.3 Interrupts

The I2C can generate interrupts when the following conditions are observed in the master module:

- Master transaction completed (RIS bit)
- Master arbitration lost (ARBLOSTRIS bit)
- Master Address/Data NACK (NACKRIS bit)
- Master bus time-out (CLKRIS bit)
- Next byte request (RIS bit)
- Stop condition on bus detected (STOPRIS bit)
- Start condition on bus detected (STARTRIS bit)
- RX DMA interrupt pending (DMARXRIS bit)
- TX DMA interrupt pending (DMATXRIS bit)
- Trigger value for FIFO has been reached and a TX FIFO request interrupt is pending (TXRIS bit)
- Trigger value for FIFO has been reached and a RX FIFO request interrupt is pending (RXRIS bit)
- Transmit FIFO is empty (TXFERIS bit)
- Receive FIFO is full (RXFFRIS bit)

Interrupts are generated when the following conditions are observed in the slave module:

- Slave transaction received (DATARIS bit)
- Slave transaction requested (DATARIS bit)
- Slave next byte transfer request (DATARIS bit)
- Stop condition on bus detected (STOPRIS bit)
- Start condition on bus detected (STARTRIS bit)
- RX DMA interrupt pending (DMARXRIS bit)
- TX DMA interrupt pending (DMATXRIS bit)
- Programmable trigger value for FIFO has been reached and a TX FIFO request interrupt is pending (TXRIS bit)
- Programmable trigger value for FIFO has been reached and a RX FIFO request interrupt is pending (RXRIS bit)
- Transmit FIFO is empty (TXFERIS bit)
- Receive FIFO is full (RXFFRIS bit)

The I2C master and slave modules have separate interrupt registers. Interrupts can be masked by clearing the appropriate bit in the I2CMIMR or I2CSIMR register. Note that the RIS bit in the Master Raw Interrupt Status (I2CMRIS) register and the DATARIS bit in the Slave Raw Interrupt Status (I2CSRIS) register have multiple interrupt causes, including a next byte transfer request interrupt. This interrupt is generated when the master and slave are requesting a receive or transmit transaction.

46.2.4 Loopback Operation

The I2C modules can be placed into an internal loopback mode for diagnostic or debug work by setting the LPBK bit in the I2C Master Configuration (I2CMCR) register. In loopback mode, the SDA and SCL signals from the master and are connected to the SDA and SCL signals of the slave module to allow internal testing of the device without having to go through I/O.

46.2.5 FIFO and µDMA Operation

Both the master and the slave module have the capability to access two 8-byte FIFOs that can be used in conjunction with the µDMA for fast transfer of data. The transmit (TX) FIFO and receive (RX) FIFO can be independently assigned to either the I2C master or I2C slave. Thus, the following FIFO assignments are allowed:

- The transmit and receive FIFOs can be assigned to the master
- The transmit and receive FIFOs can be assigned to the slave
- The transmit FIFO can be assigned to the master, while the receive FIFO is assigned to the slave and conversely

In most cases, both FIFOs will be assigned to either the master or the slave. The FIFO assignment is configured by programming the TXASGNMT and RXASGNMT bit in the I2C FIFO Control (I2CFIFOCTL) register.

Each FIFO has a programmable threshold point that indicates when the FIFO service interrupt should be generated. Additionally, a FIFO receive full and transmit empty interrupt can be enabled in the Interrupt Mask (I2CxIMR) registers of both the master and slave. Note that if we clear the TXFERIS interrupt (by setting the TXFEIC bit) when the TX FIFO is empty, the TXFERIS interrupt does not reassert even though the TX FIFO remains empty in this situation.

When a FIFO is not assigned to a master or a slave module, the FIFO interrupt and status signals to the module are forced to a state that indicates the FIFO is empty. For example, if the TX FIFO is assigned to the master module, the status signals to the slave transmit interface indicates that the FIFO is empty.

Note

The FIFOs must be empty when reassigning the FIFOs for proper functionality

46.2.5.1 Master Module Burst Mode

A BURST command is provided for the master module that allows a sequence of data transfers using the µDMA (or software, if desired) to handle the data in the FIFO. The BURST command is enabled by setting the BURST bit in the Master Control/Status (I2CMCS) register. The number of bytes transferred by a BURST request is programmed in the I2C Master Burst Length (I2CMBLEN) register and a copy of this value is automatically written to the I2C Master Burst Count (I2CMBCNT) register to be used as a down-counter during the BURST transfer. The bytes written to the I2C FIFO Data (I2CFIFODATA) register are transferred to the RX FIFO or TX FIFO depending on whether a transmit or receive is being executed. If data is NACKed during a BURST and the STOP bit is set in the I2CMCS register, the transfer terminates. If the STOP bit is not set, the software application must issue a repeated STOP or START when a NACK interrupt is asserted. In the case of a NACK, the I2CMBCNT register can be used to determine the amount of data that was transferred prior to the BURST termination. If the address is NACKed during a transfer, then a STOP is issued.

46.2.5.1.1 Master Module µDMA Functionality

When the Master Control/Status (I2CMCS) register is set to enable BURST and the master I^2C µDMA channel is enabled in the DMA Channel Map Select n (DMACHMAPn) registers in the µDMA, the master control module will assert either the internal single µDMA request signal (dma_sreq) or multiple µDMA request signal (dma_req) to the µDMA. Note that there are separate dma_req and dma_sreq signals for transmit and receive. A single µDMA request (dma_sreq) will be asserted by the master module when the Rx FIFO has at least one data byte present in the FIFO and/or when the Tx FIFO has at least one space available to fill. The dma_req (or Burst) signal will be asserted when Rx FIFO fill level is higher than trigger level and/or the Tx FIFO burst length remaining is less than 4 bytes and the FIFO fill level is less than trigger level. If a single transfer or BURST operation has completed, the µDMA sends a dma_done signal to the master module represented by the DMATX and DMARX interrupts in the I2CMIMR, I2CMRIS, I2CMMIS, and I2CMICR registers.

If the µDMA I2C channel is disabled and software is used to handle the BURST command, software can read the FIFO Status (I2CFIFOSTAT) Register and the Master Burst Count (I2CMBC) register to determine whether the FIFO needs servicing during the BURST transaction. A trigger value can be programmed in the I2CFIFOCTL register to allow for interrupts at various fill levels of the FIFOs.

The NACK and ARBLOST bits in the interrupt status registers can be enabled to indicate no acknowledgment of data transfer or an arbitration loss on the bus.

When the master module is transmitting FIFO data, software can fill the Tx FIFO in advance of setting the BURST bit in the I2CMCS register. If the FIFO is empty when the µDMA is enabled for BURST mode, the dma_req and dma_sreq both assert (assuming the I2CMBLEN register is programmed to at least four bytes and the Tx FIFO fill level is less than the trigger set). If the I2CMBLEN register value is less than four and the Tx FIFO is not full but more than trigger level, only dma_sreq asserts. Single requests will be generated as required to keep the FIFO full until the number of bytes specified in the I2CMBLEN register has been transferred to the FIFO (and the I2CMBCOUNT register reaches 0x0). At this point, no further requests are generated until the next BURST command is issued. If the µDMA is disabled, FIFOs will be serviced based on the interrupts active in the master interrupt status registers, the FIFO trigger values shown in the I2CFIFOSTATUS register and completion of a BURST transfer.

When the master module is receiving FIFO data, the Rx FIFO is initially empty and no requests are asserted. If data is read from the slave and placed into the Rx FIFO, the dma_sreq signal to the µDMA is asserted to indicate there is data to be transferred. If the Rx FIFO contains at least 4 bytes, the dma_req signal is also asserted. The µDMA will continue to transfer data out of the Rx FIFO until it has reached the amount of bytes programmed in the I2CMBLEN register.

Note

The TXFEIM interrupt mask bit in the I2CMIMR register should be clear (masking the TXFE interrupt) when the master is performing an RX Burst from the RXFIFO and should be unmasked before starting a TX FIFO transfers.

46.2.5.2 Slave Module

The slave module also has the capability to use the µDMA in Rx and Tx FIFO data transfers. If the Tx FIFO is assigned to the slave module and the TXFIFO bit is set in the I2CSCSR register, the slave module will generate a single µDMA request, dma_sreq, if the master module requests the next byte transfer. If the FIFO fill level is less than the trigger level, a µDMA multiple transfer request, dma_req, will be asserted to continue data transfers from the µDMA.

If the Rx FIFO is assigned to the slave module and the RXFIFO bit is set in the I2CSCSR register, then the slave module will generate a signal µDMA request, dma_sreq, if there is any data to be transferred. The dma_req signal will be asserted when the Rx FIFO has more data than the trigger level programmed by the RXTRIG bit in the I2CFIFOCTL register.

Note

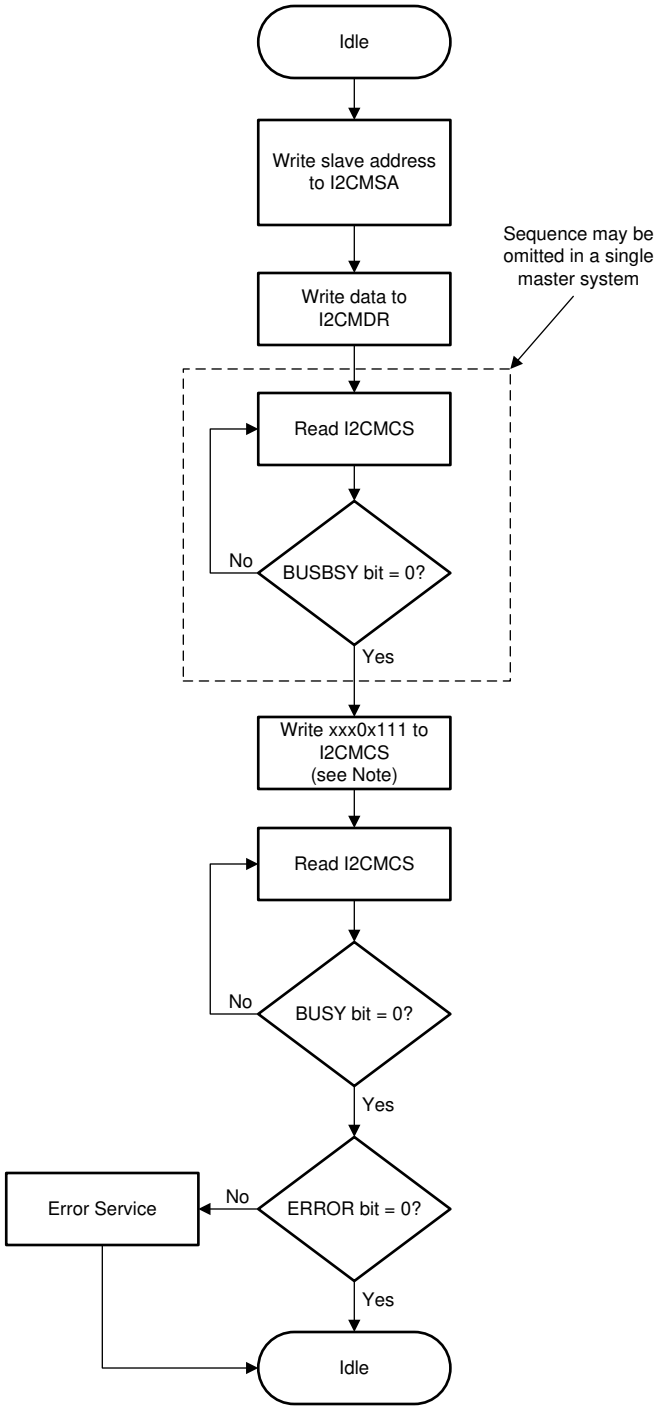
Best practice recommends that an application should not switch between the I2CSDR register and TX FIFO or conversely for successive transactions.

46.2.6 Command Sequence Flow Charts

This section details the steps required to perform the various I2C transfer types in both master and slave mode.

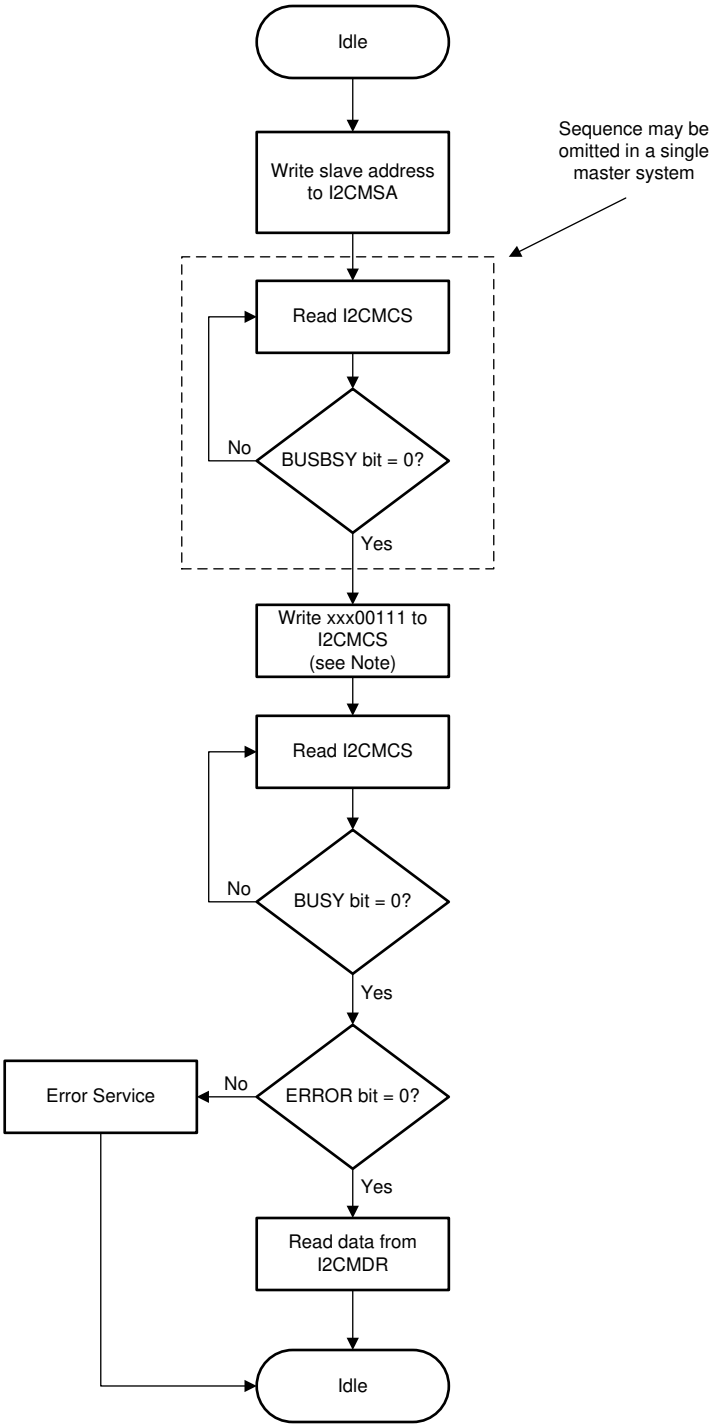
46.2.6.1 I2C Master Command Sequences

The following figures show the command sequences available for the I^{2}C master.



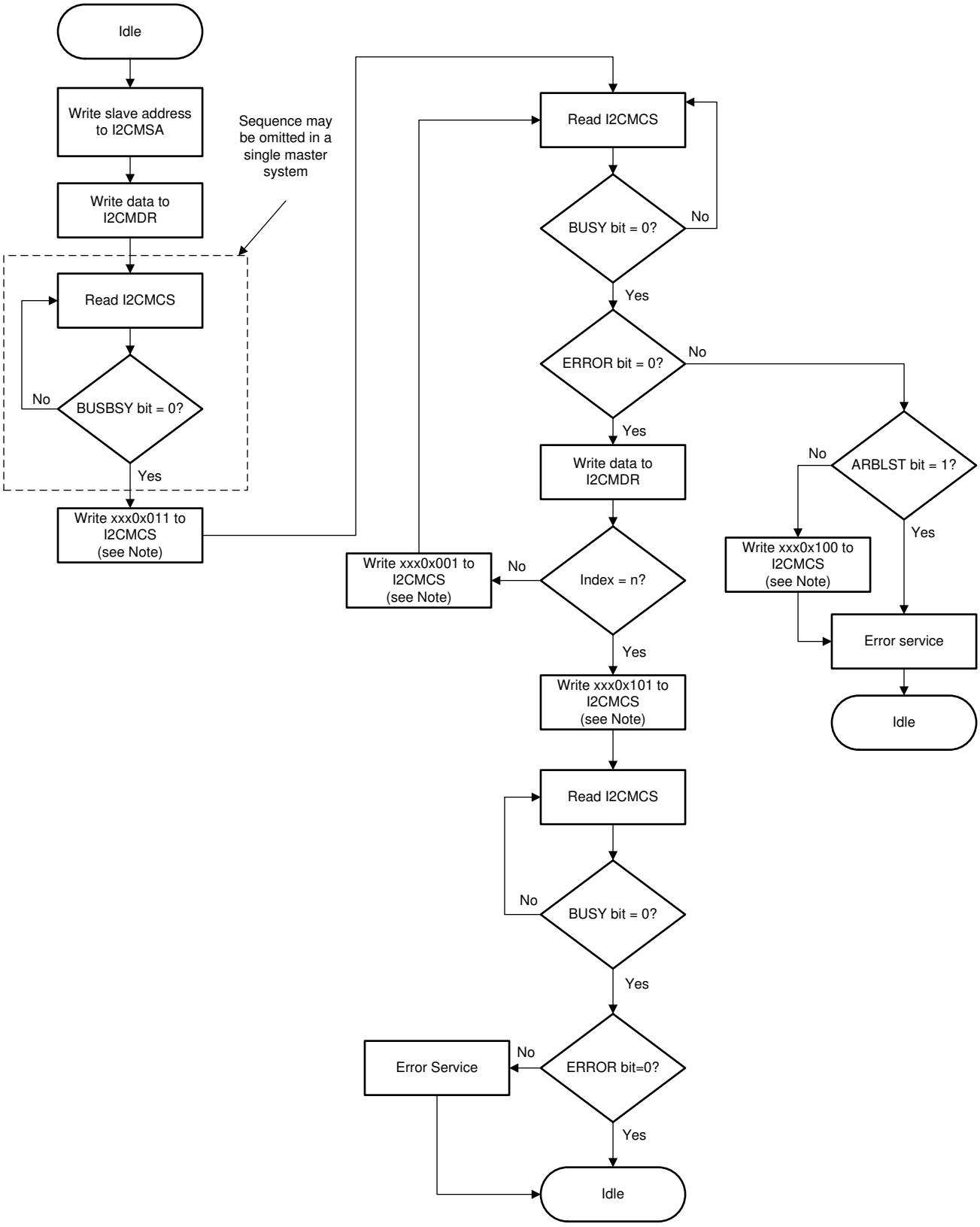
Note: x = application-specific bit

Figure 46-8. Master Single Transmit



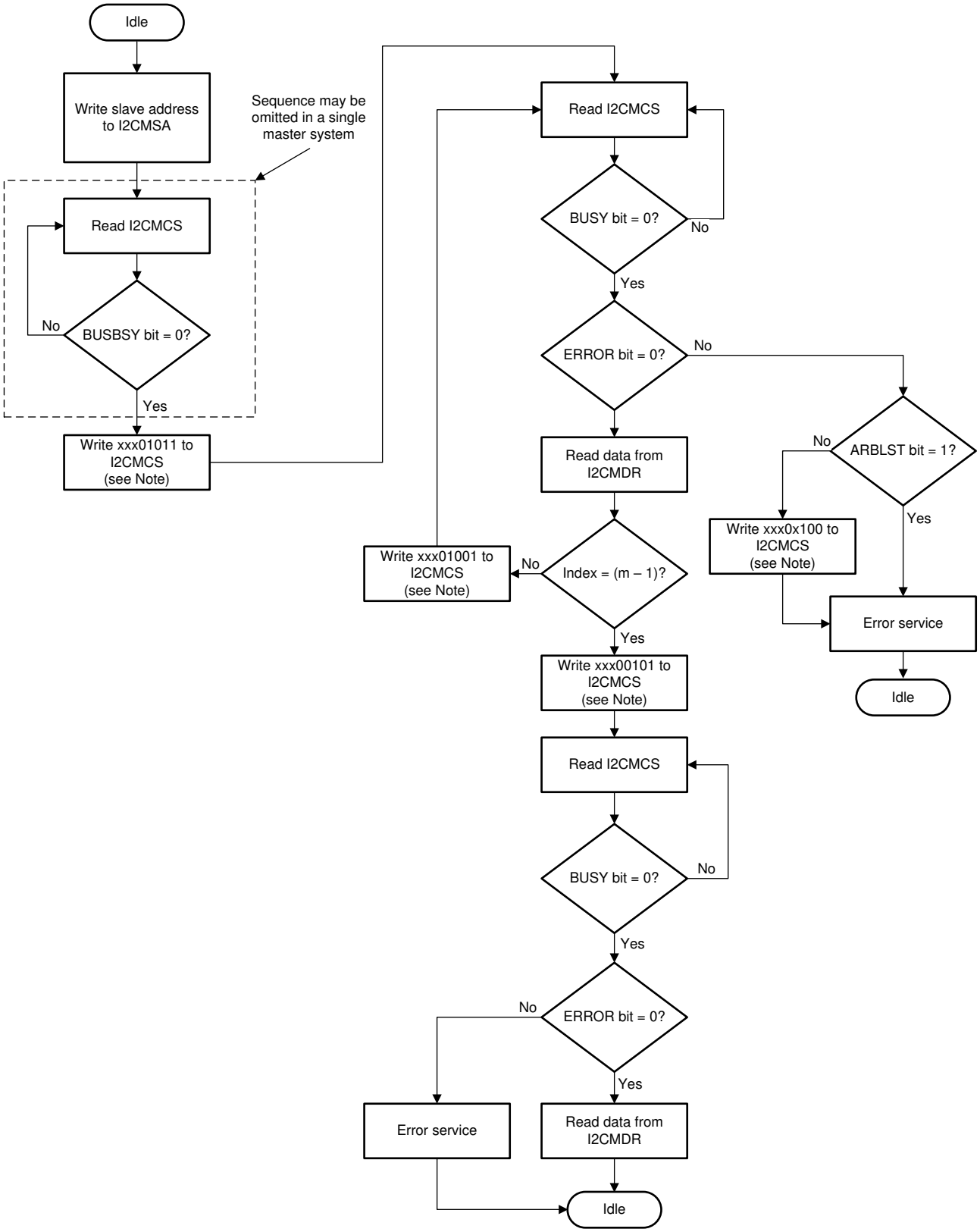
Note: x = application-specific bit

Figure 46-9. Master Single Receive



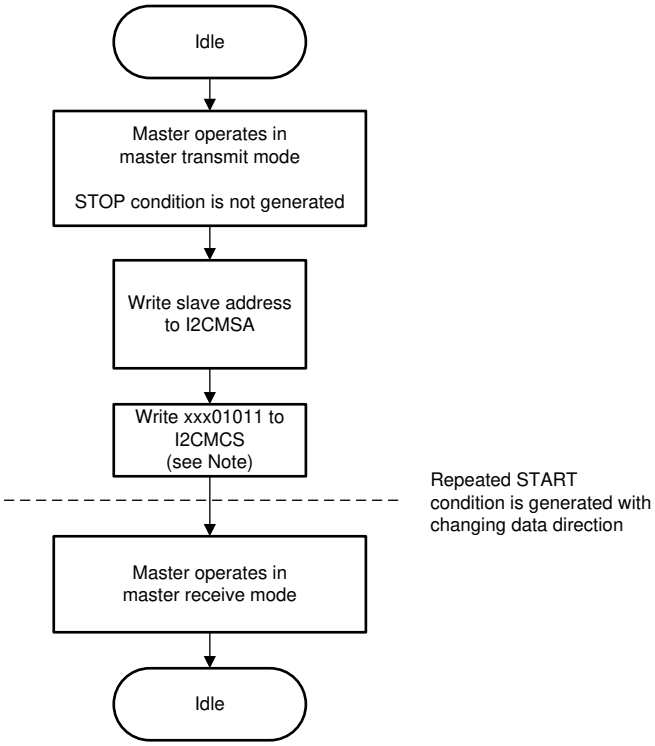
Note: x = application-specific bit

Figure 46-10. Master Transmit of Multiple Data Bytes



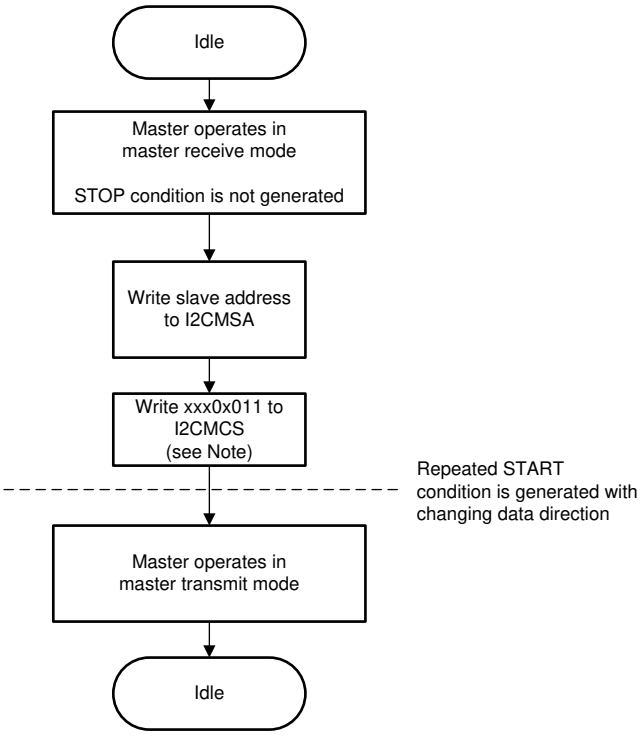
Note: x = application-specific bit

Figure 46-11. Master Receive of Multiple Data Bytes



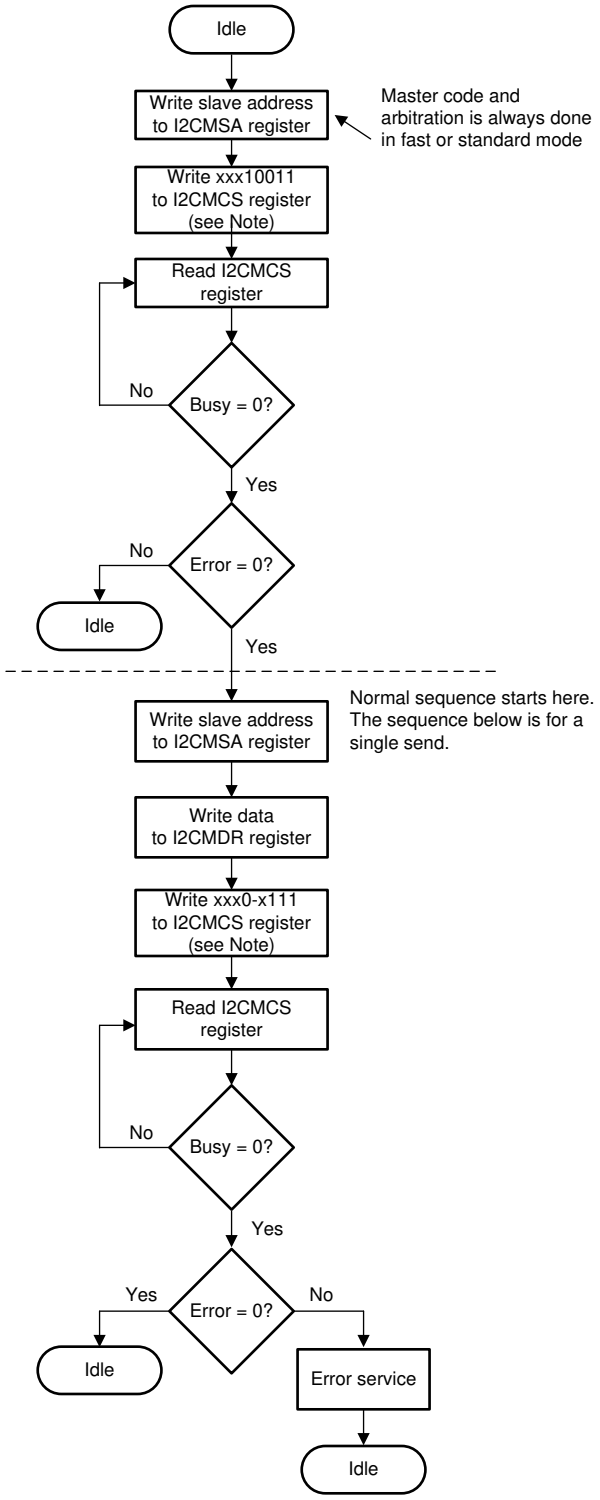
Note: x = application-specific bit

Figure 46-12. Master Receive With Repeated START After Master Transmit



Note: x = application-specific bit

Figure 46-13. Master Transmit With Repeated START After Master Receive

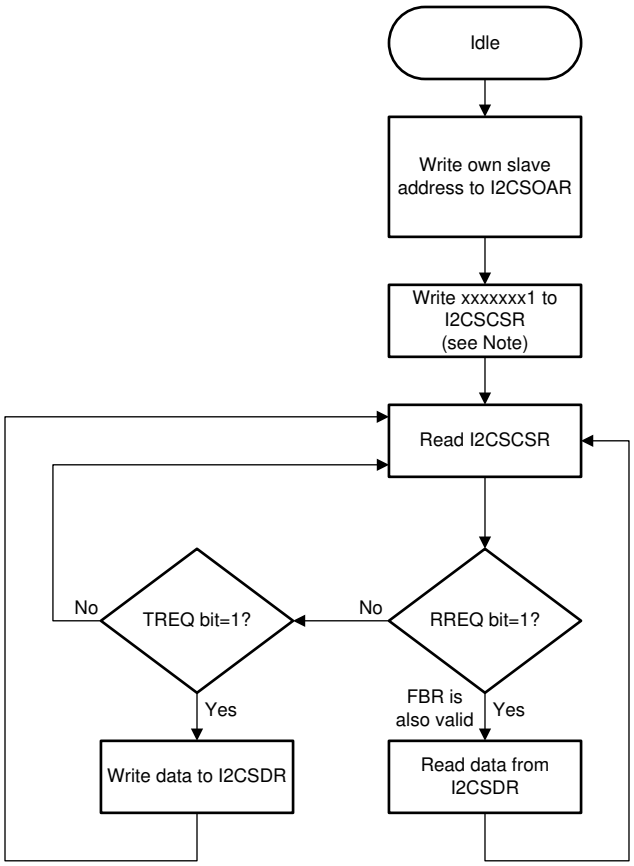


Note: x = application-specific bit

Figure 46-14. Standard High-Speed Mode Master Transmit

46.2.6.2 I2C Slave Command Sequences

Figure 46-15 shows the command sequence available for the I2C slave.



Note: x = application-specific bit

Figure 46-15. Slave Command Sequence

46.3 Initialization and Configuration

46.3.1 Configure the I2C Module to Transmit a Single Byte as a Master

The following example shows how to configure the I2C module to transmit a single byte as a master. This assumes the system clock is 200 MHz.

1. Enable the I2C clock using the CMPCLKCR0 register in the system control module.
2. In the GPIO module, configure GPxCSELy register to allow CM core to control corresponding GPIOs. To determine which GPIOs to configure, see the GPIO Muxed Pins tables in the device data sheet.
3. Configure GPxGMUXy and GPxMUXy register to assign the I^2C signals to the appropriate pins.

Note

The GPIO configuration register GPyODR must be set to normal mode when the CM-I2C is used. The open-drain operation for CM-I2C is managed by the CM-I2C module.

4. Initialize the I^2C master by writing the I2CMCS_WRITE register with a value of 0x0000.0010.
5. Set the desired SCL clock speed of 100 kbps by writing the I2CMTPR register with the correct value. The value written to the I2CMTPR register represents the number of system clock periods in one SCL clock period. The TPR value is calculated by:

   TPR = (System Clock / (2 × (SCL_LP + SCL_HP) × SCL_CLK)) – 1

   TPR = (200 MHz / (2 × (6 + 4) × 100000)) – 1

   TPR = 99 (or) 0x63

   Write the I2CMTPR register with the value of 0x0000.0063.
6. Specify the slave address of the master and that the next operation is a Transmit by writing the I2CMSA register with a value of 0x0000.0076. This sets the slave address to 0x3B.
7. Place data (byte) to be transmitted in the data register by writing the I2CMDR register with the desired data.
8. Initiate a single byte transmit of the data from the master to the slave by writing the I2CMCS register with a value of 0x0000.0007 (STOP, START, RUN).
9. Wait until the transmission completes by polling the BUSBSY bit in the I2CMCS register until it has been cleared.
10. Check the ERROR bit in the I2CMCS register to confirm the transmit was acknowledged.

46.3.2 Configure the I2C Master to High-Speed Mode

To configure the I2C master to high-speed mode:

1. Enable the I2C clock using the CMPCLKCR0 register in the system control module.
2. In the GPIO module, configure GPxCSELy register to allow CM core to control corresponding GPIOs. To determine which GPIOs to configure, see the GPIO Muxed Pins tables in the device data sheet.
3. Configure GPxGMUXy and GPxMUXy register to assign the I2C signals to the appropriate pins.

Note

The GPIO configuration register GPyODR must be set to normal mode when the CM-I2C is used. The open-drain operation for CM-I2C is managed by the CM-I2C module.

4. Initialize the I^2C master by writing the I2CMCR register with a value of 0x0000.0010.
5. Set the desired SCL clock speed of 3.33 Mbps by writing the I2CMTPR register with the correct value. The value written to the I2CMTPR register represents the number of system clock periods in one SCL clock period. The TPR value is determined by:

   TPR = (System Clock / (2 × (SCL_LP + SCL_HP) × SCL_CLK)) – 1

   TPR = (200 MHz / (2 × (2 + 1) × 3330000)) – 1

   TPR = 9

   Write the I2CMTPR register with the value of 0x0000.0009.
6. To send the master code byte, software should place the value of the master code byte into the I2CMCS_WRITE register and write the I2CMCS_WRITE register with the following value depending on the required operation:
   - For standard high-speed mode, write 0x13 to the I2CMCS_WRITE register.
   - For burst high-speed mode, write 0x50 to the I2CMCS_WRITE register.
7. This places the I2C master peripheral in high-speed mode, and all subsequent transfers (until STOP) are carried out at high-speed data rate using the normal I2CMCS_WRITE command bits, without setting the HS bit in the I2CMCS register.
8. The transaction is ended by setting the STOP bit in the I2CMCS register.
9. Wait until the transmission completes by polling the BUSBSY bit in the I2CMCS register until it has been cleared.
10. Check the ERROR bit in the I2CMCS register to confirm the transmit was acknowledged.

46.4 CM I2C Registers

This section describes the Connectivity Manager I2C Registers.

46.4.1 CM I2C Base Addresses

Table 46-3. CM I2C Base Address Table (CM)

DriverLib Name	Base Address	uDMA Access	Ethernet DMA Access
I2C0_BASE	0x4002_0000	YES	-

46.4.2 CM_I2C_REGS Registers

Table 46-4 lists the memory-mapped registers for the CM_I2C_REGS registers. All register offset addresses not listed in Table 46-4 should be considered as reserved locations and the register contents should not be modified.

Table 46-4. CM_I2C_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	I2CMSA	I2C Master Slave Address		Go
4h	I2CMCS	I2C Master Control/Status		Go
8h	I2CMDR	I2C Master Data		Go
Ch	I2CMTPR	I2C Master Timer Period		Go
10h	I2CMIMR	I2C Master Interrupt Mask		Go
14h	I2CMRIS	I2C Master Raw Interrupt Status		Go
18h	I2CMMIS	I2C Master Masked Interrupt Status		Go
1Ch	I2CMICR	I2C Master Interrupt Clear		Go
20h	I2CMCR	I2C Master Configuration		Go
24h	I2CMCLKOCNT	I2C Master Clock Low Timeout Count		Go
2Ch	I2CMBMON	I2C Master Bus Monitor		Go
30h	I2CMBLEN	I2C Master Burst Length		Go
34h	I2CMBCNT	I2C Master Burst Count		Go
800h	I2CSOAR	I2C Slave Own Address		Go
804h	I2CSCSR	I2C Slave Control/Status		Go
808h	I2CSDR	I2C Slave Data		Go
80Ch	I2CSIMR	I2C Slave Interrupt Mask		Go
810h	I2CSRIS	I2C Slave Raw Interrupt Status		Go
814h	I2CSMIS	I2C Slave Masked Interrupt Status		Go
818h	I2CSICR	I2C Slave Interrupt Clear		Go
81Ch	I2CSOAR2	I2C Slave Own Address 2		Go
820h	I2CSACKCTL	I2C Slave ACK Control		Go
F00h	I2CFIFODATARX	I2C FIFO Data RX		Go
F04h	I2CFIFOCTL	I2C FIFO Control		Go
F08h	I2CFIFOSTATUS	I2C FIFO Status		Go
FC0h	I2CPP	I2C Peripheral Properties		Go
FC4h	I2CPC	I2C Peripheral Configuration		Go

Complex bit access types are encoded to fit into small table cells. Table 46-5 shows the codes that are used for access types in this section.

Table 46-5. CM_I2C_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
RC	R C	Read to Clear
Write Type		
W	W	Write
Reset or Default Value		

Table 46-5. CM_I2C_REGS Access Type Codes (continued)

Access Type	Code	Description
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

46.4.2.1 I2CMSA Register (Offset = 0h) [Reset = 0h]

I2CMSA is shown in Figure 46-16 and described in Table 46-6.

Return to the Summary Table.

I2C Master Slave Address

Figure 46-16. I2CMSA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								SA							RS
R-0h								R/W-0h							R/W-0h

Table 46-6. I2CMSA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-1	SA	R/W	0h	I2C Slave Address This field specifies bits A6 through A0 of the slave address. Reset type: PER.RESET
0	RS	R/W	0h	Receive/Send The RS bit specifies if the next master operation is a Receive (High) or Transmit (Low). Value Description 0 Transmit 1 Receive Reset type: PER.RESET

46.4.2.2 I2CMCS Register (Offset = 4h) [Reset = 20h]

I2CMCS is shown in Figure 46-17 and described in Table 46-7.

Return to the Summary Table.

I2C Master Control/Status

Figure 46-17. I2CMCS Register

31	30	29	28	27	26	25	24
ACTDMARX	ACTDMATX	RESERVED					
R-0h	R-0h	R-0h					
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
CLKTO	BUSBSY	IDLE	ARBLST	DATACK	ADRACK	ERROR	BUSY
R-0h	R-0h	R-1h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 46-7. I2CMCS Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ACTDMARX	R	0h	DMA RX Active Status Value Description 0 DMA RX is not active 1 DMA RX is active. Reset type: PER.RESET
30	ACTDMATX	R	0h	DMA TX Active Status Value Description 0 DMA TX is not active 1 DMA TX is active. Reset type: PER.RESET
29-8	RESERVED	R	0h	Reserved
7	CLKTO	R	0h	Clock Timeout Error Value Description 0 No clock timeout error. 1 The clock timeout error has occurred. This bit is cleared when the master sends a STOP condition or if the I2C master is reset. Reset type: PER.RESET
6	BUSBSY	R	0h	Bus Busy Value Description 0 The I2C bus is idle. 1 The I2C bus is busy. The bit changes based on the START and STOP conditions. Reset type: PER.RESET
5	IDLE	R	1h	I2C Idle Value Description 0 The I2C controller is not idle. 1 The I2C controller is idle. Reset type: PER.RESET
4	ARBLST	R	0h	Arbitration Lost Value Description 0 The I2C controller won arbitration. 1 The I2C controller lost arbitration. Reset type: PER.RESET

Table 46-7. I2CMCS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	DATACK	R	0h	Acknowledge Data Value Description 0 The transmitted data was acknowledged 1 The transmitted data was not acknowledged. Reset type: PER.RESET
2	ADRACK	R	0h	Acknowledge Address Value Description 0 The transmitted address was acknowledged 1 The transmitted address was not acknowledged. Reset type: PER.RESET
1	ERROR	R	0h	Error Value Description 0 No error was detected on the last operation. 1 An error occurred on the last operation. The error can be from the slave address not being acknowledged or the transmit data not being acknowledged. Reset type: PER.RESET
0	BUSY	R	0h	I2C Busy Value Description 0 The controller is idle. 1 The controller is busy. When the BUSY bit is set, the other status bits are not valid. Reset type: PER.RESET

46.4.2.3 I2CMDR Register (Offset = 8h) [Reset = 0h]

I2CMDR is shown in Figure 46-18 and described in Table 46-8.

Return to the Summary Table.

I2C Master Data

Figure 46-18. I2CMDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DATA							
R-0h																								R/W-0h							

Table 46-8. I2CMDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	DATA	R/W	0h	This byte contains the data transferred during a transaction. Reset type: PER.RESET

46.4.2.4 I2CMTPR Register (Offset = Ch) [Reset = 00010001h]

I2CMTPR is shown in Figure 46-19 and described in Table 46-9.

Return to the Summary Table.

I2C Master Timer Period

Figure 46-19. I2CMTPR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED													PULSEL		
R-0h													R/W-1h		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								HS	TPR						
R-0h								R-0/ W-0h	R/W-1h						

Table 46-9. I2CMTPR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved
18-16	PULSEL	R/W	1h	Glitch Suppression Pulse Width This field controls the pulse width select for glitch suppression on the SCL and SDA lines. The following values are the glitch suppression values in terms of system clocks. Value Description 0x0 Reserved (Invalid configuration) 0x1 1 clock 0x2 2 clocks 0x3 3 clocks 0x4 4 clocks 0x5 8 clocks 0x6 16 clocks Reset type: PER.RESET
15-8	RESERVED	R	0h	Reserved
7	HS	R-0/W	0h	High-Speed Enable Value Description 0 The SCL Clock Period set by TPR applies to Standard mode (100 Kbps), Fast-mode (400 Kbps), or Fast-mode plus (1 Mbps). 1 The SCL Clock Period set by TPR applies to High-speed mode (3.33 Mbps). Reset type: PER.RESET
6-0	TPR	R/W	1h	Timer Period This field is used in the equation to configure SCL_PERIOD: SCL_PERIOD = 2 X (1 + TPR) X (SCL_LP + SCL_HP) X CLK_PRD where: SCL_PRD is the SCL line period (I2C clock). TPR is the Timer Period register value (range of 1 to 127). SCL_LP is the SCL Low period (fixed at 6). SCL_HP is the SCL High period (fixed at 4). CLK_PRD is the system clock period in ns. Reset type: PER.RESET

46.4.2.5 I2CMIMR Register (Offset = 10h) [Reset = 0h]

I2CMIMR is shown in Figure 46-20 and described in Table 46-10.

Return to the Summary Table.

I2C Master Interrupt Mask

Figure 46-20. I2CMIMR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				RXFFIM	TXFEIM	RXIM	TXIM
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
ARBLOSTIM	STOPIM	STARTIM	NACKIM	DMATXIM	DMARXIM	CLKIM	IM
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 46-10. I2CMIMR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11	RXFFIM	R/W	0h	Receive FIFO Full Interrupt Mask Value Description 0 The RXFFRIS interrupt is suppressed and not sent to the interrupt controller. 1 The Receive FIFO Full interrupt is sent to the interrupt controller when the RXFFRIS bit in the I2CMRIS register is set. Reset type: PER.RESET
10	TXFEIM	R/W	0h	Transmit FIFO Empty Interrupt Mask The TXFEIM interrupt mask bit in the I2CMIMR register should be clear (masking the TXFE interrupt) when the master is performing an RX Burst from the RXFIFO and should be unmasked before starting a TX FIFO transfers. Value Description 0 The TXFERIS interrupt is suppressed and not sent to the interrupt controller. 1 The Transmit FIFO Empty interrupt is sent to the interrupt controller when the TXFERIS bit in the I2CMRIS register is set. Reset type: PER.RESET
9	RXIM	R/W	0h	Receive FIFO Request Interrupt Mask Value Description 0 The RXRIS interrupt is suppressed and not sent to the interrupt controller. 1 The RX FIFO Request interrupt is sent to the interrupt controller when the RXRIS bit in the I2CMRIS register is set. Reset type: PER.RESET
8	TXIM	R/W	0h	Transmit FIFO Request Interrupt Mask Value Description 0 The TXRIS interrupt is suppressed and not sent to the interrupt controller. 1 The TX FIFO Request interrupt is sent to the interrupt controller when the TXRIS bit in the I2CMRIS register is set. Reset type: PER.RESET

Table 46-10. I2CMIMR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	ARBLOSTIM	R/W	0h	Arbitration Lost Interrupt Mask Value Description 0 The ARBLOSTRIS interrupt is suppressed and not sent to the interrupt controller. 1 The Arbitration Lost interrupt is sent to the interrupt controller when the ARBLOSTRIS bit in the I2CMRIS register is set. Reset type: PER.RESET
6	STOPIM	R/W	0h	STOP Detection Interrupt Mask Value Description 0 The STOPRIS interrupt is suppressed and not sent to the interrupt controller. 1 The STOP detection interrupt is sent to the interrupt controller when the STOPRIS bit in the I2CMRIS register is set. Reset type: PER.RESET
5	STARTIM	R/W	0h	START Detection Interrupt Mask Value Description 0 The STARTRIS interrupt is suppressed and not sent to the interrupt controller. 1 The START detection interrupt is sent to the interrupt controller when the STARTRIS bit in the I2CMRIS register is set. Reset type: PER.RESET
4	NACKIM	R/W	0h	Address/Data NACK Interrupt Mask Value Description 0 The NACKRIS interrupt is suppressed and not sent to the interrupt controller. 1 The address/data NACK interrupt is sent to the interrupt controller when the NACKRIS bit in the I2CMRIS register is set. Reset type: PER.RESET
3	DMATXIM	R/W	0h	Transmit DMA Interrupt Mask Value Description 0 The DMATXRIS interrupt is suppressed and not sent to the interrupt controller. 1 The transmit DMA complete interrupt is sent to the interrupt controller when the DMATXRIS bit in the I2CMRIS register is set. Reset type: PER.RESET
2	DMARXIM	R/W	0h	Receive DMA Interrupt Mask Value Description 0 The DMARXRIS interrupt is suppressed and not sent to the interrupt controller. 1 The receive DMA complete interrupt is sent to the interrupt controller when the DMARXRIS bit in the I2CMRIS register is set. Reset type: PER.RESET
1	CLKIM	R/W	0h	Clock Timeout Interrupt Mask Value Description 0 The CLKRIS interrupt is suppressed and not sent to the interrupt controller. 1 The clock timeout interrupt is sent to the interrupt controller when the CLKRIS bit in the I2CMRIS register is set. Reset type: PER.RESET
0	IM	R/W	0h	Master Interrupt Mask Value Description 0 The RIS interrupt is suppressed and not sent to the interrupt controller. 1 The master interrupt is sent to the interrupt controller when the RIS bit in the I2CMRIS register is set. Reset type: PER.RESET

46.4.2.6 I2CMRIS Register (Offset = 14h) [Reset = 0h]

I2CMRIS is shown in Figure 46-21 and described in Table 46-11.

Return to the Summary Table.

I2C Master Raw Interrupt Status

Figure 46-21. I2CMRIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				RXFFRIS	TXFERIS	RXRIS	TXRIS
R-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
ARBLOSTRIS	STOPRIS	STARTRIS	NACKRIS	DMATXRIS	DMARXRIS	CLKRIS	RIS
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 46-11. I2CMRIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11	RXFFRIS	R	0h	Receive FIFO Full Raw Interrupt Status Value Description 0 No interrupt 1 The Receive FIFO Full interrupt is pending. This bit is cleared by writing a 1 to the RXFFIC bit in the I2CMICR register. Reset type: PER.RESET
10	TXFERIS	R	0h	Transmit FIFO Empty Raw Interrupt Status Value Description 0 No interrupt 1 The Transmit FIFO Empty interrupt is pending. This bit is cleared by writing a 1 to the TXFEIC bit in the I2CMICR register. Note that if we clear the TXFERIS interrupt (by setting the TXFEIC bit) when the TX FIFO is empty, the TXFERIS interrupt does not reassert even though the TX FIFO remains empty in this situation. Reset type: PER.RESET
9	RXRIS	R	0h	Receive FIFO Request Raw Interrupt Status Value Description 0 No interrupt 1 The trigger level for the RX FIFO has been reached or there is data in the FIFO and the burst count is zero. Thus, a RX FIFO request interrupt is pending. This bit is cleared by writing a 1 to the RXIC bit in the I2CMICR register. Reset type: PER.RESET

Table 46-11. I2CMRIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	TXRIS	R	0h	Transmit Request Raw Interrupt Status Value Description 0 No interrupt 1 The trigger level for the TX FIFO has been reached and more data is needed to complete the burst. Thus, a TX FIFO request interrupt is pending. This bit is cleared by writing a 1 to the TXIC bit in the I2CMICR register. Reset type: PER.RESET
7	ARBLOSTRIS	R	0h	Arbitration Lost Raw Interrupt Status Value Description 0 No interrupt 1 The Arbitration Lost interrupt is pending. This bit is cleared by writing a 1 to the ARBLOSTIC bit in the I2CMICR register. Reset type: PER.RESET
6	STOPRIS	R	0h	STOP Detection Raw Interrupt Status Value Description 0 No interrupt 1 The STOP Detection interrupt is pending. This bit is cleared by writing a 1 to the STOPIC bit in the I2CMICR register. Reset type: PER.RESET
5	STARTRIS	R	0h	START Detection Raw Interrupt Status Value Description 0 No interrupt 1 The START Detection interrupt is pending. This bit is cleared by writing a 1 to the STARTIC bit in the I2CMICR register. Reset type: PER.RESET
4	NACKRIS	R	0h	Address/Data NACK Raw Interrupt Status Value Description 0 No interrupt 1 The address/data NACK interrupt is pending. This bit is cleared by writing a 1 to the NACKIC bit in the I2CMICR register. Reset type: PER.RESET
3	DMATXRIS	R	0h	Transmit DMA Raw Interrupt Status Value Description 0 No interrupt. 1 The transmit DMA complete interrupt is pending. This bit is cleared by writing a 1 to the DMATXIC bit in the I2CMICR register. Reset type: PER.RESET
2	DMARXRIS	R	0h	Receive DMA Raw Interrupt Status Value Description 0 No interrupt. 1 The receive DMA complete interrupt is pending. This bit is cleared by writing a 1 to the DMARXIC bit in the I2CMICR register. Reset type: PER.RESET
1	CLKRIS	R	0h	Clock Timeout Raw Interrupt Status Value Description 0 No interrupt. 1 The clock timeout interrupt is pending. This bit is cleared by writing a 1 to the CLKIC bit in the I2CMICR register. Reset type: PER.RESET

Table 46-11. I2CMRIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	RIS	R	0h	Master Raw Interrupt Status This interrupt includes: Master transaction completed Next byte transfer request Value Description 0 No interrupt. 1 A master interrupt is pending. This bit is cleared by writing a 1 to the IC bit in the I2CMICR register. Reset type: PER.RESET

46.4.2.7 I2CMMIS Register (Offset = 18h) [Reset = 0h]

I2CMMIS is shown in Figure 46-22 and described in Table 46-12.

Return to the Summary Table.

I2C Master Masked Interrupt Status

Figure 46-22. I2CMMIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				RXFFMIS	TXFEMIS	RXMIS	TXMIS
R-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
ARBLOSTMIS	STOPMIS	STARTMIS	NACKMIS	DMATXMIS	DMARXMIS	CLKMIS	MIS
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 46-12. I2CMMIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11	RXFFMIS	R	0h	Receive FIFO Full Interrupt Mask Value Description 0 No interrupt. 1 An unmasked Receive FIFO Full interrupt was signaled and is pending. This bit is cleared by writing a 1 to the RXFFIC bit in the I2CMICR register. Reset type: PER.RESET
10	TXFEMIS	R	0h	Transmit FIFO Empty Interrupt Mask Value Description 0 No interrupt. 1 An unmasked Transmit FIFO Empty interrupt was signaled and is pending. This bit is cleared by writing a 1 to the TXFEIC bit in the I2CMICR register. Reset type: PER.RESET
9	RXMIS	R	0h	Receive FIFO Request Interrupt Mask Value Description 0 No interrupt. 1 An unmasked Receive FIFO Request interrupt was signaled and is pending. This bit is cleared by writing a 1 to the RXIC bit in the I2CMICR register. Reset type: PER.RESET
8	TXMIS	R	0h	Transmit Request Interrupt Mask Value Description 0 No interrupt. 1 An unmasked Transmit FIFO Request interrupt was signaled and is pending. This bit is cleared by writing a 1 to the TXIC bit in the I2CMICR register. Reset type: PER.RESET

Table 46-12. I2CMMIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	ARBLOSTMIS	R	0h	Arbitration Lost Interrupt Mask Value Description 0 No interrupt. 1 An unmasked Arbitration Lost interrupt was signaled and is pending. This bit is cleared by writing a 1 to the ARBLOSTIC bit in the I2CMICR register. Reset type: PER.RESET
6	STOPMIS	R	0h	STOP Detection Interrupt Mask Value Description 0 No interrupt. 1 An unmasked STOP Detection interrupt was signaled and is pending. This bit is cleared by writing a 1 to the STOPIC bit in the I2CMICR register. Reset type: PER.RESET
5	STARTMIS	R	0h	START Detection Interrupt Mask Value Description 0 No interrupt. 1 An unmasked START Detection interrupt was signaled and is pending. This bit is cleared by writing a 1 to the STARTIC bit in the I2CMICR register. Reset type: PER.RESET
4	NACKMIS	R	0h	Address/Data NACK Interrupt Mask Value Description 0 No interrupt. 1 An unmasked Address/Data NACK interrupt was signaled and is pending. This bit is cleared by writing a 1 to the NACKIC bit in the I2CMICR register. Reset type: PER.RESET
3	DMATXMIS	R	0h	Transmit DMA Interrupt Status Value Description 0 No interrupt. 1 An unmasked transmit DMA complete interrupt was signaled and is pending. This bit is cleared by writing a 1 to the DMATXIC bit in the I2CMICR register. Reset type: PER.RESET
2	DMARXMIS	R	0h	Receive DMA Interrupt Status Value Description 0 No interrupt. 1 An unmasked receive DMA complete interrupt was signaled and is pending. This bit is cleared by writing a 1 to the DMARXIC bit in the I2CMICR register. Reset type: PER.RESET
1	CLKMIS	R	0h	Clock Timeout Masked Interrupt Status Value Description 0 No interrupt. 1 An unmasked clock timeout interrupt was signaled and is pending. This bit is cleared by writing a 1 to the CLKIC bit in the I2CMICR register. Reset type: PER.RESET
0	MIS	R	0h	Masked Interrupt Status Value Description 0 An interrupt has not occurred or is masked. 1 An unmasked master interrupt was signaled and is pending. This bit is cleared by writing a 1 to the IC bit in the I2CMICR register. Reset type: PER.RESET

46.4.2.8 I2CMICR Register (Offset = 1Ch) [Reset = 0h]

I2CMICR is shown in Figure 46-23 and described in Table 46-13.

Return to the Summary Table.

I2C Master Interrupt Clear

Figure 46-23. I2CMICR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				RXFFIC	TXFEIC	RXIC	TXIC
R-0h				R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
7	6	5	4	3	2	1	0
ARBLOSTIC	STOPIC	STARTIC	NACKIC	DMATXIC	DMARXIC	CLKIC	IC
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h

Table 46-13. I2CMICR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11	RXFFIC	R-0/W	0h	Receive FIFO Full Interrupt Clear Writing a 1 to this bit clears the RXFFIS bit in the I2CMRIS register and the RXFFMIS bit in the I2CMMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET
10	TXFEIC	R-0/W	0h	Transmit FIFO Empty Interrupt Clear Writing a 1 to this bit clears the TXFERIS bit in the I2CMRIS register and the TXFEMIS bit in the I2CMMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET
9	RXIC	R-0/W	0h	Receive FIFO Request Interrupt Clear Writing a 1 to this bit clears the RXRIS bit in the I2CMRIS register and the RXMIS bit in the I2CMMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET
8	TXIC	R-0/W	0h	Transmit FIFO Request Interrupt Clear Writing a 1 to this bit clears the TXRIS bit in the I2CMRIS register and the TXMIS bit in the I2CMMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET
7	ARBLOSTIC	R-0/W	0h	Arbitration Lost Interrupt Clear Writing a 1 to this bit clears the ARBLOSTRIS bit in the I2CMRIS register and the ARBLOSTMIS bit in the I2CMMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET
6	STOPIC	R-0/W	0h	STOP Detection Interrupt Clear Writing a 1 to this bit clears the STOPRIS bit in the I2CMRIS register and the STOPMIS bit in the I2CMMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET

Table 46-13. I2CMICR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	STARTIC	R-0/W	0h	START Detection Interrupt Clear Writing a 1 to this bit clears the STARTRIS bit in the I2CMRIS register and the STARTMIS bit in the I2CMMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET
4	NACKIC	R-0/W	0h	Address/Data NACK Interrupt Clear Writing a 1 to this bit clears the NACKRIS bit in the I2CMRIS register and the NACKMIS bit in the I2CMMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET
3	DMATXIC	R-0/W	0h	Transmit DMA Interrupt Clear Writing a 1 to this bit clears the DMATXRIS bit in the I2CMRIS register and the DMATXMIS bit in the I2CMMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET
2	DMARXIC	R-0/W	0h	Receive DMA Interrupt Clear Writing a 1 to this bit clears the DMARXRIS bit in the I2CMRIS register and the DMARXMIS bit in the I2CMMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET
1	CLKIC	R-0/W	0h	Clock Timeout Interrupt Clear Writing a 1 to this bit clears the CLKRIS bit in the I2CMRIS register and the CLKMIS bit in the I2CMMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET
0	IC	R-0/W	0h	Master Interrupt Clear Writing a 1 to this bit clears the RIS bit in the I2CMRIS register and the MIS bit in the I2CMMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET

46.4.2.9 I2CMCR Register (Offset = 20h) [Reset = 0h]

I2CMCR is shown in Figure 46-24 and described in Table 46-14.

Return to the Summary Table.

I2C Master Configuration

Figure 46-24. I2CMCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		SFE	MFE	RESERVED			LPBK
R-0h		R/W-0h	R/W-0h	R-0h			R/W-0h

Table 46-14. I2CMCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	SFE	R/W	0h	I2C Slave Function Enable Value Description 0 Slave mode is disabled. 1 Slave mode is enabled. Reset type: PER.RESET
4	MFE	R/W	0h	I2C Master Function Enable Value Description 0 Master mode is disabled. 1 Master mode is enabled. Reset type: PER.RESET
3-1	RESERVED	R	0h	Reserved
0	LPBK	R/W	0h	I2C Loopback Value Description 0 Normal operation. 1 The controller in a test mode loopback configuration. Reset type: PER.RESET

46.4.2.10 I2CMCLKOCNT Register (Offset = 24h) [Reset = 0h]

I2CMCLKOCNT is shown in Figure 46-25 and described in Table 46-15.

Return to the Summary Table.

I2C Master Clock Low Timeout Count

Figure 46-25. I2CMCLKOCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CNTL							
R-0h																								R/W-0h							

Table 46-15. I2CMCLKOCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	CNTL	R/W	0h	I2C Master Count This field contains the upper 8 bits of a 12-bit counter for the clock low timeout count. The value of CNTL must be greater than 0x1. Reset type: PER.RESET

46.4.2.11 I2CMBMON Register (Offset = 2Ch) [Reset = 3h]

I2CMBMON is shown in Figure 46-26 and described in Table 46-16.

Return to the Summary Table.

I2C Master Bus Monitor

Figure 46-26. I2CMBMON Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														SDA	SCL
R-0h														R-1h	R-1h

Table 46-16. I2CMBMON Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	SDA	R	1h	I2C SDA Status Value Description 0 The I2CSDA signal is low. 1 The I2CSDA signal is high. Reset type: PER.RESET
0	SCL	R	1h	I2C SCL Status Value Description 0 The I2CSCL signal is low. 1 The I2CSCL signal is high. Reset type: PER.RESET

46.4.2.12 I2CMBLEN Register (Offset = 30h) [Reset = 0h]

I2CMBLEN is shown in Figure 46-27 and described in Table 46-17.

Return to the Summary Table.

I2C Master Burst Length

Figure 46-27. I2CMBLEN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CNTL							
R-0h																								R/W-0h							

Table 46-17. I2CMBLEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	CNTL	R/W	0h	I2C Burst Length This field contains the programmed length of bytes of the Burst Transaction. If BURST is enabled this register must be set to a non-zero value otherwise an error will occur. Reset type: PER.RESET

46.4.2.13 I2CMBCNT Register (Offset = 34h) [Reset = 0h]

I2CMBCNT is shown in Figure 46-28 and described in Table 46-18.

Return to the Summary Table.

I2C Master Burst Count

Figure 46-28. I2CMBCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CNTL							
R-0h																								R-0h							

Table 46-18. I2CMBCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	CNTL	R	0h	I2C Master Burst Count This field contains the current count-down value of the BURST transaction. Reset type: PER.RESET

46.4.2.14 I2CSOAR Register (Offset = 800h) [Reset = 0h]

I2CSOAR is shown in Figure 46-29 and described in Table 46-19.

Return to the Summary Table.

I2C Slave Own Address

Figure 46-29. I2CSOAR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																									OAR						
R-0h																									R/W-0h						

Table 46-19. I2CSOAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6-0	OAR	R/W	0h	I2C Slave Own Address This field specifies bits A6 through A0 of the slave address. Reset type: PER.RESET

46.4.2.15 I2CSCSR Register (Offset = 804h) [Reset = 0h]

I2CSCSR is shown in Figure 46-30 and described in Table 46-20.

Return to the Summary Table.

I2C Slave Control/Status

Figure 46-30. I2CSCSR Register

31	30	29	28	27	26	25	24
ACTDMARX	ACTDMATX	RESERVED					
R-0h	R-0h	R-0h					
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		QCMDRW	QCMDST	OAR2SEL	FBR	TREQ	RREQ
R-0h		RC-0h	RC-0h	R-0h	R-0h	R-0h	R-0h

Table 46-20. I2CSCSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ACTDMARX	R	0h	DMA RX Active Status Value Description 0 DMA RX is not active 1 DMA RX is active. Reset type: PER.RESET
30	ACTDMATX	R	0h	DMA TX Active Status Value Description 0 DMA TX is not active 1 DMA TX is active. Reset type: PER.RESET
29-6	RESERVED	R	0h	Reserved
5	QCMDRW	RC	0h	Quick Command Read / Write Value Description 0 Quick command was a write 1 Quick command was a read This bit only has meaning when the QCMDST bit is set. Reset type: PER.RESET
4	QCMDST	RC	0h	Quick Command Status Value Description 0 The last transaction was a normal transaction or a transaction has not occurred. 1 The last transaction was a Quick Command transaction. Reset type: PER.RESET
3	OAR2SEL	R	0h	OAR2 Address Matched Value Description 0 Either the address is not matched or the match is in legacy mode. 1 OAR2 address matched and ACKed by the slave. This bit gets reevaluated after every address comparison. Reset type: PER.RESET

Table 46-20. I2CSCSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	FBR	R	0h	First Byte Received Value Description 0 The first byte has not been received. 1 The first byte following the slave's own address has been received. This bit is only valid when the RREQ bit is set and is automatically cleared when data has been read from the I2CSDR register. This bit is not used for slave transmit operations. Reset type: PER.RESET
1	TREQ	R	0h	Transmit Request Value Description 0 No outstanding transmit request. 1 The I2C controller has been addressed as a slave transmitter and is using clock stretching to delay the master until data has been written to the I2CSDR register. Reset type: PER.RESET
0	RREQ	R	0h	Receive Request Value Description 0 No outstanding receive data. 1 The I2C controller has outstanding receive data from the I2C master and is using clock stretching to delay the master until the data has been read from the I2CSDR register. Reset type: PER.RESET

46.4.2.16 I2CSDR Register (Offset = 808h) [Reset = 0h]

I2CSDR is shown in Figure 46-31 and described in Table 46-21.

Return to the Summary Table.

I2C Slave Data

Figure 46-31. I2CSDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DATA							
R-0h																								R/W-0h							

Table 46-21. I2CSDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	DATA	R/W	0h	Data for Transfer This field contains the data for transfer during a slave receive or transmit operation. Reset type: PER.RESET

46.4.2.17 I2CSIMR Register (Offset = 80Ch) [Reset = 0h]

I2CSIMR is shown in Figure 46-32 and described in Table 46-22.

Return to the Summary Table.

I2C Slave Interrupt Mask

Figure 46-32. I2CSIMR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							RXFFIM
R-0h							R/W-0h
7	6	5	4	3	2	1	0
TXFEIM	RXIM	TXIM	DMATXIM	DMARXIM	STOPIM	STARTIM	DATAIM
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 46-22. I2CSIMR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	RXFFIM	R/W	0h	Receive FIFO Full Interrupt Mask Value Description 0 The RXFFRIS interrupt is suppressed and not sent to the interrupt controller. 1 The Receive FIFO Full interrupt is sent to the interrupt controller when the RXFFRIS bit in the I2CSRIS register is set. Reset type: PER.RESET
7	TXFEIM	R/W	0h	Transmit FIFO Empty Interrupt Mask Value Description 0 The TXFERIS interrupt is suppressed and not sent to the interrupt controller. 1 The Transmit FIFO Empty interrupt is sent to the interrupt controller when the TXFERIS bit in the I2CSRIS register is set. Reset type: PER.RESET
6	RXIM	R/W	0h	Receive FIFO Request Interrupt Mask Value Description 0 The RXRIS interrupt is suppressed and not sent to the interrupt controller. 1 The RX FIFO Request interrupt is sent to the interrupt controller when the RXRIS bit in the I2CSRIS register is set. Reset type: PER.RESET
5	TXIM	R/W	0h	Transmit FIFO Request Interrupt Mask Value Description 0 The TXRIS interrupt is suppressed and not sent to the interrupt controller. 1 The TX FIFO Request interrupt is sent to the interrupt controller when the TXRIS bit in the I2CSRIS register is set. Reset type: PER.RESET

Table 46-22. I2CSIMR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	DMATXIM	R/W	0h	Transmit DMA Interrupt Mask Value Description 0 The DMATXRIS interrupt is suppressed and not sent to the interrupt controller. 1 The transmit DMA complete interrupt is sent to the interrupt controller when the DMATXRIS bit in the I2CSRIS register is set. Reset type: PER.RESET
3	DMARXIM	R/W	0h	Receive DMA Interrupt Mask Value Description 0 The DMARXRIS interrupt is suppressed and not sent to the interrupt controller. 1 The receive DMA complete interrupt is sent to the interrupt controller when the DMARXRIS bit in the I2CSRIS register is set. Reset type: PER.RESET
2	STOPIM	R/W	0h	Stop Condition Interrupt Mask Value Description 0 The STOPRIS interrupt is suppressed and not sent to the interrupt controller. 1 The STOP condition interrupt is sent to the interrupt controller when the STOPRIS bit in the I2CSRIS register is set. Reset type: PER.RESET
1	STARTIM	R/W	0h	Start Condition Interrupt Mask Value Description 0 The STARTRIS interrupt is suppressed and not sent to the interrupt controller. 1 The START condition interrupt is sent to the interrupt controller when the STARTRIS bit in the I2CSRIS register is set. Reset type: PER.RESET
0	DATAIM	R/W	0h	Data Interrupt Mask Value Description 0 The DATARIS interrupt is suppressed and not sent to the interrupt controller. 1 Data interrupt sent to interrupt controller when DATARIS bit in the I2CSRIS register is set. Reset type: PER.RESET

46.4.2.18 I2CSRIS Register (Offset = 810h) [Reset = 0h]

I2CSRIS is shown in Figure 46-33 and described in Table 46-23.

Return to the Summary Table.

I2C Slave Raw Interrupt Status

Figure 46-33. I2CSRIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							RXFFRIS
R-0h							R-0h
7	6	5	4	3	2	1	0
TXFERIS	RXRIS	TXRIS	DMATXRIS	DMARXRIS	STOPRIS	STARTRIS	DATARIS
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 46-23. I2CSRIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	RXFFRIS	R	0h	Receive FIFO Full Raw Interrupt Status Value Description 0 No interrupt 1 The Receive FIFO Full interrupt is pending. This bit is cleared by writing a 1 to the RXFFIC bit in the I2CSICR register. Reset type: PER.RESET
7	TXFERIS	R	0h	Transmit FIFO Empty Raw Interrupt Status Value Description 0 No interrupt 1 The Transmit FIFO Empty interrupt is pending. This bit is cleared by writing a 1 to the TXFEIC bit in the I2CSICR register. Note that if the TXFERIS interrupt is cleared (by setting the TXFEIC bit) when the TX FIFO is empty, the TXFERIS interrupt does not reassert even though the TX FIFO remains empty in this situation. Reset type: PER.RESET
6	RXRIS	R	0h	Receive FIFO Request Raw Interrupt Status Value Description 0 No interrupt 1 The trigger value for the FIFO has been reached and a RX FIFO Request interrupt is pending. This bit is cleared by writing a 1 to the RXIC bit in the I2CSICR register. Reset type: PER.RESET
5	TXRIS	R	0h	Transmit Request Raw Interrupt Status Value Description 0 No interrupt 1 The trigger value for the FIFO has been reached and a TX FIFO Request interrupt is pending. This bit is cleared by writing a 1 to the TXIC bit in the I2CSICR register. Reset type: PER.RESET

Table 46-23. I2CSRIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	DMATXRIS	R	0h	Transmit DMA Raw Interrupt Status Value Description 0 No interrupt. 1 A transmit DMA complete interrupt is pending. This bit is cleared by writing a 1 to the DMATXIC bit in the I2CSICR register. Reset type: PER.RESET
3	DMARXRIS	R	0h	Receive DMA Raw Interrupt Status Value Description 0 No interrupt. 1 A receive DMA complete interrupt is pending. This bit is cleared by writing a 1 to the DMARXIC bit in the I2CSICR register. Reset type: PER.RESET
2	STOPRIS	R	0h	Stop Condition Raw Interrupt Status Value Description 0 No interrupt. 1 A STOP condition interrupt is pending. This bit is cleared by writing a 1 to the STOPIC bit in the I2CSICR register. Reset type: PER.RESET
1	STARTRIS	R	0h	Start Condition Raw Interrupt Status Value Description 0 No interrupt. 1 A START condition interrupt is pending. This bit is cleared by writing a 1 to the STARTIC bit in the I2CSICR register. Reset type: PER.RESET
0	DATARIS	R	0h	Data Raw Interrupt Status This interrupt encompasses the following: Slave transaction received Slave transaction requested Next byte transfer request Value Description 0 No interrupt. 1 Slave Interrupt is pending. This bit is cleared by writing a 1 to the DATAIC bit in the I2CSICR register. Reset type: PER.RESET

46.4.2.19 I2CSMIS Register (Offset = 814h) [Reset = 0h]

I2CSMIS is shown in Figure 46-34 and described in Table 46-24.

Return to the Summary Table.

I2C Slave Masked Interrupt Status

Figure 46-34. I2CSMIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							RXFFMIS
R-0h							R-0h
7	6	5	4	3	2	1	0
TXFEMIS	RXMIS	TXMIS	DMATXMIS	DMARXMIS	STOPMIS	STARTMIS	DATAMIS
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 46-24. I2CSMIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	RXFFMIS	R	0h	Receive FIFO Full Interrupt Mask Value Description 0 No interrupt. 1 An unmasked Receive FIFO Full interrupt was signaled and is pending. This bit is cleared by writing a 1 to the RXFFIC bit in the I2CSICR register. Reset type: PER.RESET
7	TXFEMIS	R	0h	Transmit FIFO Empty Interrupt Mask Value Description 0 No interrupt. 1 An unmasked Transmit FIFO Empty interrupt was signaled and is pending. This bit is cleared by writing a 1 to the TXFEIC bit in the I2CSICR register. Reset type: PER.RESET
6	RXMIS	R	0h	Receive FIFO Request Interrupt Mask Value Description 0 No interrupt. 1 An unmasked Receive FIFO Request interrupt was signaled and is pending. This bit is cleared by writing a 1 to the RXIC bit in the I2CSICR register. Reset type: PER.RESET
5	TXMIS	R	0h	Transmit FIFO Request Interrupt Mask Value Description 0 No interrupt. 1 An unmasked Transmit FIFO Request interrupt was signaled and is pending. This bit is cleared by writing a 1 to the TXIC bit in the I2CSICR register. Reset type: PER.RESET

Table 46-24. I2CSMIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	DMATXMIS	R	0h	Transmit DMA Masked Interrupt Status Value Description 0 An interrupt has not occurred or is masked. 1 An unmasked transmit DMA complete interrupt was signaled is pending. This bit is cleared by writing a 1 to the DMATXIC bit in the I2CSICR register. Reset type: PER.RESET
3	DMARXMIS	R	0h	Receive DMA Masked Interrupt Status Value Description 0 An interrupt has not occurred or is masked. 1 An unmasked receive DMA complete interrupt was signaled is pending. This bit is cleared by writing a 1 to the DMARXIC bit in the I2CSICR register. Reset type: PER.RESET
2	STOPMIS	R	0h	Stop Condition Masked Interrupt Status Value Description 0 An interrupt has not occurred or is masked. 1 An unmasked STOP condition interrupt was signaled is pending. This bit is cleared by writing a 1 to the STOPIC bit in the I2CSICR register. Reset type: PER.RESET
1	STARTMIS	R	0h	Start Condition Masked Interrupt Status Value Description 0 An interrupt has not occurred or is masked. 1 An unmasked START condition interrupt was signaled is pending. This bit is cleared by writing a 1 to the STARTIC bit in the I2CSICR register. Reset type: PER.RESET
0	DATAMIS	R	0h	Data Masked Interrupt Status Value Description 0 An interrupt has not occurred or is masked. 1 An unmasked slave data interrupt was signaled is pending. This bit is cleared by writing a 1 to the DATAIC bit in the I2CSICR register. Reset type: PER.RESET

46.4.2.20 I2CSICR Register (Offset = 818h) [Reset = 0h]

I2CSICR is shown in Figure 46-35 and described in Table 46-25.

Return to the Summary Table.

I2C Slave Interrupt Clear

Figure 46-35. I2CSICR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							RXFFIC
R-0h							R-0/W-0h
7	6	5	4	3	2	1	0
TXFEIC	RXIC	TXIC	DMATXIC	DMARXIC	STOPIC	STARTIC	DATAIC
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h

Table 46-25. I2CSICR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	RXFFIC	R-0/W	0h	Receive FIFO Full Interrupt Mask Writing a 1 to this bit clears the RXFFIS bit in the I2CSRIS register and the RXFFMIS bit in the I2CSMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET
7	TXFEIC	R-0/W	0h	Transmit FIFO Empty Interrupt Mask Writing a 1 to this bit clears the TXFERIS bit in the I2CSRIS register and the TXFEMIS bit in the I2CSMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET
6	RXIC	R-0/W	0h	Receive Request Interrupt Mask Writing a 1 to this bit clears the RXRIS bit in the I2CSRIS register and the RXMIS bit in the I2CSMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET
5	TXIC	R-0/W	0h	Transmit Request Interrupt Mask Writing a 1 to this bit clears the TXRIS bit in the I2CSRIS register and the TXMIS bit in the I2CSMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET
4	DMATXIC	R-0/W	0h	Transmit DMA Interrupt Clear Writing a 1 to this bit clears the DMATXRIS bit in the I2CSRIS register and the DMATXMIS bit in the I2CSMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET
3	DMARXIC	R-0/W	0h	Receive DMA Interrupt Clear Writing a 1 to this bit clears the DMARXRIS bit in the I2CSRIS register and the DMARXMIS bit in the I2CSMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET

Table 46-25. I2CSICR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	STOPIC	R-0/W	0h	Stop Condition Interrupt Clear Writing a 1 to this bit clears the STOPRIS bit in the I2CSRIS register and the STOPMIS bit in the I2CSMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET
1	STARTIC	R-0/W	0h	Start Condition Interrupt Clear Writing a 1 to this bit clears the STARTRIS bit in the I2CSRIS register and the STARTMIS bit in the I2CSMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET
0	DATAIC	R-0/W	0h	Data Interrupt Clear Writing a 1 to this bit clears the DATARIS bit in the I2CSRIS register and the DATMIS bit in the I2CSMIS register. A read of this register returns no meaningful data. Reset type: PER.RESET

46.4.2.21 I2CSOAR2 Register (Offset = 81Ch) [Reset = 0h]

I2CSOAR2 is shown in Figure 46-36 and described in Table 46-26.

Return to the Summary Table.

I2C Slave Own Address 2

Figure 46-36. I2CSOAR2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
OAR2EN	OAR2						
R/W-0h	R/W-0h						

Table 46-26. I2CSOAR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	OAR2EN	R/W	0h	I2C Slave Own Address 2 Enable Value Description 0 The alternate address is disabled. 1 Enables the use of the alternate address in the OAR2 field. Reset type: PER.RESET
6-0	OAR2	R/W	0h	I2C Slave Own Address 2 This field specifies the alternate OAR2 address. Reset type: PER.RESET

46.4.2.22 I2CSACKCTL Register (Offset = 820h) [Reset = 0h]

I2CSACKCTL is shown in Figure 46-37 and described in Table 46-27.

Return to the Summary Table.

I2C Slave ACK Control

Figure 46-37. I2CSACKCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ACKOVAL	ACKOEN
R-0h						R/W-0h	R/W-0h

Table 46-27. I2CSACKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	ACKOVAL	R/W	0h	I2C Slave ACK Override Value Value Description 0 An ACK is sent indicating valid data or command. 1 A NACK is sent indicating invalid data or command. Reset type: PER.RESET
0	ACKOEN	R/W	0h	I2C Slave ACK Override Enable Value Description 0 A response in not provided. 1 An ACK or NACK is sent according to the value written to the ACKOVAL bit. Reset type: PER.RESET

46.4.2.23 I2CFIFODATARX Register (Offset = F00h) [Reset = 0h]

I2CFIFODATARX is shown in Figure 46-38 and described in Table 46-28.

Return to the Summary Table.

I2C FIFO Data RX

Figure 46-38. I2CFIFODATARX Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DATA							
R-0h																								R-0h							

Table 46-28. I2CFIFODATARX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	DATA	R	0h	I2C RX FIFO Read Data Byte This field contains the current byte being read in the RX FIFO stack. Reset type: PER.RESET

46.4.2.24 I2CFIFOCTL Register (Offset = F04h) [Reset = 00040004h]

I2CFIFOCTL is shown in Figure 46-39 and described in Table 46-29.

Return to the Summary Table.

I2C FIFO Control

Figure 46-39. I2CFIFOCTL Register

31	30	29	28	27	26	25	24
RXASGNMT	RXFLUSH	DMARXENA	RESERVED				
R/W-0h	R/W-0h	R/W-0h	R-0h				
23	22	21	20	19	18	17	16
RESERVED					RXTRIG		
R-0h					R/W-4h		
15	14	13	12	11	10	9	8
TXASGNMT	TXFLUSH	DMATXENA	RESERVED				
R/W-0h	R/W-0h	R/W-0h	R-0h				
7	6	5	4	3	2	1	0
RESERVED					TXTRIG		
R-0h					R/W-4h		

Table 46-29. I2CFIFOCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RXASGNMT	R/W	0h	RX Control Assignment Value Description 0 RX FIFO is assigned to Master 1 RX FIFO is assigned to Slave Reset type: PER.RESET
30	RXFLUSH	R/W	0h	RX FIFO Flush Setting this bit will Flush the RX FIFO. This bit will self-clear when the flush has completed. Reset type: PER.RESET
29	DMARXENA	R/W	0h	DMA RX Channel Enable Value Description 0 DMA RX channel disabled 1 DMA RX channel enabled Reset type: PER.RESET
28-19	RESERVED	R	0h	Reserved
18-16	RXTRIG	R/W	4h	RX FIFO Trigger Indicates at what fill level the RX FIFO will generate a trigger. Programming RXTRIG to 0x0 has no effect since no data is present to transfer out of RX FIFO. Value Description 0x0 Trigger when RX FIFO contains no bytes 0x1 Trigger when Rx FIFO contains 1 or more bytes 0x2 Trigger when Rx FIFO contains 2 or more bytes 0x3 Trigger when Rx FIFO contains 3 or more bytes 0x4 Trigger when Rx FIFO contains 4 or more bytes 0x5 Trigger when Rx FIFO contains 5 or more bytes 0x6 Trigger when Rx FIFO contains 6 or more bytes 0x7 Trigger when Rx FIFO contains 7 or more bytes. Reset type: PER.RESET
15	TXASGNMT	R/W	0h	TX Control Assignment Value Description 0 TX FIFO is assigned to Master 1 TX FIFO is assigned to Slave Reset type: PER.RESET

Table 46-29. I2CFIFOCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14	TXFLUSH	R/W	0h	TX FIFO Flush Setting this bit will Flush the TX FIFO. This bit will self-clear when the flush has completed. Reset type: PER.RESET
13	DMATXENA	R/W	0h	DMA TX Channel Enable Value Description 0 DMA TX channel disabled 1 DMA TX channel enabled Reset type: PER.RESET
12-3	RESERVED	R	0h	Reserved
2-0	TXTRIG	R/W	4h	TX FIFO Trigger Indicates at what fill level in the TX FIFO a trigger will be generated. Value Description 0x0 Trigger when the TX FIFO is empty. 0x1 Trigger when TX FIFO contains â‰¤ 1 byte 0x2 Trigger when TX FIFO contains â‰¤ 2 bytes 0x3 Trigger when TX FIFO â‰¤ 3 bytes 0x4 Trigger when FIFO â‰¤ 4 bytes 0x5 Trigger when FIFO â‰¤ 5 bytes 0x6 Trigger when FIFO â‰¤ 6 bytes 0x7 Trigger when FIFO â‰¤ 7 bytes Reset type: PER.RESET

46.4.2.25 I2CFIFOSTATUS Register (Offset = F08h) [Reset = 00010005h]

I2CFIFOSTATUS is shown in Figure 46-40 and described in Table 46-30.

Return to the Summary Table.

I2C FIFO Status

Figure 46-40. I2CFIFOSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED					RXABVTRIG	RXFF	RXFE
R-0h					R-0h	R-0h	R-1h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					TXBLWTRIG	TXFF	TXFE
R-0h					R-1h	R-0h	R-1h

Table 46-30. I2CFIFOSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved
18	RXABVTRIG	R	0h	RX FIFO Above Trigger Level Value Description 0 The number of bytes in RX FIFO is below the trigger level programmed by the RXTRIG bit in the I2CFIFOCTL register 1 The number of bytes in the RX FIFO is above the trigger level programmed by the RXTRIG bit in the I2CFIFOCTL register Reset type: PER.RESET
17	RXFF	R	0h	RX FIFO Full Value Description 0 The RX FIFO is not full. 1 The RX FIFO is full. Reset type: PER.RESET
16	RXFE	R	1h	RX FIFO Empty Value Description 0 The RX FIFO is not empty. 1 The RX FIFO is empty. Reset type: PER.RESET
15-3	RESERVED	R	0h	Reserved
2	TXBLWTRIG	R	1h	TX FIFO Below Trigger Level Value Description 0 The number of bytes in TX FIFO is above the trigger level programmed by the TXTRIG bit in the I2CFIFOCTL register 1 The number of bytes in the TX FIFO is below the trigger level programmed by the TXTRIG bit in the I2CFIFOCTL register Reset type: PER.RESET
1	TXFF	R	0h	TX FIFO Full Value Description 0 The TX FIFO is not full. 1 The TX FIFO is full. Reset type: PER.RESET

Table 46-30. I2CFIFOSTATUS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	TXFE	R	1h	TX FIFO Empty Value Description 0 The TX FIFO is not empty. 1 The TX FIFO is empty. Reset type: PER.RESET

46.4.2.26 I2CPP Register (Offset = FC0h) [Reset = 1h]

I2CPP is shown in Figure 46-41 and described in Table 46-31.

Return to the Summary Table.

I2C Peripheral Properties

Figure 46-41. I2CPP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															HS
R-0h															R-1h

Table 46-31. I2CPP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	HS	R	1h	High-Speed Capable Value Description 0 The interface is capable of Standard, Fast, or Fast mode plus operation. 1 The interface is capable of High-Speed operation. Reset type: PER.RESET

46.4.2.27 I2CPC Register (Offset = FC4h) [Reset = 1h]

I2CPC is shown in Figure 46-42 and described in Table 46-32.

Return to the Summary Table.

I2C Peripheral Configuration

Figure 46-42. I2CPC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															HS
R-0h															R/W-1h

Table 46-32. I2CPC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	HS	R/W	1h	High-Speed Capable Value Description 0 The interface is set to Standard, Fast or Fast mode plus operation. 1 The interface is set to High-Speed operation. Note that this encoding may only be used if the HS bit in the I2CPP register is set. Otherwise, this encoding is not available. Reset type: PER.RESET

46.4.3 CM_I2C_WRITE_REGS Registers

Table 46-33 lists the memory-mapped registers for the CM_I2C_WRITE_REGS registers. All register offset addresses not listed in Table 46-33 should be considered as reserved locations and the register contents should not be modified.

Table 46-33. CM_I2C_WRITE_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
4h	I2CMCS_WRITE	I2C Master Control/Status		Go
804h	I2CSCSR_WRITE	I2C Slave Control/Status		Go
F00h	I2CFIFODATATX	I2C FIFO Data TX		Go

Complex bit access types are encoded to fit into small table cells. Table 46-34 shows the codes that are used for access types in this section.

Table 46-34. CM_I2C_WRITE_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

46.4.3.1 I2CMCS_WRITE Register (Offset = 4h) [Reset = 0h]

I2CMCS_WRITE is shown in Figure 46-43 and described in Table 46-35.

Return to the Summary Table.

I2C Master Control/Status

Figure 46-43. I2CMCS_WRITE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	BURST	QCMD	HS	ACK	STOP	START	RUN
R-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h

Table 46-35. I2CMCS_WRITE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	BURST	R-0/W	0h	Burst Enable Value Description 0 Burst operation is disabled. 1 The master is enabled to burst using the receive and transmit FIFOs. See field decoding in . Note that the BURST and RUN bits are mutually exclusive. Reset type: PER.RESET
5	QCMD	R-0/W	0h	Quick Command Value Description 0 Bus transaction is not a quick command. 1 The bus transaction is a quick command. To execute a quick command, the START, STOP and RUN bits also need to be set. After the quick command is issued, the master generates a STOP. Reset type: PER.RESET
4	HS	R-0/W	0h	High-Speed Enable Value Description 0 The master operates in Standard, Fast mode, or Fast mode plus as selected by using a value in the I2CMTPR register that results in an SCL frequency of 100 kbps for Standard mode, 400 kbps for Fast mode, or 1 Mpbs for Fast mode plus. 1 The master operates in High-Speed mode with transmission speeds up to 3.33 Mbps. Reset type: PER.RESET
3	ACK	R-0/W	0h	Data Acknowledge Enable Value Description 0 The received data byte is not acknowledged automatically by the master. 1 The received data byte is acknowledged automatically by the master. See field decoding in . Reset type: PER.RESET

Table 46-35. I2CMCS_WRITE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	STOP	R-0/W	0h	Generate STOP Value Description 0 The controller does not generate the STOP condition. 1 The controller generates the STOP condition. See field decoding in . Reset type: PER.RESET
1	START	R-0/W	0h	Generate START Value Description 0 The controller does not generate the START condition. 1 The controller generates the START or repeated START condition. See field decoding in . Reset type: PER.RESET
0	RUN	R-0/W	0h	I2C Master Enable Value Description 0 In standard and high speed mode, this encoding means the master is unable to transmit or receive data. In Burst mode, this bit is not used and must be set to 0. 1 The master is able to transmit or receive data. Note that this bit cannot be set in Burst mode. See field decoding in . Note that the BURST and RUN bits are mutually exclusive. Reset type: PER.RESET

46.4.3.2 I2CSCSR_WRITE Register (Offset = 804h) [Reset = 0h]

I2CSCSR_WRITE is shown in Figure 46-44 and described in Table 46-36.

Return to the Summary Table.

I2C Slave Control/Status

Figure 46-44. I2CSCSR_WRITE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					RXFIFO	TXFIFO	DA
R-0h					R-0/W-0h	R-0/W-0h	R-0/W-0h

Table 46-36. I2CSCSR_WRITE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2	RXFIFO	R-0/W	0h	RX FIFO Enable Value Description 0 Disables RX FIFO 1 Enables RX FIFO Reset type: PER.RESET
1	TXFIFO	R-0/W	0h	TX FIFO Enable Value Description 0 Disables TX FIFO 1 Enables TX FIFO Reset type: PER.RESET
0	DA	R-0/W	0h	Device Active Value Description 0 Disables the I2C slave operation. 1 Enables the I2C slave operation. Once this bit has been set, it should not be set again unless it has been cleared by writing a 0 or by a reset, otherwise transfer failures may occur. Reset type: PER.RESET

46.4.3.3 I2CFIFODATATX Register (Offset = F00h) [Reset = 0h]

I2CFIFODATATX is shown in Figure 46-45 and described in Table 46-37.

Return to the Summary Table.

I2C FIFO Data TX

Figure 46-45. I2CFIFODATATX Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DATA							
R-0h																								R-0/W-0h							

Table 46-37. I2CFIFODATATX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	DATA	R-0/W	0h	I2C TX FIFO Write Data Byte This field contains the current byte written to the TX FIFO. For back to back transmit operations, the application should not switch between writing to the I2CSDR register and the I2CFIFODATA. Reset type: PER.RESET

This page intentionally left blank.

Chapter 47

Synchronous Serial Interface (SSI)

This chapter describes the synchronous serial interface (SSI) module. Refer to the device datasheet for the number of instances present on your device.

47.1 Introduction

Each SSI is a master or slave interface for synchronous serial communication with peripheral devices that have either Freescale SPI, or Texas Instruments Synchronous Serial Interfaces.

47.1.1 Features

The SSI module includes the following features:

- Programmable interface operation for Freescale SPI, or Texas Instruments Synchronous Serial Interfaces. In this SSI module, only the Legacy SSI mode is supported.
- Master or slave operation
- Programmable clock bit rate and prescaler
- Separate transmit and receive FIFOs, each 16-bits wide and 8 locations deep
- Programmable data frame size from 4 to 16 bits
- Internal loopback test mode for diagnostic/debug testing
- Standard FIFO-based interrupts and End-of-Transmission interrupt
- Efficient transfers using Micro Direct Memory Access Controller (µDMA)
  - Separate channels for transmit and receive
  - Receive single request asserted when data is in the FIFO; burst request asserted when FIFO contains four entries
  - Transmit single request asserted when there is space in the FIFO; burst request asserted when FIFO contains four or more entries are available to be written in the FIFO
  - Maskable µDMA interrupts for receive and transmit complete

47.1.2 Block Diagram

The SSI module block diagram is shown in Figure 47-1.

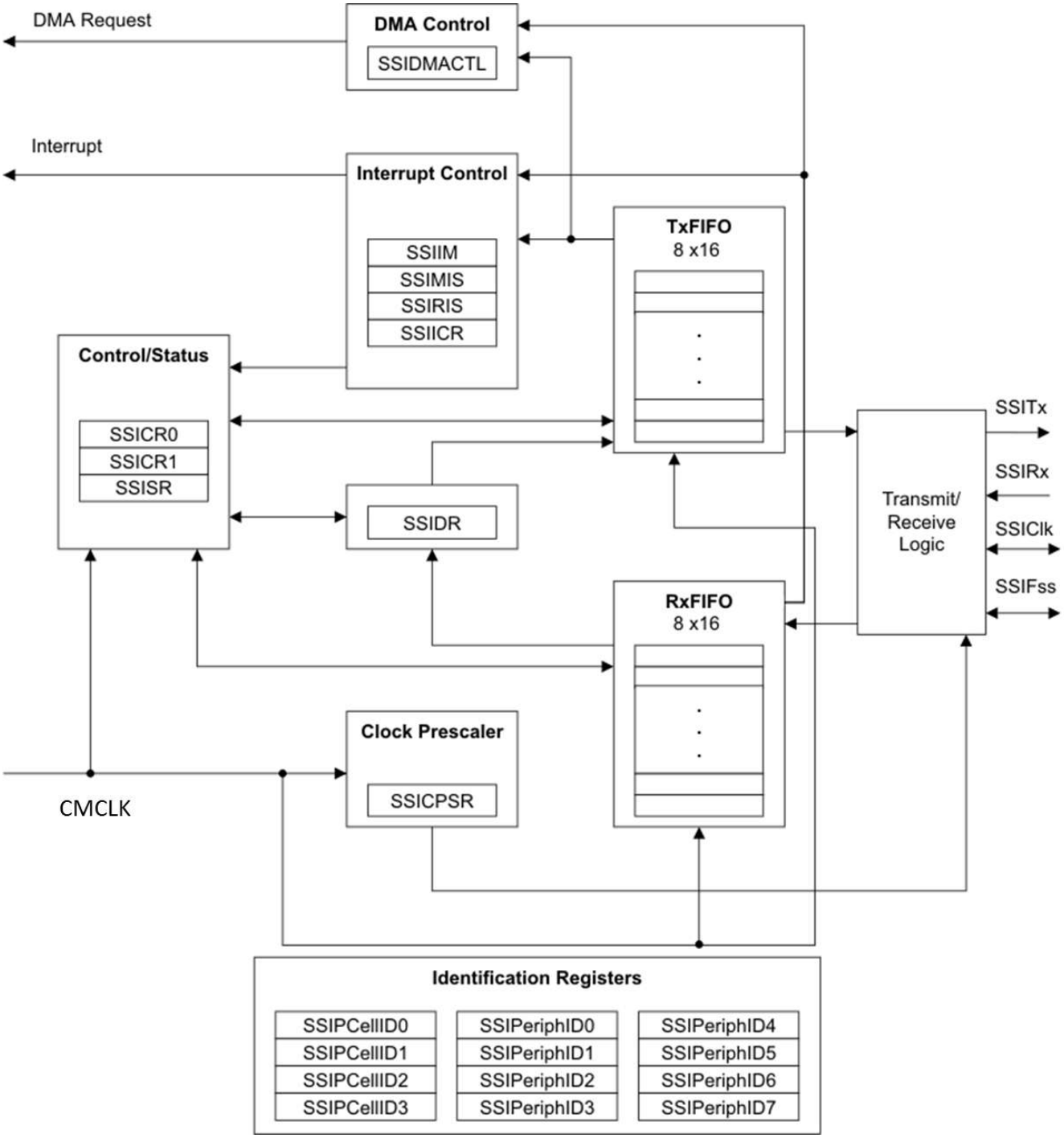


Figure 47-1. SSI Block Diagram

47.2 Functional Description

The SSI performs serial-to-parallel conversion on data received from a peripheral device. The CPU accesses data, control, and status information. The transmit and receive paths are buffered with internal FIFO memories, allowing up to eight 16-bit values to be stored independently in both transmit and receive modes. The SSI also supports the µDMA interface. The transmit and receive FIFOs can be programmed as destination/source addresses in the µDMA module. µDMA operation is enabled by setting the appropriate bit(s) in the SSIDMACTL register.

47.2.1 Bit Rate Generation

The SSI includes a programmable bit rate clock divider and prescaler to generate the serial output clock. Bit rates are supported to 2 MHz and higher, although maximum bit rate is determined by peripheral devices.

The serial bit rate is derived by dividing-down the input clock (CMCLK). The clock is first divided by an even prescale value CPSDVSR from 2 to 254, which is programmed in the SSI Clock Prescale (SSICPSR) register. The clock is further divided by a value from 1 to 256, which is 1 + SCR, where SCR is the value programmed in the SSI Control 0 (SSICR0) register.

The frequency of the output clock SSIClk is defined by:

SSIClk = CMCLK / (CPSDVSR * (1 + SCR)) (27)

Note

For master mode, the system clock must be at least two times faster than the SSIClk, with the restriction that SSIClk cannot be faster than (CMCLK/2).

For slave mode, the system clock must be at least 12 times faster than the SSIClk. In slave mode, maximum frequency of operation is (CMCLK/12).

See the *Electrical Characteristics* chapter in the device data sheet to view SSI timing parameters.

47.2.2 FIFO Operation

47.2.2.1 Transmit FIFO

The common transmit FIFO is a 16-bit wide, 8-locations deep, first-in, first-out memory buffer. The CPU writes data to the FIFO by writing the SSI Data (SSIDR) register, and data is stored in the FIFO until it is read out by the transmission logic.

When configured as a master or a slave, parallel data is written into the transmit FIFO prior to serial conversion and transmission to the attached slave or master, respectively, through the SSITx pin.

In slave mode, the SSI transmits data each time the master initiates a transaction. If the transmit FIFO is empty and the master initiates, the slave transmits the eighth most recent value in the transmit FIFO. If less than eight values have been written to the transmit FIFO since the SSI module clock was enabled using the SSI bit in the CMPCLKCR0 register, then 0 is transmitted. Take care to ensure that valid data is in the FIFO as needed. The SSI can be configured to generate an interrupt or a µDMA request when the FIFO is empty.

47.2.2.2 Receive FIFO

The common receive FIFO is a 16-bit wide, 8-locations deep, first-in, first-out memory buffer. Received data from the serial interface is stored in the buffer until read out by the CPU, which accesses the read FIFO by reading the SSIDR register.

When configured as a master or slave, serial data received through the SSIRx pin is registered prior to parallel loading into the attached slave or master receive FIFO, respectively.

When receiving data in master mode, a dummy write to the SSIDR register must occur before any read occurs.

47.2.3 SSInFSS Function

The SSInFss signal can be programmed to assert low at the start of each byte transfer for one clock or the entire frame. This is configured by programming the FSSHLDFRM bit in the SSICR1 register as shown in Table 47-1. The EOM bit is also provided to signify end of frame transmission. This bit is embedded in the TXFIFO entry for use at the interface to deassert SSInFss at the appropriate time.

Table 47-1. SSInFss Functionality

FSSHLDFRM Bit	Description
0	For Freescale format, with SPH = 0, the SSInFss signal is asserted low between continuous transfers. For SPH = 1, the SSInFss signal is deasserted (high) between continuous transfers. For TI format, the SSInFss signal is deasserted (high) after every data transfer.
1	For Freescale format with any SPH value, the SSInFss signal is forced high between continuous transfers; it is asserted low when there is available data in the Tx FIFO; otherwise, it is forced high to be ready for a new frame.

47.2.4 Interrupts

The SSI can generate interrupts when the following conditions are observed:

- Transmit FIFO service (when the transmit FIFO is half full or less)
- Receive FIFO service (when the receive FIFO is half full or more)
- Receive FIFO time-out
- Receive FIFO overrun
- End of transmission

All of the interrupt events are ORed together before being sent to the interrupt controller, so the SSI generates a single interrupt request to the controller regardless of the number of active interrupts. Each of the four individual maskable interrupts can be masked by clearing the appropriate bit in the SSI Interrupt Mask (SSIIM) register. Setting the appropriate mask bit enables the interrupt.

The individual outputs, along with a combined interrupt output, allow use of either a global interrupt service routine or modular device drivers to handle interrupts. The transmit and receive dynamic dataflow interrupts have been separated from the status interrupts so that data can be read or written in response to the FIFO trigger levels. The status of the individual interrupt sources can be read from the SSI Raw Interrupt Status (SSIRIS) and SSI Masked Interrupt Status (SSIMIS) registers.

The receive FIFO has a time-out period that is 32 periods at the rate of SSIClk (whether or not SSIClk is currently active) and is started when the RX FIFO goes from EMPTY to not-EMPTY. If the RX FIFO is emptied before 32 clocks have passed, the time-out period is reset. As a result, the ISR should clear the Receive FIFO Time-out Interrupt just after reading out the RX FIFO by writing a 1 to the RTIC bit in the SSI Interrupt Clear (SSIICR) register. The interrupt should not be cleared so late that the ISR returns before the interrupt is actually cleared, or the ISR may be re-activated unnecessarily.

The End-of-Transmission (EOT) interrupt indicates that the data has been transmitted completely. This interrupt can be used to indicate when it is safe to turn off the SSI module clock or enter sleep mode. In addition, because transmitted data and received data complete at exactly the same time, the interrupt can also indicate that read data is ready immediately, without waiting for the receive FIFO time-out period to complete.

47.2.5 Frame Formats

Each data frame is between 4- and 16-bits long, depending on the size of data programmed, and is transmitted starting with the MSB. Two basic frame types can be selected:

- Texas Instruments synchronous serial
- Freescale SPI

For all three formats, the serial clock (SSIClk) is held inactive while the SSI is idle, and SSIClk transitions at the programmed frequency only during active transmission or reception of data. The idle state of SSIClk is utilized to provide a receive timeout indication that occurs when the receive FIFO still contains data after a timeout period.

For Freescale SPI frame format, the serial frame (SSIFss) pin is active low, and is asserted (pulled down) during the entire transmission of the frame.

For Texas Instruments synchronous serial frame format, the SSIFss pin is pulsed for one serial clock period starting at its rising edge, prior to the transmission of each frame. For this frame format, both the SSI and the off-chip slave device drive their output data on the rising edge of SSIClk and latch data from the other device on the falling edge.

Figure 47-2 shows the Texas Instruments synchronous serial frame format for a single transmitted frame.

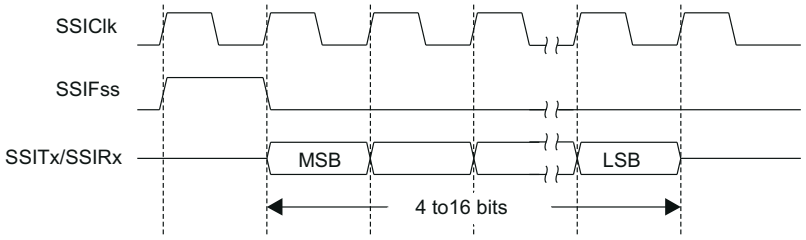


Figure 47-2. TI Synchronous Serial Frame Format (Single Transfer)

In this mode, SSIClk and SSIFss are forced low, and the transmit data line SSITx is tristated whenever the SSI is idle. Once the bottom entry of the transmit FIFO contains data, SSIFss is pulsed high for one SSIClk period. The value to be transmitted is also transferred from the transmit FIFO to the serial shift register of the transmit logic. On the next rising edge of SSIClk, the MSB of the 4- to 16-bit data frame is shifted out on the SSITx pin. Likewise, the MSB of the received data is shifted onto the SSIRx pin by the off-chip serial slave device.

Both the SSI and the off-chip serial slave device then clock each data bit into their serial shifter on each falling edge of SSIClk. The received data is transferred from the serial shifter to the receive FIFO on the first rising edge of SSIClk after the LSB has been latched.

Figure 47-3 shows the Texas Instruments synchronous serial frame format when back-to-back frames are transmitted.

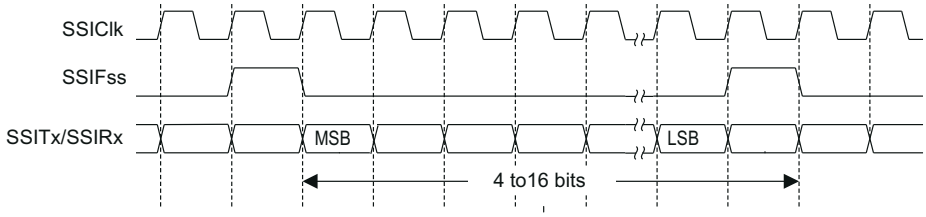


Figure 47-3. TI Synchronous Serial Frame Format (Continuous Transfer)

47.2.5.1 Freescale SPI Frame Format

The Freescale SPI interface is a four-wire interface where the SSIFss signal behaves as a slave select. The main feature of the Freescale SPI format is that the inactive state and phase of the SSIClk signal are programmable through the SPO and SPH bits in the SSICR0 control register.

47.2.5.1.1 SPO Clock Polarity Bit

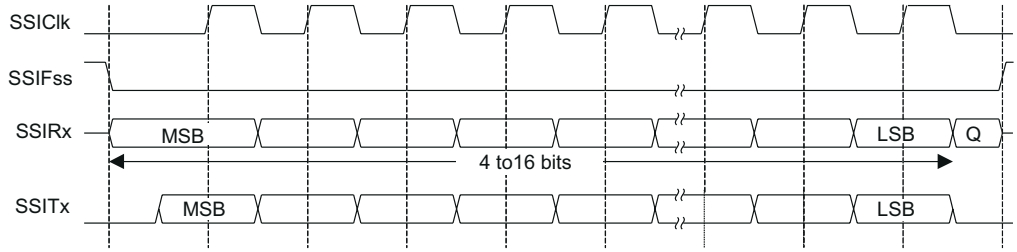
When the SPO clock polarity control bit is clear, it produces a steady state low value on the SSIClk pin. If the SPO bit is set, a steady state high value is placed on the SSIClk pin when data is not being transferred.

47.2.5.1.2 SPH Phase Control Bit

The SPH phase control bit selects the clock edge that captures data and allows it to change state. The state of this bit has the most impact on the first bit transmitted by either allowing or not allowing a clock transition before the first data capture edge. When the SPH phase control bit is clear, data is captured on the first clock edge transition. If the SPH bit is set, data is captured on the second clock edge transition.

47.2.5.2 Freescale SPI Frame Format with SPO=0 and SPH=0

Single and continuous transmission signal sequences for Freescale SPI format with SPO=0 and SPH=0 are shown in Figure 47-4 and Figure 47-5.



Note: Q is undefined.

Figure 47-4. Freescale SPI Format (Single Transfer) with SPO=0 and SPH=0

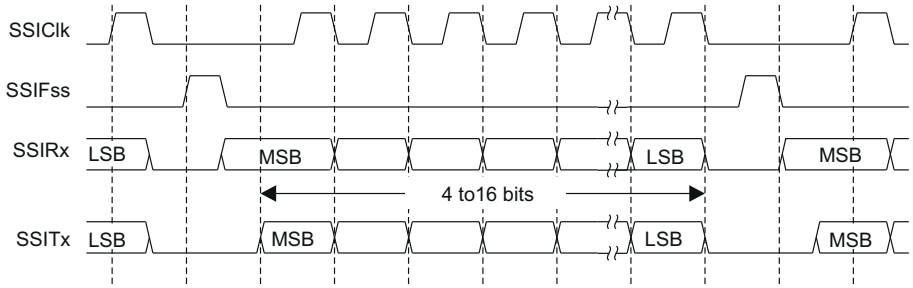


Figure 47-5. Freescale SPI Format (Continuous Transfer) with SPO=0 and SPH=0

In this configuration, during idle periods:

- SSIClk is forced low
- SSIFss is forced high
- The transmit data line SSITx is arbitrarily forced low
- When the SSI is configured as a master, it enables the SSIClk pad
- When the SSI is configured as a slave, it disables the SSIClk pad

If the SSI is enabled and valid data is in the transmit FIFO, the start of transmission is signified by the SSIFss master signal being driven low, causing slave data to be enabled onto the SSIRx input line of the master. The master SSITx output pad is enabled.

One half SSIClk period later, valid master data is transferred to the SSITx pin. Once both the master and slave data have been set, the SSIClk master clock pin goes high after one additional half SSIClk period. The data is now captured on the rising and propagated on the falling edges of the SSIClk signal.

In the case of a single word transmission, after all bits of the data word have been transferred, the SSIFss line is returned to its idle high state one SSIClk period after the last bit has been captured.

However, in the case of continuous back-to-back transmissions, the SSIFss signal must be pulsed high between each data word transfer because the slave select pin freezes the data in its serial peripheral register and does not allow it to be altered if the SPH bit is clear. Therefore, the master device must raise the SSIFss pin of the slave device between each data transfer to enable the serial peripheral data write. On completion of the continuous transfer, the SSIFss pin is returned to its idle state one SSIClk period after the last bit has been captured.

47.2.5.3 Freescale SPI Frame Format with SPO=0 and SPH=1

The transfer signal sequence for Freescale SPI format with SPO=0 and SPH=1 is shown in Figure 47-6, in which covers both single and continuous transfers.

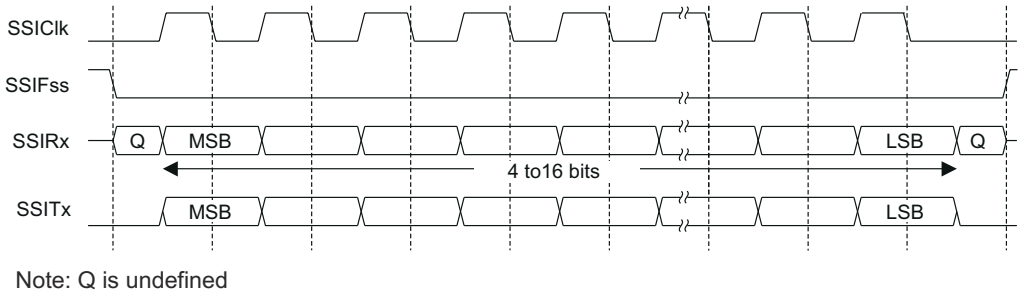


Figure 47-6. Freescale SPI Frame Format with SPO =0 and SPH=1

In this configuration, during idle periods:

- SSIClk is forced low
- SSIFss is forced high
- The transmit data line SSITx is arbitrarily forced low
- When the SSI is configured as a master, it enables the SSIClk pad
- When the SSI is configured as a slave, it disables the SSIClk pad

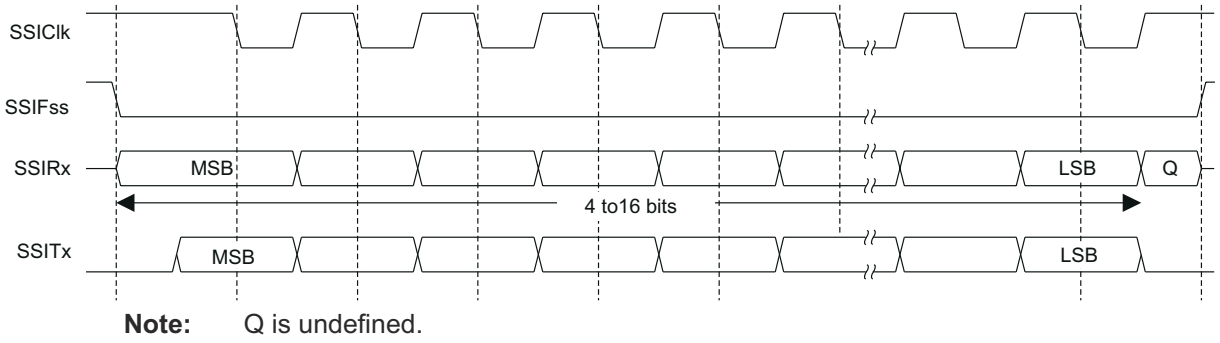
If the SSI is enabled and valid data is in the transmit FIFO, the start of transmission is signified by the SSIFss master signal being driven low. The master SSITx output is then enabled. After an additional one-half SSIClk period, both master and slave valid data are enabled onto their respective transmission lines. At the same time, the SSIClk is enabled with a rising edge transition. Data is then captured on the falling edges and propagated on the rising edges of the SSIClk signal.

In the case of a single word transfer, after all bits have been transferred, the SSIFss line is returned to its idle high state one SSIClk period after the last bit has been captured.

For continuous back-to-back transfers, the SSIFss pin is held low between successive data words, and termination is the same as that of the single word transfer.

47.2.5.4 Freescale SPI Frame Format with SPO=1 and SPH=0

Single and continuous transmission signal sequences for Freescale SPI format with SPO=1 and SPH=0 are shown in Figure 47-7 and Figure 47-8.



Note: Q is undefined.

Figure 47-7. Freescale SPI Frame Format (Single Transfer) with SPO=1 and SPH=0

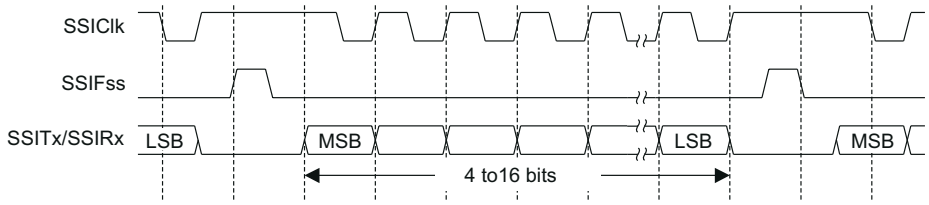


Figure 47-8. Freescale SPI Frame Format (Continuous Transfer) with SPO=1 and SPH=0

In this configuration, during idle periods:

- SSIClk is forced high
- SSIFss is forced high
- The transmit data line SSITx is arbitrarily forced low
- When the SSI is configured as a master, it enables the SSIClk pad
- When the SSI is configured as a slave, it disables the SSIClk pad

If the SSI is enabled and valid data is in the transmit FIFO, the start of transmission is signified by the SSIFss master signal being driven low, causing slave data to be immediately transferred onto the SSIRx line of the master. The master SSITx output pad is enabled.

One-half period later, valid master data is transferred to the SSITx line. Once both the master and slave data have been set, the SSIClk master clock pin becomes low after one additional half SSIClk period, meaning that data is captured on the falling edges and propagated on the rising edges of the SSIClk signal.

In the case of a single word transmission, after all bits of the data word are transferred, the SSIFss line is returned to its idle high state one SSIClk period after the last bit has been captured.

However, in the case of continuous back-to-back transmissions, the SSIFss signal must be pulsed high between each data word transfer because the slave select pin freezes the data in its serial peripheral register and does not allow it to be altered if the SPH bit is clear. Therefore, the master device must raise the SSIFss pin of the slave device between each data transfer to enable the serial peripheral data write. On completion of the continuous transfer, the SSIFss pin is returned to its idle state one SSIClk period after the last bit has been captured.

47.2.5.5 Freescale SPI Frame Format with SPO=1 and SPH=1

The transfer signal sequence for Freescale SPI format with SPO=1 and SPH=1 is shown in Figure 47-9, which covers both single and continuous transfers.

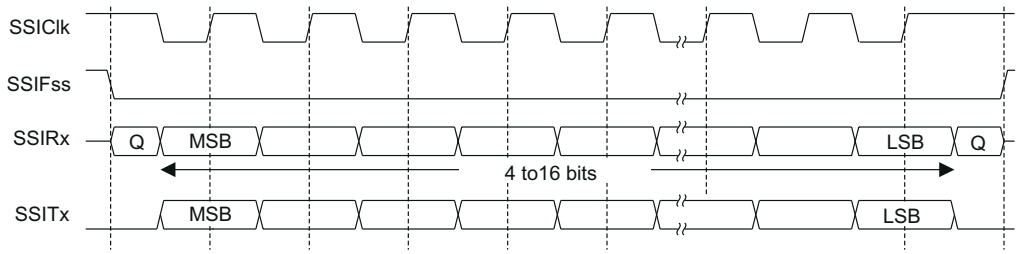


Figure 47-9. Freescale SPI Frame Format with SPO =1 and SPH =1

In this configuration, during idle periods:

- SSIClk is forced hgh
- SSIFss is forced high
- The transmit data line SSITx is arbitrarily forced low
- When the SSI is configured as a master, it enables the SSIClk pad
- When the SSI is configured as a slave, it disables the SSIClk pad

If the SSI is enabled and valid data is in the transmit FIFO, the start of transmission is signified by the SSIFss master signal being driven low. The master SSITx output pad is enabled. After an additional one-half SSIClk period, both master and slave data are enabled onto their respective transmission lines. At the same time, SSIClk is enabled with a falling edge transition. Data is then captured on the rising edges and propagated on the falling edges of the SSIClk signal.

After all bits have been transferred, in the case of a single word transmission, the SSIFss line is returned to its idle high state one SSIClk period after the last bit has been captured.

For continuous back-to-back transmissions, the SSIFss pin remains in its active low state until the final bit of the last word has been captured and then returns to its idle state as described above.

For continuous back-to-back transfers, the SSIFss pin is held low between successive data words and termination is the same as that of the single word transfer.

47.2.6 DMA Operation

The SSI peripheral provides an interface to the µDMA controller with separate channels for transmit and receive. The µDMA operation of the SSI is enabled through the SSI DMA Control (SSIDMACTL) register. When µDMA operation is enabled, the SSI asserts a µDMA request on the receive or transmit channel when the associated FIFO can transfer data. For the receive channel, a single transfer request is asserted whenever any data is in the receive FIFO. A burst transfer request is asserted whenever the amount of data in the receive FIFO is four or more items.

For the transmit channel, a single transfer request is asserted whenever at least one empty location is in the transmit FIFO. The burst request is asserted whenever the transmit FIFO has four or more empty slots. The single and burst µDMA transfer requests are handled automatically by the µDMA controller depending how the µDMA channel is configured. To enable µDMA operation for the receive channel, the RXDMAE bit of the DMA Control (SSIDMACTL) register should be set. To enable µDMA operation for the transmit channel, the TXDMAE bit of SSIDMACTL should be set. If µDMA is enabled, then the µDMA controller triggers an interrupt when a transfer is complete. The interrupt occurs on the SSI interrupt vector. Therefore, if interrupts are used for SSI operation and µDMA is enabled, the SSI interrupt handler must be designed to handle the µDMA completion interrupt.

If the µDMA is enabled and has completed a data transfer from the Tx FIFO, the DMATXRIS bit is set in the SSIRIS register and cannot be cleared by setting the DMATXIC bit in the SSI Interrupt Clear (SSIICR) register. In the DMA Completion Interrupt Service Routine, software must disable the µDMA transmit enable to the SSI by clearing the TXDMAE bit in the SSI DMA Control (SSIDMACTL) register and then setting the DMATXIC bit in the SSIICR register. This clears the DMA completion interrupt. When the µDMA is needed to transmit more data, the TXDMAE bit must be set (enabled) again. If a data transfer by the µDMA from the Rx FIFO completes, the DMARXRIS bit is set. The EOT bit in the SSIRIS register is also provided to indicate when the Tx FIFO is empty and the last bit has been transmitted out of the serializer.

See the *Micro Direct Memory Access (µDMA)* chapter for more details about programming the µDMA controller.

47.3 Initialization and Configuration

To enable and initialize the SSI, the following steps are necessary:

1. Enable the SSI module by setting the SSI bit in the CMPCLKCR0 register.
2. Configure the pinmux.

For each of the frame formats, the SSI is configured using the following steps:

1. Ensure that the SSE bit in the SSICR1 register is clear before making any configuration changes.
2. Select whether the SSI is a master or slave:
   - For master operations, set the SSICR1 register to 0x0000.0000.
   - For slave mode (output enabled), set the SSICR1 register to 0x0000.0004.
   - For slave mode (output disabled), set the SSICR1 register to 0x0000.000C.
3. Configure the clock prescale divisor by writing the SSICPSR register.
4. Write the SSICR0 register with the following configuration:
   - Serial clock rate (SCR)
   - Desired clock phase/polarity, if using Freescale SPI mode (SPH and SPO)
   - Protocol mode: Freescale SPI, TI SSF
   - Data size (DSS)
5. Optionally, configure the µDMA channel (see the *Micro Direct Memory Access (µDMA)* chapter) and enable the DMA option(s) in the SSIDMACTL register.
6. Enable the SSI by setting the SSE bit in the SSICR1 register.

As an example, assume the SSI must be configured to operate with the following parameters:

- Master operation
- Freescale SPI mode (SPO=1, SPH=1)
- 1 Mbps bit rate
- 8 data bits

Assuming the system clock (CMCLK) is 20 MHz, the bit rate calculation would be:

SSIClk = (system clock) / (CPSDVSR * (1 + SCR)) 1 x 106 = 20 x 106 / (CPSDVSR * (1 + SCR)) (28)

In this case, if CPSDVSR=0x2, SCR must be 0x9.

The configuration sequence would be as follows:

1. Ensure that the SSE bit in the SSICR1 register is clear.
2. Write the SSICR1 register with a value of 0x0000.0000.
3. Write the SSICPSR register with a value of 0x0000.0002.
4. Write the SSICR0 register with a value of 0x0000.09C7.
5. The SSI is then enabled by setting the SSE bit in the SSICR1 register.

47.4 Software

47.4.1 SSI Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/ssi

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

47.4.1.1 SSI Loopback example with interrupts - CM

FILE: ssi_ex1_loopback.c

This example showcases how to use SSI to transfer and receive data in loopback mode
cm_common_config_c28x example needs to be run on C28x1 before running this example on the CM core

Configuration:
- Frame format : TI mode
- Baud rate : 625000
- Data width : 12 bits

External Connections
- None

Watch Variables
- *txData* - Data transmitted
- *rxData* - Data received
- *errCount* - Error count

47.4.1.2 SSI Loopback example with UDMA - CM

FILE: ssi_ex2_loopback_udma.c

This example showcases how to use UDMA with SSI to transfer and receive data

This configures the SSI in loopback mode and sends and receives data for infinite time.
cm_common_config_c28x example needs to be run on C28x1 before running this example on the CM core

Configuration:
- Frame format : TI mode
- Baud rate : 625000
- Data width : 16 bits

External Connections
- None

Watch Variables
- *TxData* - Data transmitted
- *RxData* - Data received
- *errCount* - Error count

47.5 SSI Registers

This section describes the Synchronous Serial Interface registers.

47.5.1 SSI Base Addresses

Table 47-2. SSI Base Address Table (CM)

DriverLib Name	Base Address	uDMA Access	Ethernet DMA Access
SSI0_BASE	0x4000_8000	YES	-

47.5.2 SSI_REGS Registers

Table 47-3 lists the memory-mapped registers for the SSI_REGS registers. All register offset addresses not listed in Table 47-3 should be considered as reserved locations and the register contents should not be modified.

Table 47-3. SSI_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	SSICR0	SSI Control 0		Go
4h	SSICR1	SSI Control 1		Go
8h	SSIDR	SSI Data		Go
Ch	SSISR	SSI Status		Go
10h	SSICPSR	SSI Clock Prescale		Go
14h	SSIIM	SSI Interrupt Mask		Go
18h	SSIRIS	SSI Raw Interrupt Status		Go
1Ch	SSIMIS	SSI Masked Interrupt Status		Go
20h	SSIICR	SSI Interrupt Clear		Go
24h	SSIDMACTL	SSI DMA Control		Go
FB0h	SSIPV	SSI Peripheral Version		Go
FC0h	SSIPP	SSI Peripheral Properties		Go
FC4h	SSIPC	SSI Peripheral Configuration		Go
FD0h	SSIPeriphID4	SSI Peripheral Identification 4		Go
FD4h	SSIPeriphID5	SSI Peripheral Identification 5		Go
FD8h	SSIPeriphID6	SSI Peripheral Identification 6		Go
FDCh	SSIPeriphID7	SSI Peripheral Identification 7		Go
FE0h	SSIPeriphID0	SSI Peripheral Identification 0		Go
FE4h	SSIPeriphID1	SSI Peripheral Identification 1		Go
FE8h	SSIPeriphID2	SSI Peripheral Identification 2		Go
FECh	SSIPeriphID3	SSI Peripheral Identification 3		Go
FF0h	SSIPCellID0	SSI PrimeCell Identification 0		Go
FF4h	SSIPCellID1	SSI PrimeCell Identification 1		Go
FF8h	SSIPCellID2	SSI PrimeCell Identification 2		Go
FFCh	SSIPCellID3	SSI PrimeCell Identification 3		Go

Complex bit access types are encoded to fit into small table cells. Table 47-4 shows the codes that are used for access types in this section.

Table 47-4. SSI_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 47-4. SSI_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

47.5.2.1 SSICR0 Register (Offset = 0h) [Reset = 0h]

SSICR0 is shown in Figure 47-10 and described in Table 47-5.

Return to the Summary Table.

SSI Control 0

Figure 47-10. SSICR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCR								SPH	SPO	FRF		DSS			
R/W-0h								R/W-0h	R/W-0h	R/W-0h		R/W-0h			

Table 47-5. SSICR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	SCR	R/W	0h	SSI Serial Clock Rate This bit field is used to generate the transmit and receive bit rate of the SSI. The bit rate is: BR=SysClk/(CPSDVSR * (1 + SCR)) where CPSDVSR is an even value from 2-254 programmed in the SSICPSR register, and SCR is a value from 0-255. Reset type: PER.RESET
7	SPH	R/W	0h	SSI Serial clock PHase This bit is only applicable to the Freescale SPI Format.The control bit selects the clock edge that captures data and allows it to change state. This bit has the most impact on the first bit transmitted by either allowing or not allowing a clock transition before the first data capture edge. SPH Value Description 0 Data is captured on the first clock edge transition. 1 Data is captured on the second clock edge transition. Reset type: PER.RESET
6	SPO	R/W	0h	SSI Serial clock POlarity Value Description 0 A steady state Low value is placed on the pin SSInClk when data is not being transferred. 1 A steady state High value is placed on the pin SSInClk when data is not being transferred. Reset type: PER.RESET
5-4	FRF	R/W	0h	SSI FRame Format Select Value Frame Format 0x0 Freescale SPI Frame Format 0x1 Synchronous Serial Frame Format Texas Instruments 0x2 Reserved 0x3 Reserved Reset type: PER.RESET

Table 47-5. SSICR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	DSS	R/W	0h	SSI Data Size Select Value Data Size 0x0-0x2 Reserved 0x3 4-bit data 0x4 5-bit data 0x5 6-bit data 0x6 7-bit data 0x7 8-bit data 0x8 9-bit data 0x9 10-bit data 0xA 11-bit data 0xB 12-bit data 0xC 13-bit data 0xD 14-bit data 0xE 15-bit data 0xF 16-bit data Reset type: PER.RESET

47.5.2.2 SSICR1 Register (Offset = 4h) [Reset = 0h]

SSICR1 is shown in Figure 47-11 and described in Table 47-6.

Return to the Summary Table.

SSI Control 1

Figure 47-11. SSICR1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			RESERVED	RESERVED	FSSHLDFRM	HSCLKEN	DIR
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED		RESERVED	EOT	RESERVED	MS	SSE	LBM
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 47-6. SSICR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	FSSHLDFRM	R/W	0h	FSS Hold Frame Value Description 0 Pulse SSInFss at every byte (the bit DSS in the SSICR0 register must be set to 0x7 (data size 8 bits) in thisconfiguration) 1 Hold SSInFss for the whole frame Reset type: PER.RESET
9	HSCLKEN	R/W	0h	High Speed Clock Enable High speed clock enable is available only when operating as a master. Value Description 0 Use Input Clock 1 Use High Speed Clock Note: For proper functionality of high speed mode, the HSCLKEN bit in the SSICR1 register should be set before any SSI data transfer or after applying a reset to the QSSI module. In addition, the SSE bit must be set to 0x1 before the HSCLKEN bit is set. Reset type: PER.RESET
8	DIR	R/W	0h	SSI Direction of Operation Value Description 0 TX (Transmit Mode) write direction 1 RX (Receive Mode) read direction Reset type: PER.RESET
7-6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved

Table 47-6. SSICR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	EOT	R/W	0h	End of Transmission This bit is only valid for Master mode devices and operations (=0x0).MS Value Description 0 The TXRIS interrupt indicates that the transmit FIFO is half full or less. 1 The End of Transmit interrupt mode for the TXRIS interrupt is enabled.When using uDMA, the DMATX bit cannot be set to 1 in any mode. If used with uDMA, it prevents RISEOT from asserting. If the bit is kept at 0 during operation, an interrupt is still generated when the TX FIFO is half or less full with or without using the uDMA.EOT (Legacy) Reset type: PER.RESET
3	RESERVED	R/W	0h	Reserved
2	MS	R/W	0h	SSI Master/Slave Select This bit selects Master or Slave mode and can be modified only when the SSI is disabled (SSE=0). Value Description 0 The SSI is configured as a master. 1 The SSI is configured as a slave. Reset type: PER.RESET
1	SSE	R/W	0h	SSI Synchronous Serial Port Enable Value Description 0 SSI operation is disabled. 1 SSI operation is enabled. This bit must be cleared before any control registers are reprogrammed. The bit HSCLKEN in the SSICR1 register should be set only after applying reset to the QSSI module and enabling the QSSI by setting the SSE bit, and before any SSI data transfer. All other bits in the SSICR1 register and all bits in SSICR0 register can only be programmed when the SSE is clear. Reset type: PER.RESET
0	LBM	R/W	0h	SSI Loopback Mode Value Description 0 Normal serial port operation enabled. 1 Output of the transmit serial shift register is connected internally to the input of the receive serial shift register. Reset type: PER.RESET

47.5.2.3 SSIDR Register (Offset = 8h) [Reset = 0h]

SSIDR is shown in Figure 47-12 and described in Table 47-7.

Return to the Summary Table.

SSI Data

Figure 47-12. SSIDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																DATA															
R-0h																R/W-0h															

Table 47-7. SSIDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	DATA	R/W	0h	SSI Receive/Transmit Data A read operation reads the receive FIFO. A write operation writes the transmit FIFO.Software must right-justify data when the SSI is programmed for a data size that is less than 16 bits. Unused bits at the top are ignored by the transmit logic. The receive logic automatically right-justifies the data. Reset type: PER.RESET

47.5.2.4 SSISR Register (Offset = Ch) [Reset = 3h]

SSISR is shown in Figure 47-13 and described in Table 47-8.

Return to the Summary Table.

SSI Status

Figure 47-13. SSISR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											BSY	RFF	RNE	TNF	TFE
R-0h											R-0h	R-0h	R-0h	R-1h	R-1h

Table 47-8. SSISR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4	BSY	R	0h	SSI Busy Bit Value Description 0 The SSI is idle. 1 The SSI is currently transmitting and/or receiving a frame, or the transmit FIFO is not empty. Reset type: PER.RESET
3	RFF	R	0h	SSI Receive FIFO Full Value Description 0 The receive FIFO is not full. 1 The receive FIFO is full. Reset type: PER.RESET
2	RNE	R	0h	SSI Receive FIFO Not Empty Value Description 0 The receive FIFO is empty. 1 The receive FIFO is not empty. Reset type: PER.RESET
1	TNF	R	1h	SSI Transmit FIFO Not Full Value Description 0 The transmit FIFO is full. 1 The transmit FIFO is not full. Reset type: PER.RESET
0	TFE	R	1h	SSI Transmit FIFO Empty Value Description 0 The transmit FIFO is not empty. 1 The transmit FIFO is empty. Reset type: PER.RESET

47.5.2.5 SSICPSR Register (Offset = 10h) [Reset = 0h]

SSICPSR is shown in Figure 47-14 and described in Table 47-9.

Return to the Summary Table.

SSI Clock Prescale

Figure 47-14. SSICPSR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CPSDVSR							
R-0h																								R/W-0h							

Table 47-9. SSICPSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	CPSDVSR	R/W	0h	SSI Clock Prescale Divisor This value must be an even number from 2 to 254, depending on the frequency of SSInClk. The LSB always returns 0 on reads. Reset type: PER.RESET

47.5.2.6 SSIIM Register (Offset = 14h) [Reset = 0h]

SSIIM is shown in Figure 47-15 and described in Table 47-10.

Return to the Summary Table.

SSI Interrupt Mask

Figure 47-15. SSIIM Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	EOTIM	DMATXIM	DMARXIM	TXIM	RXIM	RTIM	RORIM
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 47-10. SSIIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	EOTIM	R/W	0h	End of Transmit Interrupt Mask Value Description 0 The end of transmit interrupt is masked. 1 The end of transmit interrupt is not masked. Reset type: PER.RESET
5	DMATXIM	R/W	0h	SSI Transmit DMA Interrupt Mask Value Description 0 The transmit DMA interrupt is masked. 1 The transmit DMA interrupt is not masked. Reset type: PER.RESET
4	DMARXIM	R/W	0h	SSI Receive DMA Interrupt Mask Value Description 0 The receive DMA interrupt is masked. 1 The receive DMA interrupt is not masked. Reset type: PER.RESET
3	TXIM	R/W	0h	SSI Transmit FIFO Interrupt Mask Value Description 0 The transmit FIFO interrupt is masked. 1 The transmit FIFO interrupt is not masked. Reset type: PER.RESET
2	RXIM	R/W	0h	SSI Receive FIFO Interrupt Mask Value Description 0 The receive FIFO interrupt is masked. 1 The receive FIFO interrupt is not masked. Reset type: PER.RESET
1	RTIM	R/W	0h	SSI Receive Time-Out Interrupt Mask Value Description 0 The receive FIFO time-out interrupt is masked. 1 The receive FIFO time-out interrupt is not masked. Reset type: PER.RESET

Table 47-10. SSIIM Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	RORIM	R/W	0h	SSI Receive Overrun Interrupt Mask Value Description 0 The receive FIFO overrun interrupt is masked. 1 The receive FIFO overrun interrupt is not masked. Reset type: PER.RESET

47.5.2.7 SSIRIS Register (Offset = 18h) [Reset = 8h]

SSIRIS is shown in Figure 47-16 and described in Table 47-11.

Return to the Summary Table.

SSI Raw Interrupt Status

Figure 47-16. SSIRIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	EOTRIS	DMATXRIS	DMARXRIS	TXRIS	RXRIS	RTRIS	RORRIS
R-0h	R-0h	R-0h	R-0h	R-1h	R-0h	R-0h	R-0h

Table 47-11. SSIRIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	EOTRIS	R	0h	End of Transmit Raw Interrupt Status Value Description 0 No interrupt. 1 The transmit FIFO is empty, and the last bit has been transmitted out of the serializer.This bit is cleared when a 1 is written to the EOTIC bit in the SSI Interrupt Clear (SSIICR) register. Reset type: PER.RESET
5	DMATXRIS	R	0h	SSI Transmit DMA Raw Interrupt Status Value Description 0 No interrupt. 1 The transmit DMA has completed.This bit is cleared when a 1 is written to the DMATXIC bit in the SSI Interrupt Clear (SSIICR) register. Reset type: PER.RESET
4	DMARXRIS	R	0h	SSI Receive DMA Raw Interrupt Status Value Description 0 No interrupt. 1 The receive DMA has completed.This bit is cleared when a 1 is written to the DMARXIC bit in the SSI Interrupt Clear (SSIICR) register. Reset type: PER.RESET
3	TXRIS	R	1h	SSI Transmit FIFO Raw Interrupt Status Value Description 0 No interrupt. 1 The transmit FIFO is half empty or less. If the EOT bit in the SSICR1 register is clear, If the EOT bit is set, the transmit FIFO is empty, and the last bit has been transmitted out of the serializer.This bit is cleared when the transmit FIFO is more than half full. (if the EOT bit is clear) or when it has any data in it (if the EOT bit is set) Reset type: PER.RESET

Table 47-11. SSIRIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	RXRIS	R	0h	SSI Receive FIFO Raw Interrupt Status Value Description 0 No interrupt. 1 The receive FIFO is half full or more.This bit is cleared when the receive FIFO is less than half full. Reset type: PER.RESET
1	RTRIS	R	0h	SSI Receive Time-Out Raw Interrupt Status Value Description 0 No interrupt. 1 The receive time-out has occurred.This bit is cleared when a 1 is written to the RTIC bit in the SSI Interrupt Clear (SSIICR) register. Reset type: PER.RESET
0	RORRIS	R	0h	SSI Receive Overrun Raw Interrupt Status Value Description 0 No interrupt. 1 The receive FIFO has overflowed. This bit is cleared when a 1 is written to the RORIC bit in the SSI Interrupt Clear (SSIICR) register. Reset type: PER.RESET

47.5.2.8 SSIMIS Register (Offset = 1Ch) [Reset = 0h]

SSIMIS is shown in Figure 47-17 and described in Table 47-12.

Return to the Summary Table.

SSI Masked Interrupt Status

Figure 47-17. SSIMIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	EOTMIS	DMATXMIS	DMARXMIS	TXMIS	RXMIS	RTMIS	RORMIS
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 47-12. SSIMIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	EOTMIS	R	0h	End of Transmit Masked Interrupt Status Value Description 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to the transmission of the last data bit.This bit is cleared when a 1 is written to the EOTIC bit in the SSI Interrupt Clear (SSIICR) register. Reset type: PER.RESET
5	DMATXMIS	R	0h	SSI Transmit DMA Masked Interrupt Status Value Description 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due tothe completion of the transmit DMA.This bit is cleared when a 1 is written to the DMATXIC bit in the SSI Interrupt Clear (SSIICR) register. Reset type: PER.RESET
4	DMARXMIS	R	0h	SSI Receive DMA Masked Interrupt Status Value Description 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to the completion of the receive DMA.This bit is cleared when a 1 is written to the DMARXIC bit in the SSI Interrupt Clear (SSIICR) register. Reset type: PER.RESET
3	TXMIS	R	0h	SSI Transmit FIFO Masked Interrupt Status Value Description 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to the transmit FIFO being half empty or less. (if the EOT bit is clear) or due to the transmission of the last data bit (if the EOT bit is set)This bit is cleared when the transmit FIFO is more than half empty .(if the EOT bit is clear) or when it has any data in it (if the EOT bit is set) Reset type: PER.RESET

Table 47-12. SSIMIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	RXMIS	R	0h	SSI Receive FIFO Masked Interrupt Status Value Description 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to the receive FIFO being half full or more.This bit is cleared when the receive FIFO is less than half full. Reset type: PER.RESET
1	RTMIS	R	0h	SSI Receive Time-Out Masked Interrupt Status Value Description 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to the receive time out.This bit is cleared when a 1 is written to the RTIC bit in the SSI Interrupt Clear (SSIICR) register. Reset type: PER.RESET
0	RORMIS	R	0h	SSI Receive Overrun Masked Interrupt Status Value Description 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to the receive FIFO overflowing.This bit is cleared when a 1 is written to the RORIC bit in the SSI Interrupt Clear (SSIICR) register. Reset type: PER.RESET

47.5.2.9 SSIICR Register (Offset = 20h) [Reset = 0h]

SSIICR is shown in Figure 47-18 and described in Table 47-13.

Return to the Summary Table.

SSI Interrupt Clear

Figure 47-18. SSIICR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	EOTIC	DMATXIC	DMARXIC	RESERVED		RTIC	RORIC
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0h		R-0/W1S-0h	R-0/W1S-0h

Table 47-13. SSIICR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	EOTIC	R-0/W1S	0h	End of Transmit Interrupt Clear Writing a 1 to this bit clears the EOTRIS bit in the SSIRIS register and the EOTMIS bit in the SSIMIS register. Reset type: PER.RESET
5	DMATXIC	R-0/W1S	0h	SSI Transmit DMA Interrupt Clear Writing a 1 to this bit clears the DMATXRIS bit in the SSIRIS register and the DMATXMIS bit in the SSIMIS register. Reset type: PER.RESET
4	DMARXIC	R-0/W1S	0h	SSI Receive DMA Interrupt Clear Writing a 1 to this bit clears the DMARXRIS bit in the SSIRIS register and the DMARXMIS bit in the SSIMIS register. Reset type: PER.RESET
3-2	RESERVED	R	0h	Reserved
1	RTIC	R-0/W1S	0h	SSI Receive Time-Out Interrupt Clear Writing a 1 to this bit clears the RTRIS bit in the SSIRIS register and the RTMIS bit in the SSIMIS register. Reset type: PER.RESET
0	RORIC	R-0/W1S	0h	SSI Receive Overrun Interrupt Clear Writing a 1 to this bit clears the RORRIS bit in the SSIRIS register and the RORMIS bit in the SSIMIS register. Reset type: PER.RESET

47.5.2.10 SSIDMACTL Register (Offset = 24h) [Reset = 0h]

SSIDMACTL is shown in Figure 47-19 and described in Table 47-14.

Return to the Summary Table.

SSI DMA Control

Figure 47-19. SSIDMACTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						TXDMAE	RXDMAE
R-0h						R/W-0h	R/W-0h

Table 47-14. SSIDMACTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	TXDMAE	R/W	0h	Transmit DMA Enable Value Description 0 uDMA for the transmit FIFO is disabled. 1 uDMA for the transmit FIFO is enabled. Reset type: PER.RESET
0	RXDMAE	R/W	0h	Receive DMA Enable Value Description 0 uDMA for the receive FIFO is disabled. 1 uDMA for the receive FIFO is enabled. Reset type: PER.RESET

47.5.2.11 SSIPV Register (Offset = FB0h) [Reset = 400h]

SSIPV is shown in Figure 47-20 and described in Table 47-15.

Return to the Summary Table.

SSI Peripheral Version

Figure 47-20. SSIPV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																MAJOR								MINOR							
R-0h																R-4h								R-0h							

Table 47-15. SSIPV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	MAJOR	R	4h	Major Revision This field specifies the major revision number of the module. Corresponds to the IP version used on Snowflake. Reset type: PER.RESET
7-0	MINOR	R	0h	Minor Revision This field specifies the minor revision number of the module. Corresponds to the IP version used on Snowflake. Reset type: PER.RESET

47.5.2.12 SSIPP Register (Offset = FC0h) [Reset = 9h]

SSIPP is shown in Figure 47-21 and described in Table 47-16.

Return to the Summary Table.

SSI Peripheral Properties

Figure 47-21. SSIPP Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	FSSHLDFRM	MODE		HSCLK
R-0h			R-0h	R-1h	R-0h		R-1h

Table 47-16. SSIPP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	FSSHLDFRM	R	1h	SSInFss Hold Frame Capability Value Description 0 Hold Frame capability disabled.SSInFss 1 Hold Frame capability enabled.SSinFss Reset type: PER.RESET
2-1	MODE	R	0h	Mode of Operation Indicates what SSI functionality is supported. Value Description 0x0 Legacy SSI mode Others reserved Reset type: PER.RESET
0	HSCLK	R	1h	High Speed Capability Value Description 0 High Speed clock capability disabled. 1 High speed clock capability enabled. Reset type: PER.RESET

47.5.2.13 SSIPC Register (Offset = FC4h) [Reset = 0h]

SSIPC is shown in Figure 47-22 and described in Table 47-17.

Return to the Summary Table.

SSI Peripheral Configuration

Figure 47-22. SSIPC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																															
R-0h																															

Table 47-17. SSIPC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESERVED	R	0h	Reserved

47.5.2.14 SSIPeriphID4 Register (Offset = FD0h) [Reset = 0h]

SSIPeriphID4 is shown in Figure 47-23 and described in Table 47-18.

Return to the Summary Table.

SSI Peripheral Identification 4

Figure 47-23. SSIPeriphID4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID4							
R-0h																								R-0h							

Table 47-18. SSIPeriphID4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID4	R	0h	SSI Peripheral ID Register [7:0] Can be used by software to identify the presence of this peripheral. Reset type: PER.RESET

47.5.2.15 SSIPeriphID5 Register (Offset = FD4h) [Reset = 0h]

SSIPeriphID5 is shown in Figure 47-24 and described in Table 47-19.

Return to the Summary Table.

SSI Peripheral Identification 5

Figure 47-24. SSIPeriphID5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID5							
R-0h																								R-0h							

Table 47-19. SSIPeriphID5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID5	R	0h	SSI Peripheral ID Register [15:8] Can be used by software to identify the presence of this peripheral. Reset type: PER.RESET

47.5.2.16 SSIPeriphID6 Register (Offset = FD8h) [Reset = 0h]

SSIPeriphID6 is shown in Figure 47-25 and described in Table 47-20.

Return to the Summary Table.

SSI Peripheral Identification 6

Figure 47-25. SSIPeriphID6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID6							
R-0h																								R-0h							

Table 47-20. SSIPeriphID6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID6	R	0h	SSI Peripheral ID Register [23:16] Can be used by software to identify the presence of this peripheral. Reset type: PER.RESET

47.5.2.17 SSIPeriphID7 Register (Offset = FDCh) [Reset = 0h]

SSIPeriphID7 is shown in Figure 47-26 and described in Table 47-21.

Return to the Summary Table.

SSI Peripheral Identification 7

Figure 47-26. SSIPeriphID7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID7							
R-0h																								R-0h							

Table 47-21. SSIPeriphID7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID7	R	0h	SSI Peripheral ID Register [31:24] Can be used by software to identify the presence of this peripheral. Reset type: PER.RESET

47.5.2.18 SSIPeriphID0 Register (Offset = FE0h) [Reset = 22h]

SSIPeriphID0 is shown in Figure 47-27 and described in Table 47-22.

Return to the Summary Table.

SSI Peripheral Identification 0

Figure 47-27. SSIPeriphID0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID0							
R-0h																								R-22h							

Table 47-22. SSIPeriphID0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID0	R	22h	SSI Peripheral ID Register [7:0] Can be used by software to identify the presence of this peripheral. Reset type: PER.RESET

47.5.2.19 SSIPeriphID1 Register (Offset = FE4h) [Reset = 0h]

SSIPeriphID1 is shown in Figure 47-28 and described in Table 47-23.

Return to the Summary Table.

SSI Peripheral Identification 1

Figure 47-28. SSIPeriphID1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID1							
R-0h																								R-0h							

Table 47-23. SSIPeriphID1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID1	R	0h	SSI Peripheral ID Register [15:8] Can be used by software to identify the presence of this peripheral. Reset type: PER.RESET

47.5.2.20 SSIPeriphID2 Register (Offset = FE8h) [Reset = 18h]

SSIPeriphID2 is shown in Figure 47-29 and described in Table 47-24.

Return to the Summary Table.

SSI Peripheral Identification 2

Figure 47-29. SSIPeriphID2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID2							
R-0h																								R-18h							

Table 47-24. SSIPeriphID2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID2	R	18h	SSI Peripheral ID Register [23:16] Can be used by software to identify the presence of this peripheral. Reset type: PER.RESET

47.5.2.21 SSIPeriphID3 Register (Offset = FECh) [Reset = 1h]

SSIPeriphID3 is shown in Figure 47-30 and described in Table 47-25.

Return to the Summary Table.

SSI Peripheral Identification 3

Figure 47-30. SSIPeriphID3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID3							
R-0h																								R-1h							

Table 47-25. SSIPeriphID3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID3	R	1h	SSI Peripheral ID Register [31:24] Can be used by software to identify the presence of this peripheral. Reset type: PER.RESET

47.5.2.22 SSIPCellID0 Register (Offset = FF0h) [Reset = Dh]

SSIPCellID0 is shown in Figure 47-31 and described in Table 47-26.

Return to the Summary Table.

SSI PrimeCell Identification 0

Figure 47-31. SSIPCellID0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CID0							
R-0h																								R-Dh							

Table 47-26. SSIPCellID0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	CID0	R	Dh	SSI PrimeCell ID Register [7:0] Provides software a standard cross-peripheral identification system. Reset type: PER.RESET

47.5.2.23 SSIPCellID1 Register (Offset = FF4h) [Reset = F0h]

SSIPCellID1 is shown in Figure 47-32 and described in Table 47-27.

Return to the Summary Table.

SSI PrimeCell Identification 1

Figure 47-32. SSIPCellID1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CID1							
R-0h																								R-F0h							

Table 47-27. SSIPCellID1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	CID1	R	F0h	SSI PrimeCell ID Register [15:8] Provides software a standard cross-peripheral identification system. Reset type: PER.RESET

47.5.2.24 SSIPCellID2 Register (Offset = FF8h) [Reset = 5h]

SSIPCellID2 is shown in Figure 47-33 and described in Table 47-28.

Return to the Summary Table.

SSI PrimeCell Identification 2

Figure 47-33. SSIPCellID2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CID2							
R-0h																								R-5h							

Table 47-28. SSIPCellID2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	CID2	R	5h	SSI PrimeCell ID Register [23:16] Provides software a standard cross-peripheral identification system. Reset type: PER.RESET

47.5.2.25 SSIPCellID3 Register (Offset = FFCh) [Reset = B1h]

SSIPCellID3 is shown in Figure 47-34 and described in Table 47-29.

Return to the Summary Table.

SSI PrimeCell Identification 3

Figure 47-34. SSIPCellID3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CID3							
R-0h																								R-B1h							

Table 47-29. SSIPCellID3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	CID3	R	B1h	SSI PrimeCell ID Register [31:24] Provides software a standard cross-peripheral identification system. Reset type: PER.RESET

This page intentionally left blank.

Chapter 48

Universal Asynchronous Receiver/Transmitter (UART)

This chapter describes the Universal Asynchronous Receivers/Transmitters (UARTs).

48.1 Introduction

The UART module performs the functions of parallel-to-serial and serial-to-parallel conversions.

48.1.1 Features

The Universal Asynchronous Receiver/Transmitter (UART) module in this device contains the following features:

- Programmable baud-rate generator allowing speeds up to 7.8125 Mbps for regular speed (divide by 16) and 15.625 Mbps for high speed (divide by 8)
- Separate 16-deep and 8-bit wide transmit (TX) and receive (RX) FIFOs to reduce CPU interrupt service loading
- Programmable FIFO length, including 1-byte deep operation providing conventional double-buffered interface
- FIFO trigger levels of ⅛, ¼, ½, ¾, and ⅞
- Standard asynchronous communication bits for start, stop, and parity
- Line-break generation and detection
- Fully programmable serial interface characteristics
  - 5, 6, 7, or 8 data bits
  - Even, odd, stick, or no parity bit generation and detection
  - 1 or 2 stop bit generation
- IrDA serial-IR (SIR) encoder and decoder providing
  - Programmable use of IrDA SIR or UART input/output
  - Support of IrDA SIR encoder and decoder functions for data rates up to 115.2 kbps half-duplex
  - Support of normal 3/16 and low-power (1.41 to 2.23 µs) bit durations
  - Programmable internal clock generator enabling division of reference clock by 1 to 256 for low-power mode bit duration
- EIA-485 9-bit support
- Standard FIFO-level and End-of-Transmission (EOT) interrupts
- Efficient transfers using Micro Direct Memory Access Controller (µDMA)
  - Separate channels for transmit and receive
  - Receive single request asserted when data is in the FIFO; burst request asserted at programmed FIFO level
  - Transmit single request asserted when there is space in the FIFO; burst request asserted at programmed FIFO level
- CMCLK (125-MHz maximum) is used to generate the baud clock.

48.1.2 Block Diagram

Figure 48-1 shows the UART block diagram.

48.2 Functional Description

The UART module performs the functions of parallel-to-serial and serial-to-parallel conversions. The UART module is similar in functionality to a 16C550 UART, but is not register-compatible.

The UART is configured for transmit or receive through the TXE and RXE bits of the UART Control (UARTCTL) register. Transmit and receive are both enabled out of reset. Before any control registers are programmed, the UART must be disabled by clearing the UARTEN bit in UARTCTL. If the UART is disabled during a TX or RX operation, the current transaction is completed prior to the UART stopping.

The UART module also includes a serial IR (SIR) encoder and decoder block that can be connected to an infrared transceiver to implement an IrDA SIR physical layer. The SIR function is programmed using the UARTCTL register.

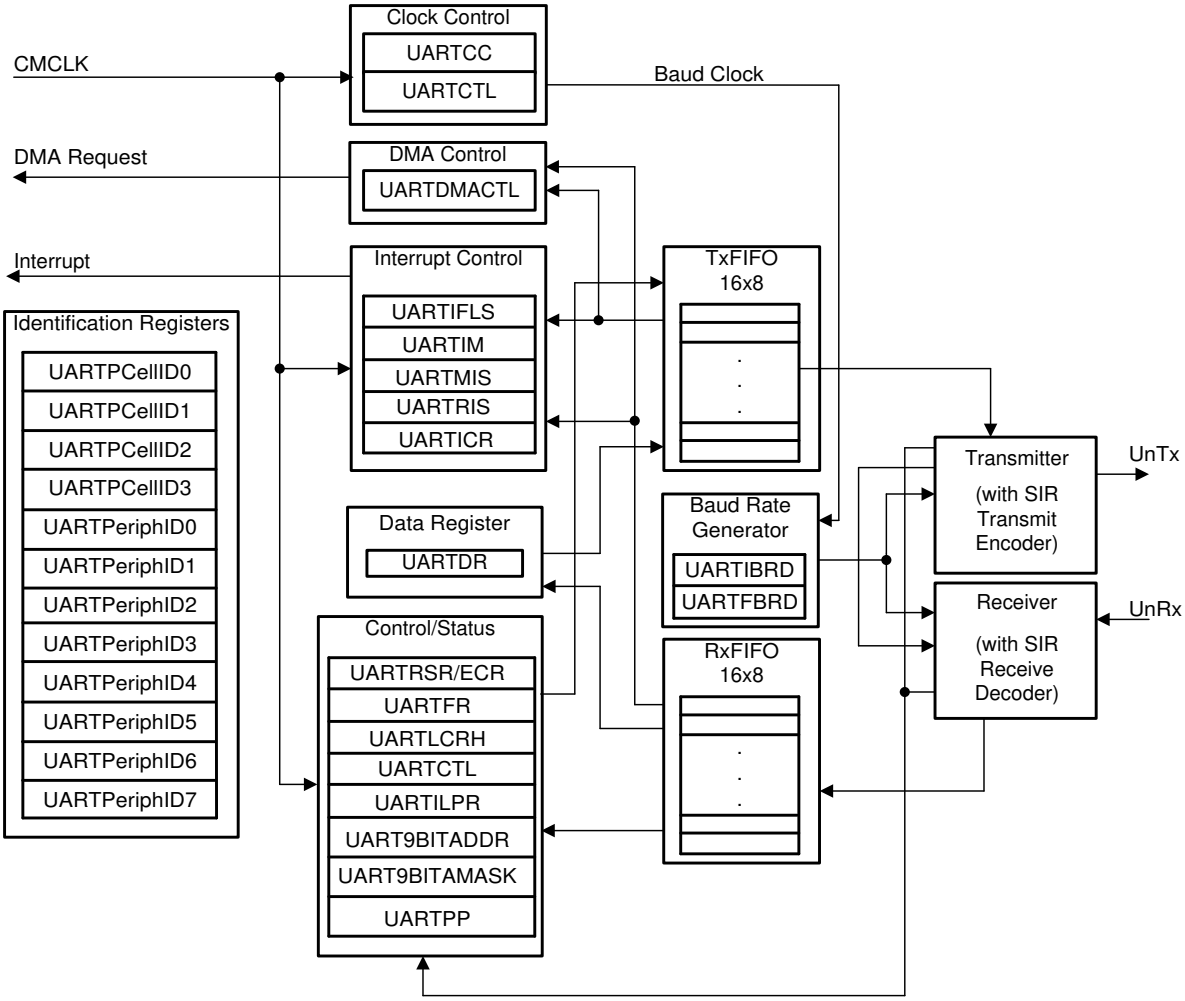


Figure 48-1. UART Module Block Diagram

48.2.1 Transmit and Receive Logic

The transmit logic performs parallel-to-serial conversion on the data read from the transmit FIFO. The control logic outputs the serial bit stream beginning with a start bit and followed by the data bits (LSB first), parity bit, and the stop bits according to the programmed configuration in the control registers. See Figure 48-2 for details.

The receive logic performs serial-to-parallel conversion on the received bit stream after a valid start pulse has been detected. Overrun, parity, frame error checking, and line-break detection are also performed, and their status accompanies the data that is written to the receive FIFO.

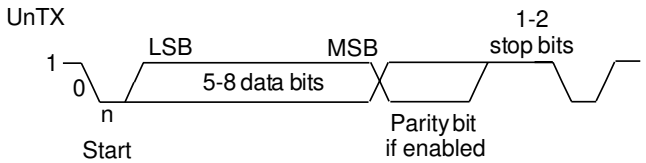


Figure 48-2. UART Character Frame

48.2.2 Baud-Rate Generation

The baud-rate divisor is a 22-bit number consisting of a 16-bit integer and a 6-bit fractional part. The number formed by these two values is used by the baud-rate generator to determine the bit period. Having a fractional baud-rate divisor allows the UART to generate all the standard baud rates.

The 16-bit integer is loaded through the UART Integer Baud-Rate Divisor (UARTIBRD) register and the 6-bit fractional part is loaded with the UART Fractional Baud-Rate Divisor (UARTFBRD) register. The baud-rate divisor (BRD) has the following relationship to the system clock (where BRDI is the integer part of the BRD, and BRDF is the fractional part, separated by a decimal place.)

$$BRD = BRDI + BRDF = UARTSysClk / (ClkDiv \times Baud\ Rate) \quad (29)$$

where

UARTSysClk is the system clock (CMCLK) connected to the UART, and ClkDiv is either 16 (if HSE in UARTCTL is clear) or 8 (if HSE in UARTCTL is set).

By default, this is the main system clock (CMCLK).

The 6-bit fractional number (that is to be loaded into the DIVFRAC bit field in the UARTFBRD register) can be calculated by taking the fractional part of the baud-rate divisor, multiplying by 64, and adding 0.5 to account for rounding errors:

$$UARTFBRD[DIVFRAC] = integer(BRDF \times 64 + 0.5) \quad (30)$$

The UART generates an internal baud-rate reference clock at 8× or 16× the baud-rate (referred to as Baud8 and Baud16, depending on the setting of the HSE bit [bit 5] in UARTCTL). This reference clock is divided by 8 or 16 to generate the transmit clock, and is used for error detection during receive operations.

Along with the UART Line Control High Byte (UARTLCRH) register, the UARTIBRD and UARTFBRD registers form an internal 30-bit register. This internal register is only updated when a write operation to UARTLCRH is performed, so any changes to the baud-rate divisor must be followed by a write to the UARTLCRH register for the changes to take effect.

To update the baud-rate registers, there are four possible sequences:

- UARTIBRD write, UARTFBRD write, and UARTLCRH write
- UARTFBRD write, UARTIBRD write, and UARTLCRH write
- UARTIBRD write and UARTLCRH write
- UARTFBRD write and UARTLCRH write

48.2.3 Data Transmission

Data received or transmitted is stored in two 16-byte FIFOs, though the receive FIFO has an extra four bits per character for status information. For transmission, data is written into the transmit FIFO. If the UART is enabled, it causes a data frame to start transmitting with the parameters indicated in the UARTLCRH register. Data continues to be transmitted until there is no data left in the transmit FIFO. The BUSY bit in the UART Flag (UARTFR) register is asserted as soon as data is written to the transmit FIFO (that is, if the FIFO is non-empty) and remains asserted while data is being transmitted. The BUSY bit is negated only when the transmit FIFO is empty, and the last character has been transmitted from the shift register, including the stop bits. The UART can indicate that it is busy even though the UART may no longer be enabled.

When the receiver is idle (the UnRx signal is continuously 1), and the data input goes Low (a start bit has been received), the receive counter begins running and data is sampled on the eighth cycle of Baud16 or fourth cycle of Baud8 depending on the setting of the HSE bit (bit 5) in UARTCTL (described in Section 48.2.1).

The start bit is valid and recognized if the UnRx signal is still low on the eighth cycle of Baud16 (HSE clear) or the fourth cycle of Baud8 (HSE set), otherwise it is ignored. After a valid start bit is detected, successive data bits are sampled on every 16th cycle of Baud16 or eighth cycle of Baud8 (that is, one bit period later) according to the programmed length of the data characters and value of the HSE bit in UARTCTL. The parity bit is then checked if parity mode is enabled. Data length and parity are defined in the UARTLCRH register.

Lastly, a valid stop bit is confirmed if the UnRx signal is High, otherwise a framing error has occurred. When a full word is received, the data is stored in the receive FIFO along with any error bits associated with that word.

48.2.4 Serial IR (SIR)

The UART peripheral includes an IrDA SIR encoder and decoder block. The IrDA SIR block provides functionality that converts between an asynchronous UART data stream and a half-duplex serial SIR interface. No analog processing is performed on-chip. The role of the SIR block is to provide a digital encoded output and decoded input to the UART. When enabled, the SIR block uses the UnTx and UnRx pins for the SIR protocol. These signals must be connected to an infrared transceiver to implement an IrDA SIR physical layer link. The SIR block can receive and transmit, but the SIR block is only half-duplex so the SIR block cannot do both at the same time. Transmission must be stopped before data can be received. The IrDA SIR physical layer specifies a minimum 10-ms delay between transmission and reception. The SIR block has two modes of operation:

- In normal IrDA mode, a zero logic level is transmitted as a high pulse of 3/16th duration of the selected baud rate bit period on the output pin, while logic one levels are transmitted as a static low signal. These levels control the driver of an infrared transmitter, sending a pulse of light for each zero. On the reception side, the incoming light pulses energize the photo transistor base of the receiver, pulling the output low and driving the UART input pin low.
- In low-power IrDA mode, the width of the transmitted infrared pulse is set to three times the period of the internally generated IrLPBaud16 signal (1.63 µs, assuming a nominal 1.8432-MHz frequency) by changing the appropriate bit in the UARTCTL register.

Whether the device is in normal or low-power IrDA mode, a start bit is deemed valid if the decoder is still low one period of IrLPBaud16 after the low was first detected. This enables a normal-mode UART to receive data from a low-power mode UART that can transmit pulses as small as 1.41 µs. Thus, for both low-power and normal mode operation, the ILPDVSR field in the UARTILPR register must be programmed such that 1.42 MHz < $f_{IrLPBaud16}$< 2.12 MHz, resulting in a low-power pulse duration of 1.41 to 2.11 µs (three times the period of IrLPBaud16). The minimum frequency of IrLPBaud16 make sure that pulses less than one period of IrLPBaud16 are rejected, but pulses greater than 1.4 µs are accepted as valid pulses.

Figure 48-3 shows the UART transmit and receive signals, with and without IrDA modulation.

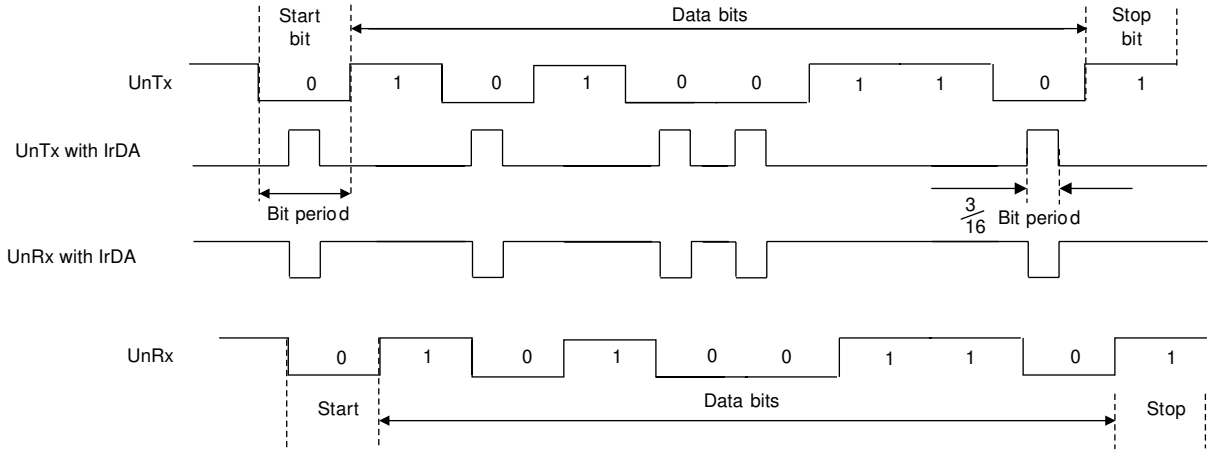


Figure 48-3. IrDA Data Modulation

In both normal and low-power IrDA modes:

- During transmission, the UART data bit is used as the base for encoding
- During reception, the decoded bits are transferred to the UART receive logic

The IrDA SIR physical layer specifies a half-duplex communication link, with a minimum 10-ms delay between transmission and reception. This delay must be generated by software because the delay is not automatically supported by the UART. The delay is required because the infrared receiver electronics can become biased or even saturated from the optical power coupled from the adjacent transmitter LED. This delay is known as latency or receiver setup time.

48.2.5 9-Bit UART Mode

The UART provides a 9-bit mode that is enabled with the 9BITEN bit in the UART9BITADDR register. This feature is useful in a multi-drop configuration of the UART, where a single transmitter connected to multiple receivers can communicate with a particular receiver through the address or set of addresses along with a qualifier for an address byte. All the receivers check for the address qualifier in the place of the parity bit and, if set, then compare the byte received with the preprogrammed address. If the address matches, then the receiver receives or sends further data. If the address does not match, the receiver drops the address byte and any subsequent data bytes. If the UART is in 9-bit mode, then the receiver operates with no parity mode. The address can be predefined to match with the received byte and the address can be configured with the UART9BITADDR register. The matching can be extended to a set of addresses using the address mask in the UART9BITAMASK register. By default, the UART9BITAMASK is 0xFF, meaning that only the specified address is matched.

When not finding a match, the rest of the data bytes with the ninth bit cleared are dropped. If a match is found, then an interrupt is generated to the NVIC for further action. The subsequent data bytes with the cleared ninth bit are stored in the FIFO. Software can mask this interrupt in case µDMA or FIFO operations are enabled for this instance and processor intervention is not required. All the send transactions with 9-bit mode are data bytes and the ninth bit is cleared. Software can override the ninth bit to be set (to indicate address) by overriding the parity settings to sticky parity with odd parity enabled for a particular byte. To match the transmission time with correct parity settings, the address byte can be transmitted as a single then a burst transfer. The transmit FIFO does not hold the address/data bit; hence, software can enable the address bit appropriately.

48.2.6 FIFO Operation

The UART has two 16-deep 8-bit wide FIFOs; one for transmit and one for receive. Both FIFOs are accessed by way of the UART Data (UARTDR) register. Read operations of the UARTDR register return a 12-bit value consisting of eight data bits and four error flags while write operations place 8-bit data in the transmit FIFO.

Out of reset, both FIFOs are disabled and act as 1 byte-deep holding registers. The FIFOs are enabled by setting the FEN bit in the UARTLCRH register.

FIFO status can be monitored through the UART Flag (UARTFR) register and the UART Receive Status (UARTRSR) register. Hardware monitors empty, full, and overrun conditions. The UARTFR register contains empty and full flags (TXFE, TXFF, RXFE, and RXFF bits), and the UARTRSR register shows overrun status with the OE bit. If the FIFOs are disabled, the empty and full flags are set according to the status of the 1 byte-deep holding registers.

The trigger points, at which the FIFOs generate interrupts, is controlled by way of the UART Interrupt FIFO Level Select (UARTIFLS) register. Both FIFOs can be individually configured to trigger interrupts at different levels. Available configurations include ⅛, ¼, ½, ¾, and ⅞. For example, if the ¼ option is selected for the receive FIFO, the UART generates a receive interrupt after four data bytes are received. Out of reset, both FIFOs are configured to trigger an interrupt at the ½ mark.

48.2.7 Interrupts

The UART can generate interrupts when the following conditions are observed:

- Overrun error
- Break error
- Parity error
- Framing error
- Receive time-out
- Transmit (when condition defined in the TXIFLSEL bit in the UARTIFLS register is met, or if the EOT bit in UARTCTL is set, when the last bit of all transmitted data leaves the serializer)
- Receive (when condition defined in the RXIFLSEL bit in the UARTIFLS register is met)

All of the interrupt events are ORed together before being sent to the interrupt controller, so the UART can only generate a single interrupt request to the controller at any given time. Software can service multiple interrupt events in a single interrupt service routine by reading the UART Masked Interrupt Status (UARTMIS) register.

The interrupt events that can trigger a controller-level interrupt are defined in the UART Interrupt Mask (UARTIM) register by setting the corresponding IM bits. If interrupts are not used, the raw interrupt status is visible by way of the UART Raw Interrupt Status (UARTRIS) register.

Note

For receive time-out, the RTIM bit in the UARTIM register must be set to see the RTMIS and RTRIS status in the UARTMIS and UARTRIS registers.

Interrupts are always cleared (for the UARTMIS and UARTRIS registers) by writing a 1 to the corresponding bit in the UART Interrupt Clear (UARTICR) register.

The receive time-out interrupt is asserted when the receive FIFO is not empty, and no further data is received over a 32-bit period when the HSE bit is clear or over a 64-bit period when the HSE bit is set. The receive time-out interrupt is cleared either when the FIFO becomes empty through reading all the data (or by reading the holding register), or when a 1 is written to the corresponding bit in the UARTICR register.

The receive interrupt changes state when one of the following events occurs:

- If the FIFOs are enabled and the receive FIFO reaches the programmed trigger level, the RXRIS bit is set. The receive interrupt is cleared by reading data from the receive FIFO until it becomes less than the trigger level, or by clearing the interrupt by writing a 1 to the RXIC bit.

- If the FIFOs are disabled (have a depth of one location) and data is received thereby filling the location, the RXRIS bit is set. The receive interrupt is cleared by performing a single read of the receive FIFO, or by clearing the interrupt by writing a 1 to the RXIC bit.

The transmit interrupt changes state when one of the following events occurs:

- If the FIFOs are enabled and the transmit FIFO progresses through the programmed trigger level, the TXRIS bit is set. The transmit interrupt is based on a transition through level, therefore the FIFO must be written past the programmed trigger level otherwise no further transmit interrupts will be generated. The transmit interrupt is cleared by writing data to the transmit FIFO until it becomes greater than the trigger level, or by clearing the interrupt by writing a 1 to the TXIC bit.
- If the FIFOs are disabled (have a depth of one location) and there is no data present in the transmitters single location, the TXRIS bit is set. It is cleared by performing a single write to the transmit FIFO, or by clearing the interrupt by writing a 1 to the TXIC bit.

48.2.8 Loopback Operation

The UART can be placed into an internal loopback mode for diagnostic or debug work by setting the LBE bit in the UARTCTL register. In loopback mode, data transmitted on the UnTx output is received on the UnRx input. Note that the LBE bit should be set before the UART is enabled.

48.2.9 DMA Operation

The UART provides an interface to the µDMA controller with separate channels for transmit and receive. The DMA operation of the UART is enabled through the UART DMA Control (UARTDMACTL) register. When DMA operation is enabled, the UART asserts a DMA request on the receive or transmit channel when the associated FIFO can transfer data.

For the receive channel, a single transfer request is asserted whenever any data is in the receive FIFO. A burst transfer request is asserted whenever the amount of data in the receive FIFO is at or above the FIFO trigger level configured in the UARTIFLS register.

For the transmit channel, a single transfer request is asserted whenever there is at least one empty location in the transmit FIFO. The burst request is asserted whenever the transmit FIFO contains fewer characters than the FIFO trigger level. The single and burst DMA transfer requests are handled automatically by the µDMA controller depending on how the DMA channel is configured.

To enable DMA operation for the receive channel, set the RXDMAE bit of the DMA Control (UARTDMACTL) register. To enable DMA operation for the transmit channel, set the TXDMAE bit of the UARTDMACTL register. The UART can also be configured to stop using DMA for the receive channel if a receive error occurs. If the DMAERR bit of the UARTDMACR register is set and a receive error occurs, the DMA receive requests are automatically disabled. This error condition can be cleared by clearing the appropriate UART error interrupt.

When the µDMA is finished transferring data to the TX FIFO or from the RX FIFO, a dma_done signal is sent to the UART to indicate completion. The dma_done status is indicated through the DMATXRIS and DMARXIS bits of the UARTRIS register. An interrupt can be generated from these status bits by setting the DMATXIM and DMARXIM bits in the UARTIM register.

Note

The DMATXRIS bit can be used to indicate the completion of data transfer from the µDMA to the TX FIFO. To indicate transfer completion from the serializer of the UART, the EOT bit can be enabled in the UARTCTL register.

See Chapter 49 for more details about programming the µDMA controller.

48.3 Initialization and Configuration

To enable and initialize the UART, perform the following steps:

1. Disable the UART by clearing the UARTEN bit in the UARTCTL register.
2. Write the integer portion of the BRD to the UARTIBRD register.
3. Write the fractional portion of the BRD to the UARTFBRD register.
4. Write the desired serial parameters to the UARTLCRH register.
5. Optionally, configure the µDMA channel (see Chapter 49) and enable the DMA options in the UARTDMACTL register.
6. Enable the UART by setting the UARTEN bit in the UARTCTL register.

48.4 Software

48.4.1 UART Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/uart

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

48.4.1.1 UART Echoback - CM

FILE: uart_ex1_echoback.c

This test receives and echo-backs data through the UART0 port.

A terminal such as 'putty' can be used to view the data from the CM-UART and to send information to the CM-UART. Characters received by the CM-UART port are sent back to the host.

Running the Application Open a COM port with the following settings using a terminal:
- Find correct COM port
- Bits per second = 115200
- Data Bits = 8
- Parity = None
- Stop Bits = 1
- Hardware Control = None

The program will print out a greeting and then ask you to enter a character which it will echo back to the terminal.

Watch Variables
- None

External Connections
Connect the UART0 port to a PC via a transceiver and cable.
- GPIO85 is UART0RX/CMUARTRXA(Connect to Pin3, PC-TX, of serial DB9 cable)
- GPIO84 is UART0TX/CMUARTTXA(Connect to Pin2, PC-RX, of serial DB9 cable)

The pin muxing for the UART0 port needs to be done by the master CPU1. The common configuration example provided in the C28x folder can be used for making GPIO85 as the UART Rx pin and GPIO84 as the UART Tx pin.

48.4.1.2 UART Loopback example with UDMA - CM

FILE: uart_ex2_loopback_udma.c

This example showcases how to use UDMA with UART to transfer and receive data

This configures the UART in loopback mode and sends and receives data for infinite time.
cm_common_config_c28x example needs to be run on C28x1 before running this example on the CM core

Configuration:
- Find correct COM port
- Bits per second = 115200
- Data Bits = 8
- Parity = None
- Stop Bits = 1
- Hardware Control = None

External Connections
- None

Watch Variables
- *TxData* - Data transmitted
- *RxData* - Data received

- *errCount* - Error count

48.5 UART Registers

This section describes the Connectivity Manager Universal Asynchronous Interface Registers.

48.5.1 UART Base Addresses

Table 48-1. UART Base Address Table (CM)

DriverLib Name	Base Address	uDMA Access	Ethernet DMA Access
UART0_BASE	0x4000_C000	YES	-

48.5.2 UART_REGS Registers

Table 48-2 lists the memory-mapped registers for the UART_REGS registers. All register offset addresses not listed in Table 48-2 should be considered as reserved locations and the register contents should not be modified.

Table 48-2. UART_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	UARTDR	UART Data		Go
4h	UARTRSR	UART Receive Status/Error Clear		Go
18h	UARTFR	UART Flag		Go
20h	UARTILPR	UART IrDA Low-Power Register		Go
24h	UARTIBRD	UART Integer Baud-Rate Divisor		Go
28h	UARTFBRD	UART Fractional Baud-Rate Divisor		Go
2Ch	UARTLCRH	UART Line Control		Go
30h	UARTCTL	UART Control		Go
34h	UARTIFLS	UART Interrupt FIFO Level Select		Go
38h	UARTIM	UART Interrupt Mask		Go
3Ch	UARTRIS	UART Raw Interrupt Status		Go
40h	UARTMIS	UART Masked Interrupt Status		Go
44h	UARTICR	UART Interrupt Clear		Go
48h	UARTDMACTL	UART DMA Control		Go
A4h	UART9BITADDR	UART 9-Bit Self Address		Go
A8h	UART9BITAMASK	UART 9-Bit Self Address Mask		Go
FC0h	UARTPP	UART Peripheral Properties		Go
FD0h	UARTPeriphID4	UART Peripheral Identification 4		Go
FD4h	UARTPeriphID5	UART Peripheral Identification 5		Go
FD8h	UARTPeriphID6	UART Peripheral Identification 6		Go
FDCh	UARTPeriphID7	UART Peripheral Identification 7		Go
FE0h	UARTPeriphID0	UART Peripheral Identification 0		Go
FE4h	UARTPeriphID1	UART Peripheral Identification 1		Go
FE8h	UARTPeriphID2	UART Peripheral Identification 2		Go
FECh	UARTPeriphID3	UART Peripheral Identification 3		Go
FF0h	UARTPCellID0	UART PrimeCell Identification 0		Go
FF4h	UARTPCellID1	UART PrimeCell Identification 1		Go
FF8h	UARTPCellID2	UART PrimeCell Identification 2		Go
FFCh	UARTPCellID3	UART PrimeCell Identification 3		Go

Complex bit access types are encoded to fit into small table cells. Table 48-3 shows the codes that are used for access types in this section.

Table 48-3. UART_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set

Table 48-3. UART_REGS Access Type Codes (continued)

Access Type	Code	Description
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

48.5.2.1 UARTDR Register (Offset = 0h) [Reset = 0h]

UARTDR is shown in Figure 48-4 and described in Table 48-4.

Return to the Summary Table.

IMPORTANT: This register is read sensitive. This register is the data register (the interface to the FIFOs). For transmitted data, if the FIFO is enabled, data written to this location is pushed onto the transmit FIFO. If the FIFO is disabled, data is stored in the transmitter holding register (the bottom word of the transmit FIFO). A write to this register initiates a transmission from the UART.
For received data, if the FIFO is enabled, the data byte and the 4-bit status (break, frame, parity, and overrun) is pushed onto the 12-bit wide receive FIFO. If the FIFO is disabled, the data byte and status are stored in the receiving holding register (the bottom word of the receive FIFO). The received data can be retrieved by reading this register.

Figure 48-4. UARTDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				OE	BE	PE	FE	DATA							
R-0h				R-0h	R-0h	R-0h	R-0h	R/W-0h							

Table 48-4. UARTDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11	OE	R	0h	UART Overrun Error 0 No data has been lost due to a FIFO overrun. 1 New data was received when the FIFO was full, resulting in data loss. Reset type: PER.RESET
10	BE	R	0h	UART Break Error 0 No break condition has occurred 1 A break condition has been detected, indicating that the receive data input was held Low for longer than a full-word transmission time (defined as start, data, parity, and stop bits). In FIFO mode, this error is associated with the character at the top of the FIFO. When a break occurs, only one 0 character is loaded into the FIFO. The next character is only enabled after the received data input goes to a 1 (marking state), and the next valid start bit is received. Reset type: PER.RESET
9	PE	R	0h	UART Parity Error 0 No parity error has occurred 1 The parity of the received data character does not match the parity defined by bits 2 and 7 of the UARTLCRH register. In FIFO mode, this error is associated with the character at the top of the FIFO. Reset type: PER.RESET
8	FE	R	0h	UART Framing Error 0 No framing error has occurred 1 The received character does not have a valid stop bit (a valid stop bit is 1). Reset type: PER.RESET

Table 48-4. UARTDR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-0	DATA	R/W	0h	Data Transmitted or Received Data that is to be transmitted via the UART is written to this field. When read, this field contains the data that was received by the UART. For transmitted data, if the FIFO is enabled, data written to this location is pushed onto the transmit FIFO. If the FIFO is disabled, data is stored in the transmitter holding register (the bottom word of the transmit FIFO). A write to this register initiates a transmission from the UART. For received data, if the FIFO is enabled, the data byte and the 4-bit status (break, frame, parity, and overrun) is pushed onto the 12-bit wide receive FIFO. If the FIFO is disabled, the data byte and status are stored in the receiving holding register (the bottom word of the receive FIFO). The received data can be retrieved by reading this register. Reset type: PER.RESET

48.5.2.2 UARTRSR Register (Offset = 4h) [Reset = 0h]

UARTRSR is shown in Figure 48-5 and described in Table 48-5.

Return to the Summary Table.

The UARTRSR/UARTECR register is the receive status register/error clear register.
In addition to the UARTDR register, receive status can also be read from the UARTRSR register.
If the status is read from this register, then the status information corresponds to the entry read from UARTDR prior to reading UARTRSR. The status information for overrun is set immediately when an overrun condition occurs.
The UARTRSR register cannot be written.
A write of any value to the UARTECR register clears the framing, parity, break, and overrun errors.
All the bits are cleared on reset.

Figure 48-5. UARTRSR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												OE	BE	PE	FE
R-0h												R-0h	R-0h	R-0h	R-0h

Table 48-5. UARTRSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	OE	R	0h	UART Overrun Error 0 No data has been lost due to a FIFO overrun. 1 New data was received when the FIFO was full, resulting in data loss. This bit is cleared by a write to UARTECR.The FIFO contents remain valid because no further data is written when the FIFO is full, only the contents of the shift register are overwritten. The CPU must read the data in order to empty the FIFO. Reset type: PER.RESET
2	BE	R	0h	UART Break Error 0 No break condition has occurred 1 A break condition has been detected, indicating that the receive data input was held Low for longer than a full-word transmission time (defined as start, data, parity, and stop bits). This bit is cleared to 0 by a write to UARTECR. In FIFO mode, this error is associated with the character at the top of the FIFO. When a break occurs, only one 0 character is loaded into the FIFO. The next character is only enabled after the receive data input goes to a 1 (marking state) and the next valid start bit is received. Reset type: PER.RESET
1	PE	R	0h	UART Parity Error 0 No parity error has occurred 1 The parity of the received data character does not match the parity defined by bits 2 and 7 of the UARTLCRH register. This bit is cleared to 0 by a write to UARTECR. Reset type: PER.RESET

Table 48-5. UARTRSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	FE	R	0h	UART Framing Error 0 No framing error has occurred 1 The received character does not have a valid stop bit (a valid stop bit is 1). This bit is cleared to 0 by a write to UARTECR. In FIFO mode, this error is associated with the character at the top of the FIFO. Reset type: PER.RESET

48.5.2.3 UARTFR Register (Offset = 18h) [Reset = 90h]

UARTFR is shown in Figure 48-6 and described in Table 48-6.

Return to the Summary Table.

The UARTFR register is the flag register. After reset, the TXFF, RXFF, and BUSY bits are 0, and TXFE and RXFE bits are 1.

Figure 48-6. UARTFR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							RESERVED
R-0h							R-0h
7	6	5	4	3	2	1	0
TXFE	RXFF	TXFF	RXFE	BUSY	RESERVED	RESERVED	RESERVED
R-1h	R-0h	R-0h	R-1h	R-0h	R-0h	R-0h	R-0h

Table 48-6. UARTFR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	TXFE	R	1h	UART Transmit FIFO Empty The meaning of this bit depends on the state of the FEN bit in the UARTLCRH register. 0 The transmitter has data to transmit. 1 If the FIFO is disabled (FEN is 0), the transmit holding register is empty. If the FIFO is enabled (FEN is 1), the transmit FIFO is empty. Reset type: PER.RESET
6	RXFF	R	0h	UART Receive FIFO Full The meaning of this bit depends on the state of the FEN bit in the UARTLCRH register. 0 The receiver can receive data. 1 If the FIFO is disabled (FEN is 0), the receive holding register is full. If the FIFO is enabled (FEN is 1), the receive FIFO is full. Reset type: PER.RESET
5	TXFF	R	0h	UART Transmit FIFO Full The meaning of this bit depends on the state of the FEN bit in the UARTLCRH register. 0 The transmitter is not full. 1 If the FIFO is disabled (FEN is 0), the transmit holding register is full. If the FIFO is enabled (FEN is 1), the transmit FIFO is full. Reset type: PER.RESET

Table 48-6. UARTFR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	RXFE	R	1h	UART Receive FIFO Empty The meaning of this bit depends on the state of the FEN bit in the UARTLCRH register. 0 The receiver is not empty. 1 If the FIFO is disabled (FEN is 0), the receive holding register is empty. If the FIFO is enabled (FEN is 1), the receive FIFO is empty. Reset type: PER.RESET
3	BUSY	R	0h	UART Busy 0 The UART is not busy. 1 The UART is busy transmitting data. This bit remains set until the complete byte, including all stop bits, has been sent from the shift register. This bit is set as soon as the transmit FIFO becomes non-empty (regardless of whether UART is enabled). Reset type: PER.RESET
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

48.5.2.4 UARTILPR Register (Offset = 20h) [Reset = 0h]

UARTILPR is shown in Figure 48-7 and described in Table 48-7.

Return to the Summary Table.

The UARTILPR register stores the 8-bit low-power counter divisor value used to derive the low-power SIR pulse width clock.

Figure 48-7. UARTILPR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								ILPDVSR							
R-0h																								R/W-0h							

Table 48-7. UARTILPR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	ILPDVSR	R/W	0h	IrDA Low-Power Divisor This field contains the 8-bit low-power divisor value. The UARTILPR register stores the 8-bit low-power counter divisor value used to derive the low-power SIR pulse width clock by dividing down the system clock (SysClk). All the bits are cleared when reset. The internal IrLPBaud16 clock is generated by dividing down SysClk according to the low-power divisor value written to UARTILPR. The duration of SIR pulses generated when low-power mode is enabled is three times the period of the IrLPBaud16 clock. The low-power divisor value is calculated as follows: ILPDVSR = SysClk / FIrLPBaud16 where FIrLPBaud16 is nominally 1.8432 MHz. Because the IrLPBaud16 clock is used to sample transmitted data irrespective of mode, the ILPDVSR field must be programmed in both low power and normal mode,such that FIrLPBaud16 is between 1.42 and 2.12 MHz, resulting in a low-power pulse duration of 1.41-2.11 us (three times the period of IrLPBaud16). The minimum frequency of IrLPBaud16 ensures that pulses less than one period of IrLPBaud16 are rejected, but pulses greater than 1.4 us are accepted as valid pulses. Note: Zero is an illegal value. Programming a zero value results in no IrLPBaud16 pulses being generated Reset type: PER.RESET

48.5.2.5 UARTIBRD Register (Offset = 24h) [Reset = 0h]

UARTIBRD is shown in Figure 48-8 and described in Table 48-8.

Return to the Summary Table.

The UARTIBRD register is the integer part of the baud-rate divisor value. All the bits are cleared on reset. The minimum possible divide ratio is 1 (when UARTIBRD=0), in which case the UARTFBRD register is ignored. When changing the UARTIBRD register, the new value does not take effect until transmission/reception of the current character is complete. Any changes to the baud-rate divisor must be followed by a write to the UARTLCRH register.

Figure 48-8. UARTIBRD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																DIVINT															
R-0h																R/W-0h															

Table 48-8. UARTIBRD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	DIVINT	R/W	0h	Integer Baud-Rate Divisor The minimum possible divide ratio is 1 (when UARTIBRD=0), in which case the UARTFBRD register is ignored. When changing the UARTIBRD register, the new value does not take effect until transmission/reception of the current character is complete. Any changes to the baud-rate divisor must be followed by a write to the UARTLCRH register. Reset type: PER.RESET

48.5.2.6 UARTFBRD Register (Offset = 28h) [Reset = 0h]

UARTFBRD is shown in Figure 48-9 and described in Table 48-9.

Return to the Summary Table.

The UARTFBRD register is the fractional part of the baud-rate divisor value. All the bits are cleared on reset. When changing the UARTFBRD register, the new value does not take effect until transmission/reception of the current character is complete. Any changes to the baud-rate divisor must be followed by a write to the UARTLCRH register. See "Baud-Rate Generation" on page 1165 for configuration details.

Figure 48-9. UARTFBRD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED										DIVFRAC					
R-0h										R/W-0h					

Table 48-9. UARTFBRD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5-0	DIVFRAC	R/W	0h	Fractional Baud-Rate Divisor The UARTFBRD register is the fractional part of the baud-rate divisor value. All the bits are cleared on reset. When changing the UARTFBRD register, the new value does not take effect until transmission/reception of the current character is complete. Any changes to the baud-rate divisor must be followed by a write to the UARTLCRH register. Reset type: PER.RESET

48.5.2.7 UARTLCRH Register (Offset = 2Ch) [Reset = 0h]

UARTLCRH is shown in Figure 48-10 and described in Table 48-10.

Return to the Summary Table.

The UARTLCRH register is the line control register. Serial parameters such as data length, parity, and stop bit selection are implemented in this register.
When updating the baud-rate divisor (UARTIBRD and/or UARTIFRD), the UARTLCRH register must also be written. The write strobe for the baud-rate divisor registers is tied to the UARTLCRH register.

Figure 48-10. UARTLCRH Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
SPS	WLEN		FEN	STP2	EPS	PEN	BRK
R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 48-10. UARTLCRH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	SPS	R/W	0h	UART Stick Parity Select UART Stick Parity Select 0 Stick parity is disabled (default) 1 Stick parity is enabled. When bits 1, 2, and 7 of UARTLCRH are set, the parity bit is transmitted and checked as a 0. When bits 1 and 7 are set and 2 is cleared, the parity bit is transmitted and checked as a 1. Reset type: PER.RESET
6-5	WLEN	R/W	0h	UART Word Length The bits indicate the number of data bits transmitted or received in a frame as follows: 0x0 5 bits (default) 0x1 6 bits 0x2 7 bits 0x3 8 bits Reset type: PER.RESET
4	FEN	R/W	0h	UART Enable FIFOs 0 The FIFOs are disabled. The FIFOs become 1-byte-deep holding registers. 1 The transmit and receive FIFO buffers are enabled (FIFO mode). Reset type: PER.RESET
3	STP2	R/W	0h	UART Two Stop Bits Select 0 One stop bit is transmitted at the end of a frame. 1 Two stop bits are transmitted at the end of a frame. The receive logic does not check for two stop bits being received. Reset type: PER.RESET

Table 48-10. UARTLCRH Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	EPS	R/W	0h	UART Even Parity Select 0 Odd parity is performed, which checks for an odd number of 1s. 1 Even parity generation and checking is performed during transmission and reception, which checks for an even number of 1s in data and parity bits. This bit has no effect when parity is disabled by the PEN bit. Reset type: PER.RESET
1	PEN	R/W	0h	UART Parity Enable 0 Parity is disabled and no parity bit is added to the data frame. 1 Parity checking and generation is enabled. Reset type: PER.RESET
0	BRK	R/W	0h	UART Send Break 0 Normal use. 1 A Low level is continually output on the UnTx signal, after completing transmission of the current character. For the proper execution of the break command, software must set this bit for at least two frames (character periods). Reset type: PER.RESET

48.5.2.8 UARTCTL Register (Offset = 30h) [Reset = 300h]

UARTCTL is shown in Figure 48-11 and described in Table 48-11.

Return to the Summary Table.

The UARTCTL register is the control register. All the bits are cleared on reset except for the Transmit Enable (TXE) and Receive Enable (RXE) bits, which are set.
To enable the UART module, the UARTEN bit must be set. If software requires a configuration change in the module, the UARTEN bit must be cleared before the configuration changes are written. If the UART is disabled during a transmit or receive operation, the current transaction is completed prior to the UART stopping.

Figure 48-11. UARTCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED		RESERVED	RESERVED	RXE	TXE
R/W-0h	R/W-0h	R-0h		R/W-0h	R/W-0h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
LBE	RESERVED	HSE	EOT	RESERVED	SIRLP	SIREN	UARTEN
R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 48-11. UARTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13-12	RESERVED	R	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RXE	R/W	1h	UART Receive Enable 0 The receive section of the UART is disabled. 1 The receive section of the UART is enabled. If the UART is disabled in the middle of a receive, it completes the current character before stopping.To enable reception, the UARTEN bit must also be set. Reset type: PER.RESET
8	TXE	R/W	1h	UART Transmit Enable 0 The transmit section of the UART is disabled. 1 The transmit section of the UART is enabled. If the UART is disabled in the middle of a transmission, it completes the current character before stopping. To enable transmission, the UARTEN bit must also be set. Reset type: PER.RESET
7	LBE	R/W	0h	UART Loop Back Enable 0 Normal operation. 1 The UnTx path is fed through the UnRx path. Reset type: PER.RESET
6	RESERVED	R	0h	Reserved

Table 48-11. UARTCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	HSE	R/W	0h	High-Speed Enable 0 The UART is clocked using the system clock divided by 16. 1 The UART is clocked using the system clock divided by 8. Reset type: PER.RESET
4	EOT	R/W	0h	End of Transmission This bit determines the behavior of the TXRIS bit in the UARTRIS register. 0 The TXRIS bit is set when the transmit FIFO condition specified in UARTIFLS is met. 1 The TXRIS bit is set only after all transmitted data, including stop bits, have cleared the serializer. Reset type: PER.RESET
3	RESERVED	R/W	0h	Reserved
2	SIRLP	R/W	0h	UART SIR Low-Power Mode This bit selects the IrDA encoding mode. 0 Low-level bits are transmitted as an active High pulse with a width of 3/16th of the bit period. 1 The UART operates in SIR Low-Power mode. Low-level bits are transmitted with a pulse width which is 3 times the period of the IrLPBaud16 input signal, regardless of the selected bit rate. Setting this bit uses less power, but might reduce transmission distances. Reset type: PER.RESET
1	SIREN	R/W	0h	UART SIR Enable 0 Normal operation. 1 The IrDA SIR block is enabled, and the UART will transmit and receive data using SIR protocol. Reset type: PER.RESET
0	UARTEN	R/W	0h	UART Enable 0 The UART is disabled. 1 The UART is enabled. If the UART is disabled in the middle of transmission or reception, it completes the current character before stopping. Reset type: PER.RESET

48.5.2.9 UARTIFLS Register (Offset = 34h) [Reset = 12h]

UARTIFLS is shown in Figure 48-12 and described in Table 48-12.

Return to the Summary Table.

The UARTIFLS register is the interrupt FIFO level select register. You can use this register to define the FIFO level at which the TXRIS and RXRIS bits in the UARTRIS register are triggered.
The interrupts are generated based on a transition through a level rather than being based on the level. That is, the interrupts are generated when the fill level progresses through the trigger level.
For example, if the receive trigger level is set to the half-way mark, the interrupt is triggered as the module is receiving the 9th character.
Therefore, changing the trigger level does not trigger an interrupt using the new level until another character is received.
Out of reset, the TXIFLSEL and RXIFLSEL bits are configured so that the FIFOs trigger an interrupt at the half-way mark.

Figure 48-12. UARTIFLS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED										RXIFLSEL			TXIFLSEL		
R-0h										R/W-2h			R/W-2h		

Table 48-12. UARTIFLS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5-3	RXIFLSEL	R/W	2h	UART Receive Interrupt FIFO Level Select The trigger points for the receive interrupt are as follows: Value Description 0x0 RX FIFO greater than or equal to 1/8 full 0x1 RX FIFO greater than or equal to 1/4 full 0x2 RX FIFO greater than or equal to 1/2 full (default) 0x3 RX FIFO greater than or equal to 3/4 full 0x4 RX FIFO greater than or equal to 7/8 full 0x5-0x7 Reserved Reset type: PER.RESET
2-0	TXIFLSEL	R/W	2h	Value Description 0x0 TX FIFO less than or equal 7/8 empty 0x1 TX FIFO less than or equal 3/4 empty 0x2 TX FIFO less than or equal 1/2 empty (default) 0x3 TX FIFO less than or equal 1/4 empty 0x4 TX FIFO less than or equal 1/8 empty 0x5-0x7 Reserved Note: If the EOT bit in UARTCTL is set, the transmit interrupt is generated once the FIFO is completely empty and all data including stop bits have left the transmit serializer. In this case, the setting of TXIFLSEL is ignored. Reset type: PER.RESET

48.5.2.10 UARTIM Register (Offset = 38h) [Reset = 0h]

UARTIM is shown in Figure 48-13 and described in Table 48-13.

Return to the Summary Table.

The UARTIM register is the interrupt mask set/clear register.
On a read, this register gives the current value of the mask on the relevant interrupt. Setting a bit allows the corresponding raw interrupt signal to be routed to the interrupt controller. Clearing a bit prevents the raw interrupt signal from being sent to the interrupt controller.

Figure 48-13. UARTIM Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						DMATXIM	DMARXIM
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED			9BITIM	RESERVED	OEIM	BEIM	PEIM
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
FEIM	RTIM	TXIM	RXIM	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 48-13. UARTIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17	DMATXIM	R/W	0h	Transmit DMA Interrupt Mask 0 The DMATXRIS interrupt is suppressed and not sent to the interrupt controller. 1 An interrupt is sent to the interrupt controller when the DMATXRIS bit in the UARTRIS register is set. Reset type: PER.RESET
16	DMARXIM	R/W	0h	Receive DMA Interrupt Mask 0 The DMARXRIS interrupt is suppressed and not sent to the interrupt controller. 1 An interrupt is sent to the interrupt controller when the DMARXRIS bit in the UARTRIS register is set. Reset type: PER.RESET
15-13	RESERVED	R	0h	Reserved
12	9BITIM	R/W	0h	9-Bit Mode Interrupt Mask 0 The 9BITRIS interrupt is suppressed and not sent to the interrupt controller. 1 An interrupt is sent to the interrupt controller when the 9BITRIS bit in the UARTRIS register is set. Reset type: PER.RESET
11	RESERVED	R/W	0h	Reserved
10	OEIM	R/W	0h	UART Overrun Error Interrupt Mask 0 The OERIS interrupt is suppressed and not sent to the interrupt controller. 1 An interrupt is sent to the interrupt controller when the OERIS bit in the UARTRIS register is set. Reset type: PER.RESET

Table 48-13. UARTIM Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	BEIM	R/W	0h	UART Break Error Interrupt Mask 0 The BERIS interrupt is suppressed and not sent to the interrupt controller. 1 An interrupt is sent to the interrupt controller when the BERIS bit in the UARTRIS register is set. Reset type: PER.RESET
8	PEIM	R/W	0h	UART Parity Error Interrupt Mask 0 The PERIS interrupt is suppressed and not sent to the interrupt controller. 1 An interrupt is sent to the interrupt controller when the PERIS bit in the UARTRIS register is set. Reset type: PER.RESET
7	FEIM	R/W	0h	UART Framing Error Interrupt Mask 0 The FERIS interrupt is suppressed and not sent to the interrupt controller. 1 An interrupt is sent to the interrupt controller when the FERIS bit in the UARTRIS register is set. Reset type: PER.RESET
6	RTIM	R/W	0h	UART Receive Time-Out Interrupt Mask 0 The RTRIS interrupt is suppressed and not sent to the interrupt controller. 1 An interrupt is sent to the interrupt controller when the RTRIS bit in the UARTRIS register is set. Reset type: PER.RESET
5	TXIM	R/W	0h	UART Transmit Interrupt Mask 0 The TXRIS interrupt is suppressed and not sent to the interrupt controller. 1 An interrupt is sent to the interrupt controller when the TXRIS bit in the UARTRIS register is set. Reset type: PER.RESET
4	RXIM	R/W	0h	UART Receive Interrupt Mask 0 The RXRIS interrupt is suppressed and not sent to the interrupt controller. 1 An interrupt is sent to the interrupt controller when the RXRIS bit in the UARTRIS register is set. Reset type: PER.RESET
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	RESERVED	R/W	0h	Reserved

48.5.2.11 UARTRIS Register (Offset = 3Ch) [Reset = 0h]

UARTRIS is shown in Figure 48-14 and described in Table 48-14.

Return to the Summary Table.

The UARTRIS register is the raw interrupt status register. On a read, this register gives the current raw status value of the corresponding interrupt. A write has no effect.

Figure 48-14. UARTRIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						DMATXRIS	DMARXRIS
R-0h						R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED			9BITRIS	RESERVED	OERIS	BERIS	PERIS
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
FERIS	RTRIS	TXRIS	RXRIS	RESERVED	RESERVED	RESERVED	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 48-14. UARTRIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17	DMATXRIS	R	0h	Transmit DMA Raw Interrupt Status 0 No interrupt 1 The transmit DMA has completed. This bit is cleared by writing a 1 to the DMATXIC bit in the UARTICR register. Reset type: PER.RESET
16	DMARXRIS	R	0h	Receive DMA Raw Interrupt Status 0 No interrupt 1 The receive DMA has completed. This bit is cleared by writing a 1 to the DMARXIC bit in the UARTICR register. Reset type: PER.RESET
15-13	RESERVED	R	0h	Reserved
12	9BITRIS	R	0h	9-Bit Mode Raw Interrupt Status 0 No interrupt 1 A receive address match has occurred. This bit is cleared by writing a 1 to the 9BITIC bit in the UARTICR register. Reset type: PER.RESET
11	RESERVED	R	0h	Reserved
10	OERIS	R	0h	UART Overrun Error Raw Interrupt Status 0 No interrupt 1 An overrun error has occurred. This bit is cleared by writing a 1 to the OEIC bit in the UARTICR register. Reset type: PER.RESET

Table 48-14. UARTRIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	BERIS	R	0h	UART Break Error Raw Interrupt Status 0 No interrupt 1 A break error has occurred. This bit is cleared by writing a 1 to the BEIC bit in the UARTICR register. Reset type: PER.RESET
8	PERIS	R	0h	UART Parity Error Raw Interrupt Status 0 No interrupt 1 A parity error has occurred. This bit is cleared by writing a 1 to the PEIC bit in the UARTICR register. Reset type: PER.RESET
7	FERIS	R	0h	UART Framing Error Raw Interrupt Status 0 No interrupt 1 A framing error has occurred. This bit is cleared by writing a 1 to the FEIC bit in the UARTICR register. Reset type: PER.RESET
6	RTRIS	R	0h	UART Receive Time-Out Raw Interrupt Status 0 No interrupt 1 A receive time out has occurred. This bit is cleared by writing a 1 to the RTIC bit in the UARTICR register. Reset type: PER.RESET
5	TXRIS	R	0h	UART Transmit Raw Interrupt Status 0 No interrupt 1 If the EOT bit in the UARTCTL register is clear, the transmit FIFO level has passed through the condition defined in the UARTIFLS register.If the EOT bit is set, the last bit of all transmitted data and flags has left the serializer. This bit is cleared by writing a 1 to the TXIC bit in the UARTICR register or by writing data to the transmit FIFO until it becomes greater than the trigger level, if the FIFO is enabled, or by writing a single byte if the FIFO is disabled. Reset type: PER.RESET
4	RXRIS	R	0h	UART Receive Raw Interrupt Status 0 No interrupt 1 The receive FIFO level has passed through the condition defined in the UARTIFLS register. This bit is cleared by writing a 1 to the RXIC bit in the UARTICR register or by reading data from the receive FIFO until it becomes less than the trigger level, if the FIFO is enabled, or by reading a single byte if the FIFO is disabled. Reset type: PER.RESET
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

48.5.2.12 UARTMIS Register (Offset = 40h) [Reset = 0h]

UARTMIS is shown in Figure 48-15 and described in Table 48-15.

Return to the Summary Table.

The UARTMIS register is the masked interrupt status register. On a read, this register gives the current masked status value of the corresponding interrupt. A write has no effect.

Figure 48-15. UARTMIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						DMATXMIS	DMARXMIS
R-0h						R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED			9BITMIS	RESERVED	OEMIS	BEMIS	PEMIS
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
FEMIS	RTMIS	TXMIS	RXMIS	RESERVED	RESERVED	RESERVED	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 48-15. UARTMIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17	DMATXMIS	R	0h	Transmit DMA Masked Interrupt Status 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to the completion of the transmit DMA. This bit is cleared by writing a 1 to the DMATXIC bit in the UARTICR register. Reset type: PER.RESET
16	DMARXMIS	R	0h	Receive DMA Masked Interrupt Status 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to the completion of the receive DMA. This bit is cleared by writing a 1 to the DMARXIC bit in the UARTICR register. Reset type: PER.RESET
15-13	RESERVED	R	0h	Reserved
12	9BITMIS	R	0h	9-Bit Mode Masked Interrupt Status 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to a receive address match. This bit is cleared by writing a 1 to the 9BITIC bit in the UARTICR register. Reset type: PER.RESET
11	RESERVED	R	0h	Reserved
10	OEMIS	R	0h	UART Overrun Error Masked Interrupt Status 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to an overrun error. This bit is cleared by writing a 1 to the OEIC bit in the UARTICR register. Reset type: PER.RESET

Table 48-15. UARTMIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	BEMIS	R	0h	UART Break Error Masked Interrupt Status 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to a break error. This bit is cleared by writing a 1 to the BEIC bit in the UARTICR register. Reset type: PER.RESET
8	PEMIS	R	0h	UART Parity Error Masked Interrupt Status 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to a parity error. This bit is cleared by writing a 1 to the PEIC bit in the UARTICR register. Reset type: PER.RESET
7	FEMIS	R	0h	UART Framing Error Masked Interrupt Status 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to a framing error. This bit is cleared by writing a 1 to the FEIC bit in the UARTICR register. Reset type: PER.RESET
6	RTMIS	R	0h	UART Receive Time-Out Masked Interrupt Status 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to a receive time out. This bit is cleared by writing a 1 to the RTIC bit in the UARTICR register. Reset type: PER.RESET
5	TXMIS	R	0h	UART Transmit Masked Interrupt Status 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to passing through the specified transmit FIFO level (if the EOT bit is clear) or due to the transmission of the last data bit (if the EOT bit is set). This bit is cleared by writing a 1 to the TXIC bit in the UARTICR register or by writing data to the transmit FIFO until it becomes greater than the trigger level, if the FIFO is enabled, or by writing a single byte if the FIFO is disabled. Reset type: PER.RESET
4	RXMIS	R	0h	UART Receive Masked Interrupt Status 0 An interrupt has not occurred or is masked. 1 An unmasked interrupt was signaled due to passing through the specified receive FIFO level. This bit is cleared by writing a 1 to the RXIC bit in the UARTICR register or by reading data from the receive FIFO until it becomes less than the trigger level, if the FIFO is enabled, or by reading a single byte if the FIFO is disabled. Reset type: PER.RESET
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

48.5.2.13 UARTICR Register (Offset = 44h) [Reset = 0h]

UARTICR is shown in Figure 48-16 and described in Table 48-16.

Return to the Summary Table.

The UARTICR register is the interrupt clear register. On a write of 1, the corresponding interrupt (both raw interrupt and masked interrupt, if enabled) is cleared. A write of 0 has no effect.

Figure 48-16. UARTICR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						DMATXIC	DMARXIC
R-0h						R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
RESERVED			9BITIC	EOTIC	OEIC	BEIC	PEIC
R-0h			R/W-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
FEIC	RTIC	TXIC	RXIC	RESERVED	RESERVED	RESERVED	RESERVED
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 48-16. UARTICR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17	DMATXIC	R-0/W1S	0h	Transmit DMA Interrupt Clear Writing a 1 to this bit clears the DMATXRIS bit in the UARTRIS register and the DMATXMIS bit in the UARTMIS register. Reset type: PER.RESET
16	DMARXIC	R-0/W1S	0h	Receive DMA Interrupt Clear Writing a 1 to this bit clears the DMARXRIS bit in the UARTRIS register and the DMARXMIS bit in the UARTMIS register. Reset type: PER.RESET
15-13	RESERVED	R	0h	Reserved
12	9BITIC	R/W	0h	9-Bit Mode Interrupt Clear Writing a 1 to this bit clears the 9BITRIS bit in the UARTRIS register and the 9BITMIS bit in the UARTMIS register. Reset type: PER.RESET
11	EOTIC	R-0/W1S	0h	End of Transmission Interrupt Clear Writing a 1 to this bit clears the EOTRIS bit in the UARTRIS register and the EOTMIS bit in the UARTMIS register. Reset type: PER.RESET
10	OEIC	R-0/W1S	0h	Overrun Error Interrupt Clear Writing a 1 to this bit clears the OERIS bit in the UARTRIS register and the OEMIS bit in the UARTMIS register. Reset type: PER.RESET
9	BEIC	R-0/W1S	0h	Break Error Interrupt Clear Writing a 1 to this bit clears the BERIS bit in the UARTRIS register and the BEMIS bit in the UARTMIS register. Reset type: PER.RESET
8	PEIC	R-0/W1S	0h	Parity Error Interrupt Clear Writing a 1 to this bit clears the PERIS bit in the UARTRIS register and the PEMIS bit in the UARTMIS register. Reset type: PER.RESET

Table 48-16. UARTICR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	FEIC	R-0/W1S	0h	Framing Error Interrupt Clear Writing a 1 to this bit clears the FERIS bit in the UARTRIS register and the FEMIS bit in the UARTMIS register. Reset type: PER.RESET
6	RTIC	R-0/W1S	0h	Receive Time-Out Interrupt Clear Writing a 1 to this bit clears the RTRIS bit in the UARTRIS register and the RTMIS bit in the UARTMIS register. Reset type: PER.RESET
5	TXIC	R-0/W1S	0h	Transmit Interrupt Clear Writing a 1 to this bit clears the TXRIS bit in the UARTRIS register and the TXMIS bit in the UARTMIS register. Reset type: PER.RESET
4	RXIC	R-0/W1S	0h	Receive Interrupt Clear Writing a 1 to this bit clears the RXRIS bit in the UARTRIS register and the RXMIS bit in the UARTMIS register. Reset type: PER.RESET
3	RESERVED	R-0/W1S	0h	Reserved
2	RESERVED	R-0/W1S	0h	Reserved
1	RESERVED	R-0/W1S	0h	Reserved
0	RESERVED	R-0/W1S	0h	Reserved

48.5.2.14 UARTDMACTL Register (Offset = 48h) [Reset = 0h]

UARTDMACTL is shown in Figure 48-17 and described in Table 48-17.

Return to the Summary Table.

UART DMA Control

Figure 48-17. UARTDMACTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					DMAERR	TXDMAE	RXDMAE
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 48-17. UARTDMACTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2	DMAERR	R/W	0h	DMA on Error 0 DMA receive requests are unaffected when a receive error occurs. 1 DMA receive requests are automatically disabled when a receive error occurs. Reset type: PER.RESET
1	TXDMAE	R/W	0h	Transmit DMA Enable 0 DMA for the transmit FIFO is disabled. 1 DMA for the transmit FIFO is enabled. Reset type: PER.RESET
0	RXDMAE	R/W	0h	Receive DMA Enable 0 DMA for the receive FIFO is disabled. 1 DMA for the receive FIFO is enabled. Reset type: PER.RESET

48.5.2.15 UART9BITADDR Register (Offset = A4h) [Reset = 0h]

UART9BITADDR is shown in Figure 48-18 and described in Table 48-18.

Return to the Summary Table.

The UART9BITADDR register is used to write the specific address that should be matched with the receiving byte when the 9-bit Address Mask (UART9BITAMASK) is set to 0xFF. This register is used in conjunction with UART9BITAMASK to form a match for address-byte received.

Figure 48-18. UART9BITADDR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
9BITEN	RESERVED						
R/W-0h	R-0h						
7	6	5	4	3	2	1	0
ADDR							
R/W-0h							

Table 48-18. UART9BITADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	9BITEN	R/W	0h	Enable 9-Bit Mode 0 9-bit mode is disabled. 1 9-bit mode is enabled. Reset type: PER.RESET
14-8	RESERVED	R	0h	Reserved
7-0	ADDR	R/W	0h	Self Address for 9-Bit Mode This field contains the address that should be matched when UART9BITAMASK is 0xFF. Reset type: PER.RESET

48.5.2.16 UART9BITAMASK Register (Offset = A8h) [Reset = FFh]

UART9BITAMASK is shown in Figure 48-19 and described in Table 48-19.

Return to the Summary Table.

The UART9BITAMASK register is used to enable the address mask for 9-bit mode. The address bits are masked to create a set of addresses to be matched with the received address byte.

Figure 48-19. UART9BITAMASK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								MASK							
R-0h																								R/W-FFh							

Table 48-19. UART9BITAMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	MASK	R/W	FFh	Self Address Mask for 9-Bit Mode This field contains the address mask that creates a set of addresses that should be matched. Reset type: PER.RESET

48.5.2.17 UARTPP Register (Offset = FC0h) [Reset = 2h]

UARTPP is shown in Figure 48-20 and described in Table 48-20.

Return to the Summary Table.

The UARTPP register provides information regarding the properties of the UART module.

Figure 48-20. UARTPP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												MSE	MS	NB	SC
R-0h												R-0h	R-0h	R-1h	R-0h

Table 48-20. UARTPP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	MSE	R	0h	Modem Support Extended 0 The UART module does not provide extended support for modem control. 1 The UART module provides extended support for modem control including UARTnDTR, UARTnDSR, UARTnDCD, and UARTnRI. Reset type: PER.RESET
2	MS	R	0h	Modem Support 0 The UART module does not provide support for modem control. 1 The UART module provides support for modem control including UARTnRTS and UARTnCTS. Reset type: PER.RESET
1	NB	R	1h	9-Bit Support 0 The UART module does not provide support for the transmission of 9-bit data for RS-485 support. 1 The UART module provides support for the transmission of 9-bit data for RS-485 support. Reset type: PER.RESET
0	SC	R	0h	Smart Card Support 0 The UART module does not provide smart card support. 1 The UART module provides smart card support. Reset type: PER.RESET

48.5.2.18 UARTPeriphID4 Register (Offset = FD0h) [Reset = 60h]

UARTPeriphID4 is shown in Figure 48-21 and described in Table 48-21.

Return to the Summary Table.

UART Peripheral Identification 4

Figure 48-21. UARTPeriphID4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID4							
R-0h																								R-60h							

Table 48-21. UARTPeriphID4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID4	R	60h	UART Peripheral ID Register [7:0] Can be used by software to identify the presence of this peripheral. Reset type: PER.RESET

48.5.2.19 UARTPeriphID5 Register (Offset = FD4h) [Reset = 0h]

UARTPeriphID5 is shown in Figure 48-22 and described in Table 48-22.

Return to the Summary Table.

UART Peripheral Identification 5

Figure 48-22. UARTPeriphID5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID5							
R-0h																								R-0h							

Table 48-22. UARTPeriphID5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID5	R	0h	UART Peripheral ID Register [15:8] Can be used by software to identify the presence of this peripheral. Reset type: PER.RESET

48.5.2.20 UARTPeriphID6 Register (Offset = FD8h) [Reset = 0h]

UARTPeriphID6 is shown in Figure 48-23 and described in Table 48-23.

Return to the Summary Table.

UART Peripheral Identification 6

Figure 48-23. UARTPeriphID6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID6							
R-0h																								R-0h							

Table 48-23. UARTPeriphID6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID6	R	0h	UART Peripheral ID Register [23:16] Can be used by software to identify the presence of this peripheral. Reset type: PER.RESET

48.5.2.21 UARTPeriphID7 Register (Offset = FDCh) [Reset = 0h]

UARTPeriphID7 is shown in Figure 48-24 and described in Table 48-24.

Return to the Summary Table.

UART Peripheral Identification 7

Figure 48-24. UARTPeriphID7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID7							
R-0h																								R-0h							

Table 48-24. UARTPeriphID7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID7	R	0h	UART Peripheral ID Register [31:24] Can be used by software to identify the presence of this peripheral. Reset type: PER.RESET

48.5.2.22 UARTPeriphID0 Register (Offset = FE0h) [Reset = 11h]

UARTPeriphID0 is shown in Figure 48-25 and described in Table 48-25.

Return to the Summary Table.

UART Peripheral Identification 0

Figure 48-25. UARTPeriphID0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID0							
R-0h																								R-11h							

Table 48-25. UARTPeriphID0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID0	R	11h	UART Peripheral ID Register [7:0] Can be used by software to identify the presence of this peripheral. Reset type: PER.RESET

48.5.2.23 UARTPeriphID1 Register (Offset = FE4h) [Reset = 0h]

UARTPeriphID1 is shown in Figure 48-26 and described in Table 48-26.

Return to the Summary Table.

UART Peripheral Identification 1

Figure 48-26. UARTPeriphID1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID1							
R-0h																								R-0h							

Table 48-26. UARTPeriphID1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID1	R	0h	UART Peripheral ID Register [15:8] Can be used by software to identify the presence of this peripheral. Reset type: PER.RESET

48.5.2.24 UARTPeriphID2 Register (Offset = FE8h) [Reset = 18h]

UARTPeriphID2 is shown in Figure 48-27 and described in Table 48-27.

Return to the Summary Table.

UART Peripheral Identification 2

Figure 48-27. UARTPeriphID2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID2							
R-0h																								R-18h							

Table 48-27. UARTPeriphID2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID2	R	18h	UART Peripheral ID Register [23:16] Can be used by software to identify the presence of this peripheral. Reset type: PER.RESET

48.5.2.25 UARTPeriphID3 Register (Offset = FECh) [Reset = 1h]

UARTPeriphID3 is shown in Figure 48-28 and described in Table 48-28.

Return to the Summary Table.

UART Peripheral Identification 3

Figure 48-28. UARTPeriphID3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID3							
R-0h																								R-1h							

Table 48-28. UARTPeriphID3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID3	R	1h	UART Peripheral ID Register [31:24] Can be used by software to identify the presence of this peripheral. Reset type: PER.RESET

48.5.2.26 UARTPCellID0 Register (Offset = FF0h) [Reset = Dh]

UARTPCellID0 is shown in Figure 48-29 and described in Table 48-29.

Return to the Summary Table.

UART PrimeCell Identification 0

Figure 48-29. UARTPCellID0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CID0							
R-0h																								R-Dh							

Table 48-29. UARTPCellID0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	CID0	R	Dh	UART PrimeCell ID Register [7:0] Provides software a standard cross-peripheral identification system. Reset type: PER.RESET

48.5.2.27 UARTPCellID1 Register (Offset = FF4h) [Reset = F0h]

UARTPCellID1 is shown in Figure 48-30 and described in Table 48-30.

Return to the Summary Table.

UART PrimeCell Identification 1

Figure 48-30. UARTPCellID1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CID1							
R-0h																								R-F0h							

Table 48-30. UARTPCellID1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	CID1	R	F0h	UART PrimeCell ID Register [15:8] Provides software a standard cross-peripheral identification system. Reset type: PER.RESET

48.5.2.28 UARTPCellID2 Register (Offset = FF8h) [Reset = 5h]

UARTPCellID2 is shown in Figure 48-31 and described in Table 48-31.

Return to the Summary Table.

UART PrimeCell Identification 2

Figure 48-31. UARTPCellID2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CID2							
R-0h																								R-5h							

Table 48-31. UARTPCellID2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	CID2	R	5h	UART PrimeCell ID Register [23:16] Provides software a standard cross-peripheral identification system. Reset type: PER.RESET

48.5.2.29 UARTPCellID3 Register (Offset = FFCh) [Reset = B1h]

UARTPCellID3 is shown in Figure 48-32 and described in Table 48-32.

Return to the Summary Table.

UART PrimeCell Identification 3

Figure 48-32. UARTPCellID3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CID3							
R-0h																								R-B1h							

Table 48-32. UARTPCellID3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	CID3	R	B1h	UART PrimeCell ID Register [31:24] Provides software a standard cross-peripheral identification system. Reset type: PER.RESET

48.5.3 UART_REGS_WRITE Registers

Table 48-33 lists the memory-mapped registers for the UART_REGS_WRITE registers. All register offset addresses not listed in Table 48-33 should be considered as reserved locations and the register contents should not be modified.

Table 48-33. UART_REGS_WRITE Registers

Offset	Acronym	Register Name	Write Protection	Section
4h	UARTECR	UART Error Clear		Go

Complex bit access types are encoded to fit into small table cells. Table 48-34 shows the codes that are used for access types in this section.

Table 48-34. UART_REGS_WRITE Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

48.5.3.1 UARTECR Register (Offset = 4h) [Reset = 0h]

UARTECR is shown in Figure 48-33 and described in Table 48-35.

Return to the Summary Table.

The UARTECR register is the error clear register.
In addition to the UARTDR register, receive status can also be read from the UARTRSR register. If the status is read from this register, then the status information corresponds to the entry read from UARTDR prior to reading UARTRSR. The status information for overrun is set immediately when an overrun condition occurs.
The UARTRSR register cannot be written.
A write of any value to the UARTECR register clears the framing, parity, break, and overrun errors.
All the bits are cleared on reset.

Figure 48-33. UARTECR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								DATA							
R-0/W-0h																								R-0/W-0h							

Table 48-35. UARTECR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R-0/W	0h	Reserved
7-0	DATA	R-0/W	0h	Error Clear A write to this register of any data clears the framing, parity, break, and overrun flags. Reset type: PER.RESET

This page intentionally left blank.

Chapter 49

Micro Direct Memory Access (µDMA)

The µDMA controller provides a way to offload data transfer tasks from the Arm® Cortex®-M4 processor, allowing for more efficient use of the processor and the available bus bandwidth. The µDMA controller can perform transfers between memory and peripherals. It has dedicated channels for each supported on-chip module and can be programmed to automatically perform transfers between peripherals and memory when the peripheral is ready to transfer more data.

49.1 Introduction

49.1.1 Features

The µDMA controller provides the following features:

- Arm® PrimeCell 32-channel configurable µDMA controller
- Support for memory-to-memory, memory-to-peripheral, and peripheral-to-memory in multiple transfer modes:
  - Basic mode
  - Ping-pong mode
  - Memory scatter-gather mode
  - Peripheral scatter-gather mode
  - Auto request mode
- Highly flexible and configurable channel operation:
  - Independently configured and operated channels
  - Dedicated channels for supported on-chip modules
  - Flexible channel assignments
  - One channel each for receive and transmit path for bidirectional modules
  - Dedicated channel for software-initiated transfers
  - Per-channel configurable priority scheme
  - Optional software-initiated requests for any channel
- Two levels of priority
- Data sizes of 8, 16, and 32 bits
- Programmable transfer size in binary steps from 1 to 1024
- Source and destination address increment size of byte, halfword, word, or no increment
- Maskable peripheral requests
- Supports two interrupts:
  - µDMA Software interrupt: µDMA generates an interrupt when a software channel completes all its transfers
  - µDMA Error interrupt: µDMA generates an interrupt an when error is detected on a DMA transfer
- DMA transfers triggered by a peripheral event generate a corresponding peripheral interrupt when the DMA has completes all transfers.

49.1.2 Block Diagram

Figure 49-1 shows the µDMA block diagram.

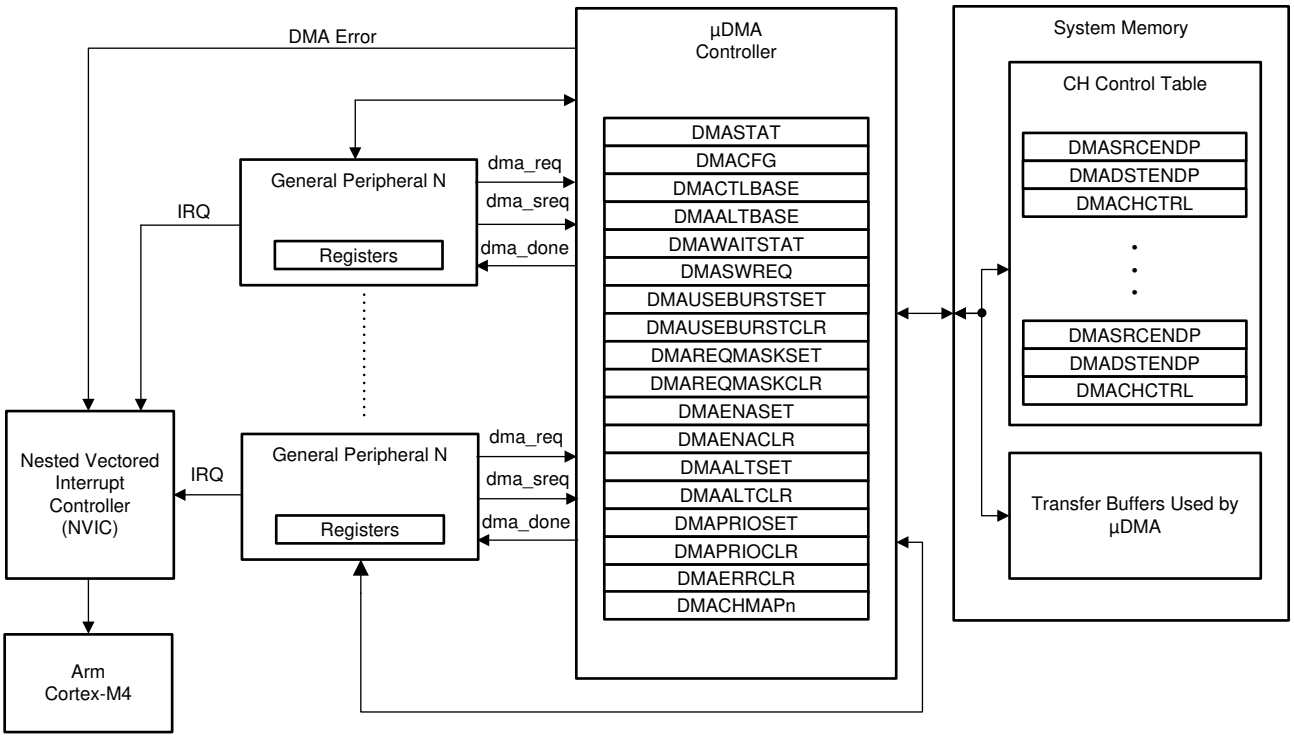


Figure 49-1. µDMA Block Diagram

49.2 Functional Description

The µDMA controller is a flexible and highly configurable DMA controller designed to work efficiently with the Cortex®-M4 processor core of the microcontroller. The µDMA controller supports multiple data sizes and address increment schemes, multiple levels of priority among DMA channels, and several transfer modes to allow for sophisticated programmed data transfers. Bus use by the µDMA controller is always subordinate to the processor core, so it never delays a bus transaction by the processor. Because the µDMA controller is only using otherwise idle bus cycles, the data transfer bandwidth it provides is essentially free, with no affect on the rest of the system. The bus architecture has been optimized to greatly enhance the ability of the processor core and the µDMA controller to efficiently share the on-chip bus, thus improving performance. The optimizations include RAM striping and peripheral bus segmentation, which in many cases let both the processor core and the µDMA controller access the bus and perform simultaneous data transfers.

Each peripheral function that is supported has a dedicated channel on the µDMA controller that can be configured independently. The µDMA controller implements a unique configuration method using channel control structures that are maintained in system memory by the processor. While simple transfer modes are supported, it is also possible to build up sophisticated "task" lists in memory that allow the µDMA controller to perform arbitrary-sized transfers to and from arbitrary locations as part of a single transfer request. The µDMA controller also supports the use of ping-pong buffering to accommodate constant streaming of data to or from a peripheral.

Each channel also has a configurable arbitration size. The arbitration size is the number of items that are transferred in a burst before the µDMA controller rearbitrates for channel priority. Using the arbitration size, it is possible to control exactly how many items are transferred to or from a peripheral each time it makes a µDMA service request.

49.2.1 Channel Assignments

µDMA channels 0-31 are assigned to different peripherals according to Table 49-1. The DMA Channel Map (DMACHMAPx) registers can be used to specify the first, second, or third channel mapping assignment.

Note

Channels noted in the table as "Available for software" may be assigned to peripherals in the future. However, they are currently available for software use. Channel 30 is dedicated for software use.

The USB endpoints mapped to µDMA channels 0-3 can be changed with the USBDMASEL register.

Because of the way the µDMA controller interacts with peripherals, the µDMA channel for the peripheral must be enabled in order for the µDMA controller to be able to read and write the peripheral registers, even if a different µDMA channel is used to perform the µDMA transfer. To minimize confusion and chance of software errors, it is best practice to use a peripheral's µDMA channel for performing all µDMA transfers for that peripheral, even if it is processor-triggered and using AUTO mode, which could be considered a software transfer.

If the software channel is used, interrupts occur on the dedicated µDMA interrupt vector. If the peripheral channel is used, then the interrupt occurs on the interrupt vector for the peripheral.

Table 49-1. µDMA Channel Assignment Mapping

µDMA Channel	Primary Assignment	Single/Burst
0	USB_DMAA_Rx	B
1	USB_DMAA_TX	B
2	USB_DMAB_RX	B
3	USB_DMAB_TX	B
4	USB_DMAC_RX	B
5	USB_DMAC_TX	B
6	Reserved	
7	Reserved	
8	UART0_RX	SB
9	UART0_TX	SB
10	SSI0_RX	SB
11	SSI0_TX	SB
12	Available for software	B
13	Available for software	B
14	ECAT_SYNC_TRIG	B
15	Reserved	
16	Reserved	
17	Reserved	
18	Reserved	
19	Reserved	
20	Reserved	
21	AES_SDMAREQ_CTXIN_S	B
22	AES_SDMAREQ_DIN_S	B
23	AES_SDMAREQ_CTXOUT_S	B
24	AES_SDMAREQ_DOUT_S	B
25	Reserved	
26	Available for software	B
27	Available for software	B

Table 49-1. µDMA Channel Assignment Mapping (continued)

µDMA Channel	Primary Assignment	Single/Burst
28	I2C0_RX	SB
29	I2C0_TX	SB
30	Available for software	B
31	Reserved	

49.2.2 Priority

The µDMA controller assigns priority to each channel based on the channel number and the priority level bit for the channel. Channel number 0 has the highest priority and as the channel number increases, the priority of a channel decreases. Each channel has a priority level bit to provide two levels of priority: default priority and high priority. If the priority level bit is set, then that channel has higher priority than all other channels at default priority. If multiple channels are set for high priority, then the channel number is used to determine relative priority among all the high-priority channels.

The priority bit for a channel can be set using the DMA Channel Priority Set (DMAPRIOSET) register and cleared with the DMA Channel Priority Clear (DMAPRIOCLR) register.

49.2.3 Arbitration Size

When a µDMA channel requests a transfer, the µDMA controller arbitrates among all the channels making a request and services the µDMA channel with the highest priority. When a transfer begins, it continues for a selectable number of transfers before rearbitrating among the requesting channels again. The arbitration size can be configured for each channel, ranging from 1 to 1024 item transfers. After the µDMA controller transfers the number of items specified by the arbitration size, it then checks among all the channels making a request and services the channel with the highest priority.

If a lower priority µDMA channel uses a large arbitration size, the latency for higher priority channels is increased because the µDMA controller completes the lower priority burst before checking for higher priority requests. Therefore, lower priority channels should not use a large arbitration size for best response on high priority channels.

The arbitration size can also be thought of as a burst size. It is the maximum number of items that are transferred at any one time in a burst. Here, the term arbitration refers to determination of µDMA channel priority, not arbitration for the bus. When the µDMA controller arbitrates for the bus, the processor always takes priority. Furthermore, the µDMA controller is held off whenever the processor must perform a bus transaction on the same bus, even in the middle of a burst transfer.

49.2.4 Request Types

The µDMA controller responds to two types of requests from a peripheral: single or burst. Each peripheral may support either or both types of requests. A single request means that the peripheral is ready to transfer one item, while a burst request means that the peripheral is ready to transfer multiple items.

The µDMA controller responds differently depending on whether the peripheral is making a single request or a burst request. If both are asserted, and the µDMA channel has been set up for a burst transfer, then the burst request takes precedence. Table 49-2 shows how each peripheral supports the two request types.

Table 49-2. Request Type Support

Peripheral	Event That Generates Single Request	Event That Generates Burst Request
USB TX	None	FIFO TXRDY
USB RX	None	FIFO RXRDY
UART TX	TX FIFO not full	TX FIFO level (configurable)
UART RX	RX FIFO not empty	RX FIFO level (configurable)
SSI TX	TX FIFO not full	TX FIFO level (fixed at 4)
SSI RX	RX FIFO Not Empty	RX FIFO level (fixed at 4)
ECAT	None	
AES	None	Context in DMA request (AES0 Cin) Context out DMA request (AES0 Cout) Data in DMA request (AES0 Din) Data out DMA request (AES0 Dout)
I^2C TX	TX buffer not full	TX FIFO level (configurable)
I^2C RX	RX buffer not empty	RX FIFO level (configurable)

49.2.4.1 Single Request

When a single request is detected, and not a burst request, the µDMA controller transfers one item and then stops to wait for another request.

49.2.4.2 Burst Request

When a burst request is detected, the µDMA controller transfers the number of items that is the lesser of the arbitration size or the number of items remaining in the transfer. Therefore, the arbitration size should be the same as the number of data items that the peripheral can accommodate when making a burst request. For example, the UART generates a burst request based on the FIFO trigger level. In this case, the arbitration size should be set to the amount of data that the FIFO can transfer when the trigger level is reached. A burst transfer runs to completion once it is started, and cannot be interrupted, even by a higher priority channel. Burst transfers complete in a shorter time than the same number of non-burst transfers.

It may be desirable to use only burst transfers and not allow single transfers. For example, perhaps the nature of the data is such that it only makes sense when transferred together as a single unit rather than one piece at a time. The single request can be disabled by using the DMA Channel Useburst Set (DMAUSEBURSTSET) register. By setting the bit for a channel in this register, the µDMA controller only responds to burst requests for that channel.

49.2.5 Channel Configuration

The µDMA controller uses an area of system memory to store a set of channel control structures in a table. The control table may have one or two entries for each µDMA channel. Each entry in the table structure contains source and destination pointers, transfer size, and transfer mode. The control table can be located anywhere in system memory, but it must be contiguous and aligned on a 1024-byte boundary.

Table 49-3 lists the layout in memory of the channel control table. Each channel may have one or two control structures in the control table: a primary control structure and an optional alternate control structure. The table is organized so that all of the primary entries are in the first half of the table, and all the alternate structures are in the second half of the table. The primary entry is used for simple transfer modes where transfers can be reconfigured and restarted after each transfer is complete. In this case, the alternate control structures are not used and therefore only the first half of the table must be allocated in memory; the second half of the control table is not necessary, and that memory can be used for something else. If a more complex transfer mode is used such as ping-pong or scatter-gather, then the alternate control structure is also used and memory space should be allocated for the entire table.

Any unused memory in the control table may be used by the application. This includes the control structures for any channels that are unused by the application as well as the unused control word for each channel.

Table 49-3. Control Structure Memory Map

Offset	Channel
0x0	0, Primary
0x10	1, Primary
...	...
0x1F0	31, Primary

Table 49-4 summarizes an individual control structure entry in the control table. Each entry is aligned on a 16-byte boundary. The entry contains four long words: the source end pointer, the destination end pointer, the control word, and an unused entry. The end pointers point to the ending address of the transfer and are inclusive. If the source or destination is non-incrementing (as for a peripheral register), then the pointer should point to the transfer address.

Table 49-4. Channel Control Structure

Offset	Description
0x000	Source End Pointer
0x004	Destination End Pointer
0x008	Control Word
0x00C	Unused

The control word contains the following fields:

- Source and destination data sizes
- Source and destination address increment size
- Number of transfers before bus arbitration
- Total number of items to transfer
- Useburst flag
- Transfer mode

The control word and each field are described in detail in Section 49.3.5. The µDMA controller updates the transfer size and transfer mode fields as the transfer is performed. At the end of a transfer, the transfer size indicates 0, and the transfer mode indicates "stopped." Because the control word is modified by the µDMA controller, it must be reconfigured before each new transfer. The source and destination end pointers are not modified, so they can be left unchanged if the source or destination addresses remain the same.

Prior to starting a transfer, a µDMA channel must be enabled by setting the appropriate bit in the DMA Channel Enable Set (DMAENASET) register. A channel can be disabled by setting the channel bit in the DMA Channel Enable Clear (DMAENACLR) register. At the end of a complete µDMA transfer, the controller automatically disables the channel.

49.2.6 Transfer Modes

The µDMA controller supports several transfer modes. Two of the modes support simple one-time transfers. Several complex modes support a continuous flow of data.

49.2.6.1 Stop Mode

While Stop is not actually a transfer mode, it is a valid value for the mode field of the control word. When the mode field has this value, the µDMA controller does not perform any transfers and disables the channel if it is enabled. At the end of a transfer, the µDMA controller updates the control word to set the mode to Stop.

49.2.6.2 Basic Mode

In Basic mode, the µDMA controller performs transfers as long as there are more items to transfer, and a transfer request is present. This mode is used with peripherals that assert a µDMA request signal whenever the peripheral is ready for a data transfer. Basic mode should not be used in any situation where the request is momentary, even though the entire transfer should be completed. For example, a software-initiated transfer creates a momentary request, and in Basic mode, only the number of transfers specified by the ARBSIZE field in the DMA Channel Control Word (DMACHCTL) register is transferred on a software request, even if there is more data to transfer.

When all of the items have been transferred using Basic mode, the µDMA controller sets the mode for that channel to Stop.

BASIC mode can be programmed to ignore when XFERSIZE reaches 0x000 and continue copying on request until the channel is stopped manually. If the NXTUSEBURST bit in the µDMA Channel Control Word (DMACHCTL) register is set while in BASIC mode and the XFERSIZE reaches 0x000 and is not written back, transfers continue until the request is deasserted by the peripheral.

49.2.6.3 Auto Mode

Auto mode is similar to Basic mode, except that once a transfer request is received, the transfer runs to completion, even if the µDMA request is removed. This mode is suitable for software-triggered transfers. Generally, Auto mode is not used with a peripheral.

When all the items have been transferred using Auto mode, the µDMA controller sets the mode for that channel to Stop.

49.2.6.4 Ping-Pong

Ping-Pong mode is used to support a continuous data flow to or from a peripheral. To use Ping-Pong mode, both the primary and alternate data structures must be implemented. Both structures are set up by the processor for data transfer between memory and a peripheral. The transfer is started using the primary control structure. When the transfer using the primary control structure is complete, the µDMA controller reads the alternate control structure for that channel to continue the transfer. Each time this happens, an interrupt is generated, and the processor can reload the control structure for the just-completed transfer. Data flow can continue indefinitely this way, using the primary and alternate control structures to switch back and forth between buffers as the data flows to or from the peripheral.

For an example showing operation in Ping-Pong mode, see Figure 49-2.

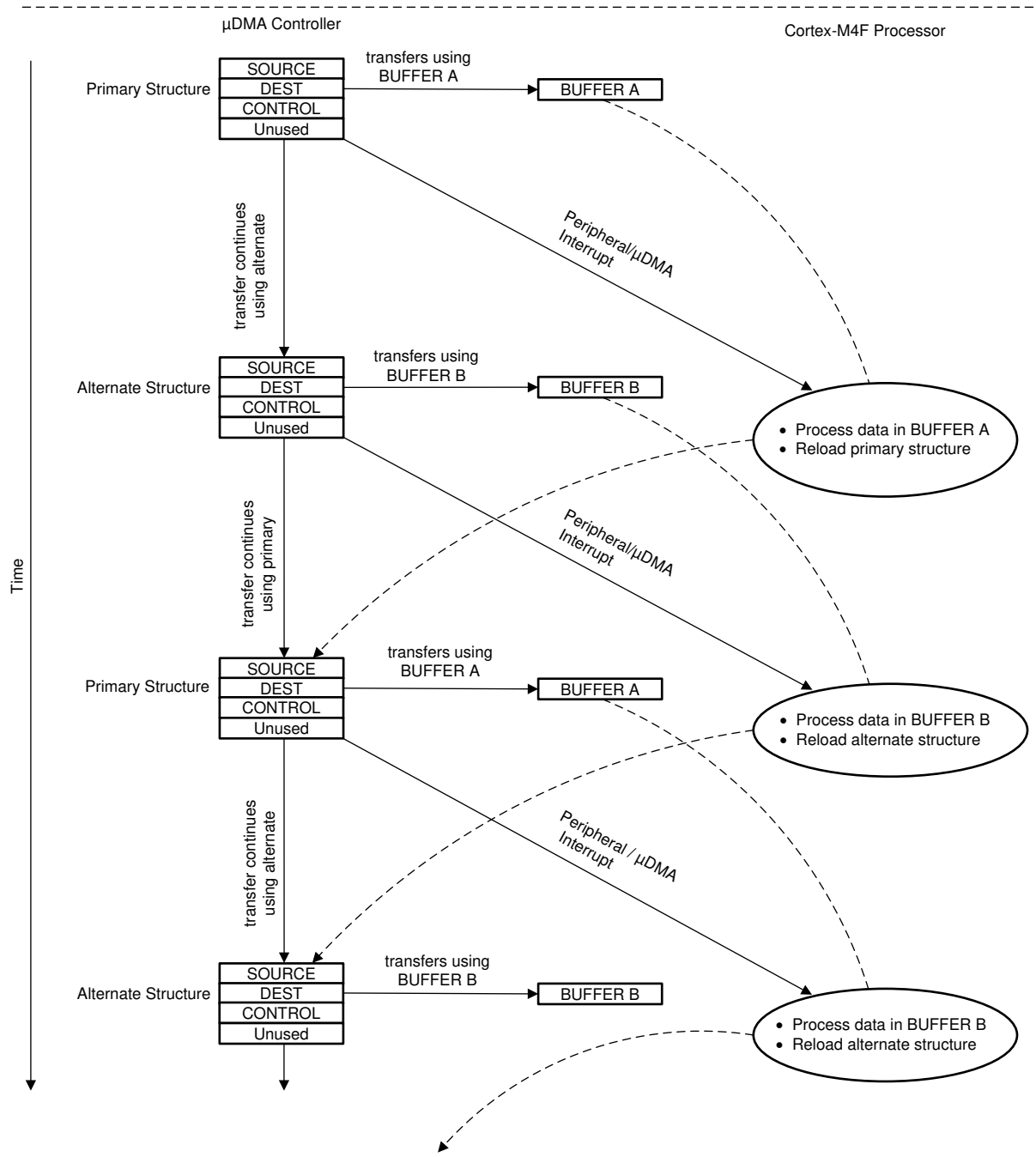


Figure 49-2. Example of Ping-Pong µDMA Transaction

49.2.6.5 Memory Scatter-Gather

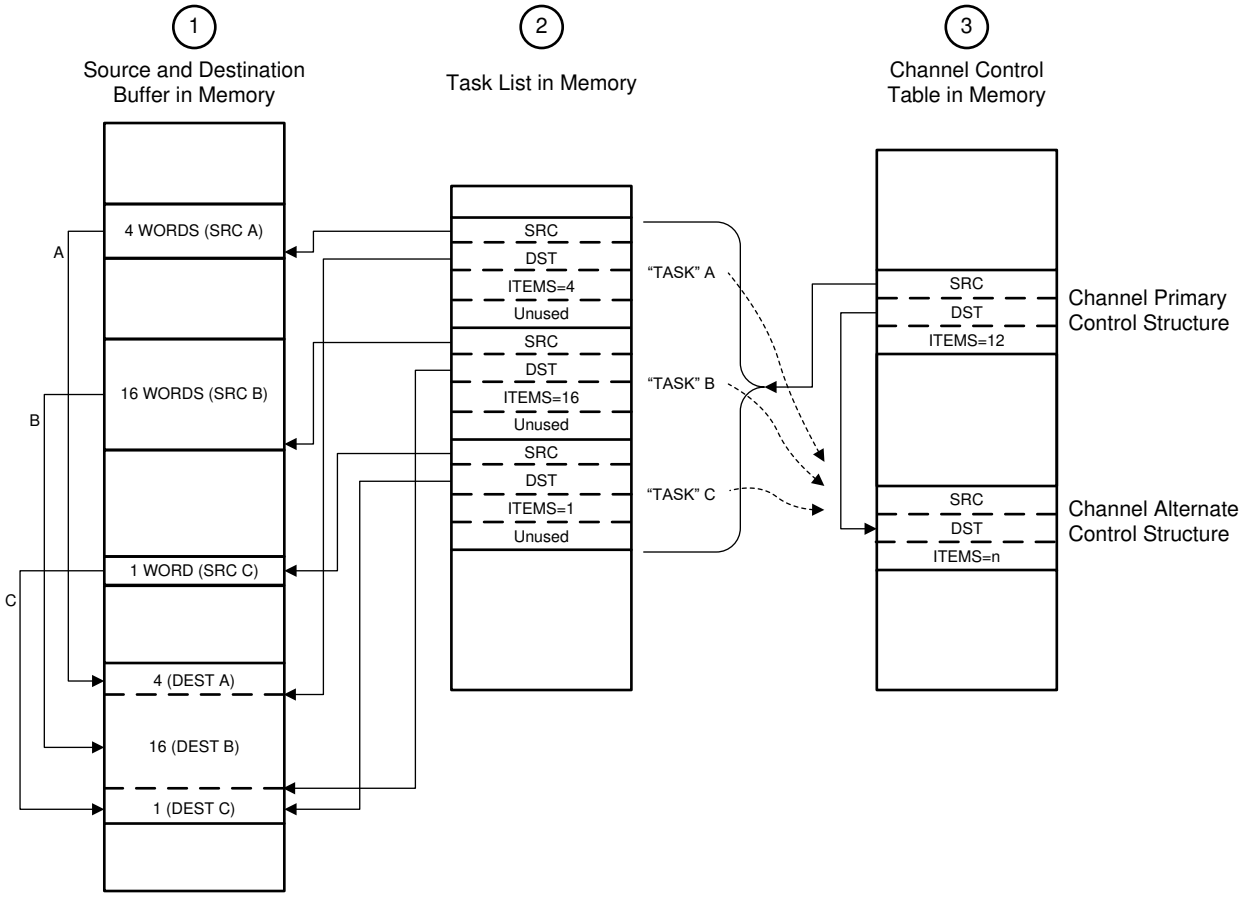
Memory Scatter-Gather mode is a complex mode used when data must be transferred to or from varied locations in memory instead of a set of contiguous locations in a memory buffer. For example, a gather µDMA operation could be used to selectively read the payload of several stored packets of a communication protocol and store them together in sequence in a memory buffer.

In Memory Scatter-Gather mode, the primary control structure is used to program the alternate control structure from a table in memory. The table is set up by the processor software and contains a list of control structures, each containing the source and destination end pointers, and the control word for a specific transfer. The mode of each control word must be set to Scatter-Gather mode. Each entry in the table is copied in turn to the alternate structure where it is then executed. The µDMA controller alternates between using the primary control structure to copy the next transfer instruction from the list and then executing the new transfer instruction. The end of the list is marked by programming the control word for the last entry to use Auto transfer mode. Once the last transfer is performed using Auto mode, the µDMA controller stops. A completion interrupt is generated only after the last transfer. It is possible to loop the list by having the last entry copy the primary control structure to point back to the beginning of the list (or to a new list). It is also possible to trigger a set of other channels to perform a transfer, either directly, by programming a write to the software trigger for another channel, or indirectly, by causing a peripheral action that results in a µDMA request.

By programming the µDMA controller using this method, a set of up to 256 arbitrary transfers can be performed based on a single µDMA request.

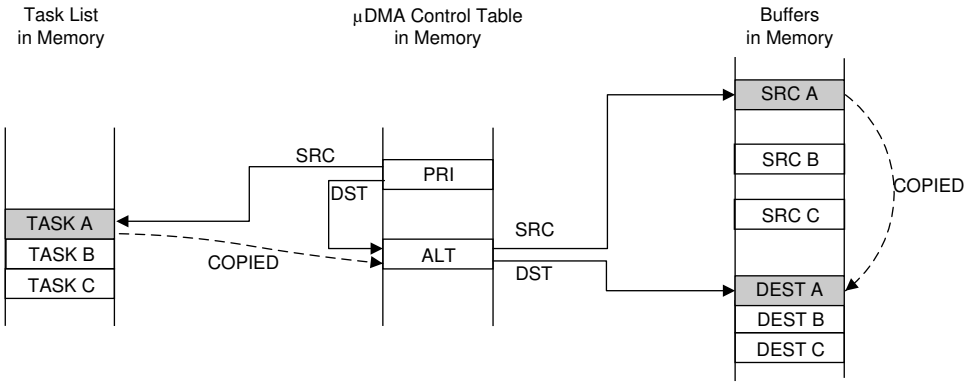
For an example of operation in Memory Scatter-Gather mode, see Figure 49-3 and Figure 49-4. This example shows a gather operation, where data in three separate buffers in memory is copied together into one buffer. Figure 49-3 shows how the application sets up a µDMA task list in memory that is used by the controller to perform three sets of copy operations from different locations in memory. The primary control structure for the channel that is used for the operation is configured to copy from the task list to the alternate control structure.

Figure 49-4 shows the sequence as the µDMA controller performs the three sets of copy operations. First, using the primary control structure, the µDMA controller loads the alternate control structure with task A. It then performs the copy operation specified by task A, copying the data from the source buffer A to the destination buffer. Next, the µDMA controller again uses the primary control structure to load task B into the alternate control structure, and then performs the B operation with the alternate control structure. The process is repeated for task C.



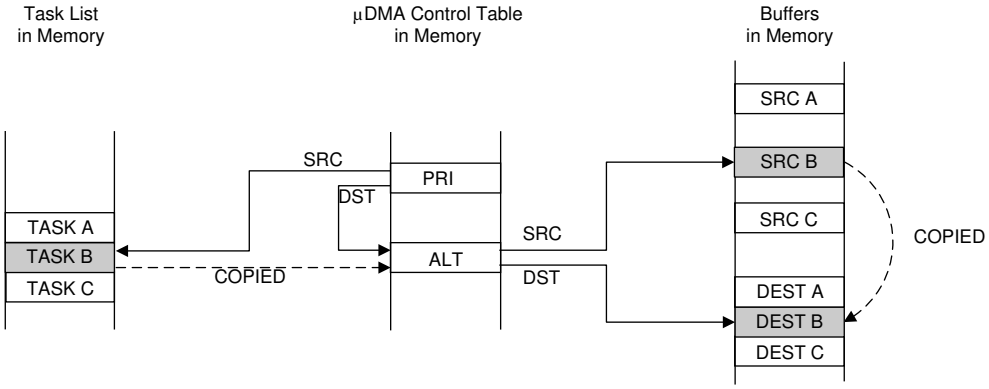
NOTES:
1. Application needs to copy data items from three separate locations in memory into one combined buffer.
2. Application sets up µDMA task list in memory, which contains the pointers and control configuration for three µDMA copy tasks.
3. Application sets up the channel primary control structure to copy each task configuration, one at a time, to the alternate control structure, where it is executed by the µDMA controller.
4. The SRC and DST pointers in the task list must point to the last location in the corresponding buffer.

Figure 49-3. Memory Scatter-Gather, Setup and Configuration



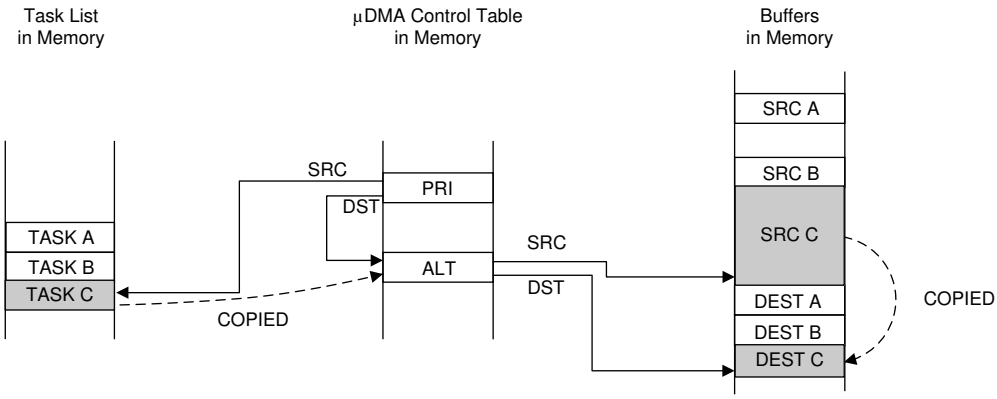
Using the channel's primary control structure, the µDMA controller copies task A configuration to the channel's alternate control structure.

Then, using the channel's alternate control structure, the µDMA controller copies data from the source buffer A to the destination buffer.



Using the channel's primary control structure, the µDMA controller copies task B configuration to the channel's alternate control structure.

Then, using the channel's alternate control structure, the µDMA controller copies data from the source buffer B to the destination buffer.



Using the channel's primary control structure, the µDMA controller copies task C configuration to the channel's alternate control structure.

Then, using the channel's alternate control structure, the µDMA controller copies data from the source buffer C to the destination buffer.

Figure 49-4. Memory Scatter-Gather, µDMA Copy Sequence

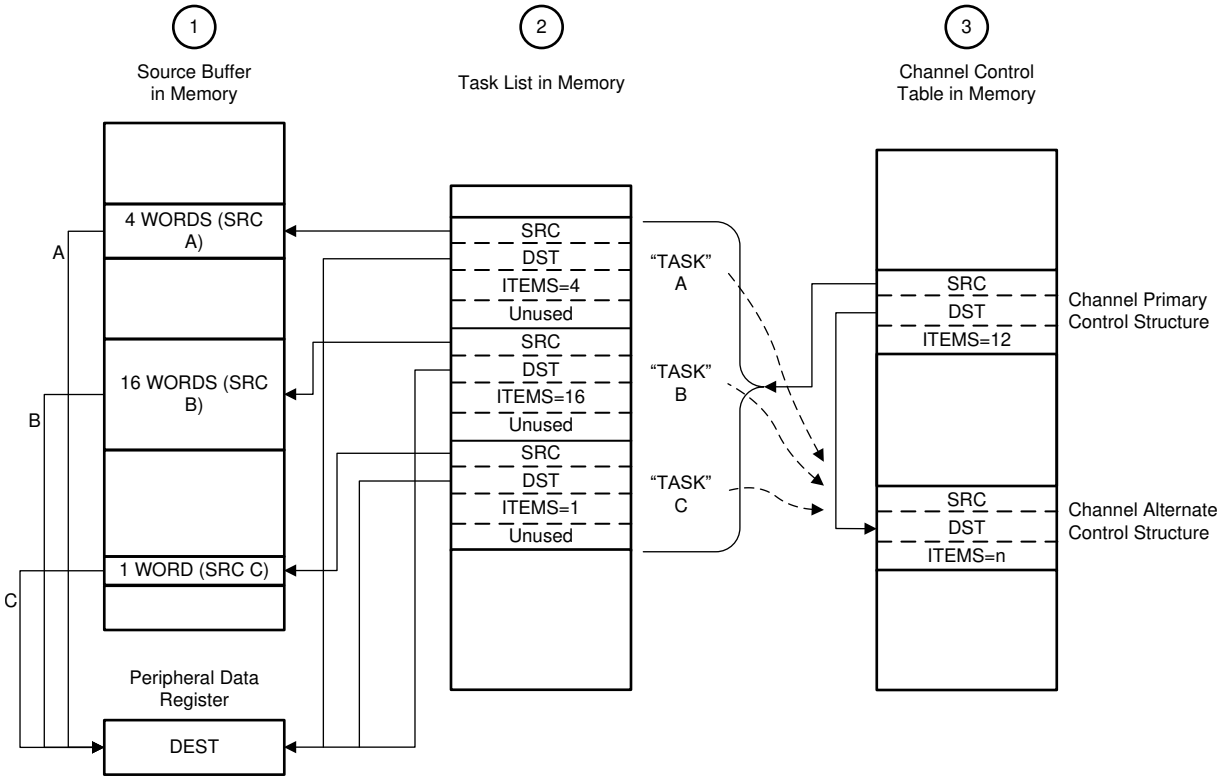
49.2.6.6 Peripheral Scatter-Gather

Peripheral Scatter-Gather mode is very similar to Memory Scatter-Gather, except that the transfers are controlled by a peripheral making a µDMA request. Upon detecting a request from the peripheral, the µDMA controller uses the primary control structure to copy one entry from the list to the alternate control structure and then performs the transfer. At the end of this transfer, the primary control structure will copy the next task to the alternate control structure . If the next task is a memory-to-memory transfer, execution will start immediately and run to completion; if the next task is a peripheral-type transfer, the µDMA will wait for a peripheral request to begin.

By using this method, the µDMA controller can transfer data to or from a peripheral from a set of arbitrary locations whenever the peripheral is ready to transfer data.

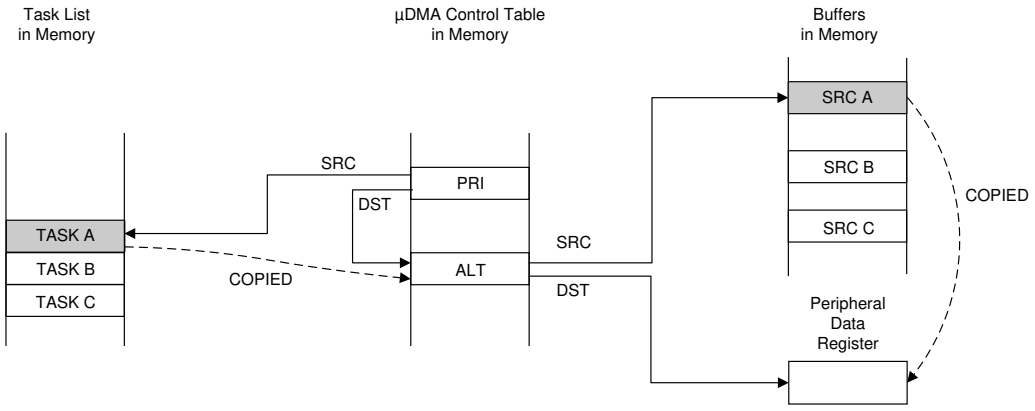
For an example of operation in Peripheral Scatter-Gather mode, see Figure 49-5 and Figure 49-6. This example shows a gather operation, where data from three separate buffers in memory is copied to a single peripheral data register. Figure 49-5 shows how the application sets up a µDMA task list in memory that is used by the controller to perform three sets of copy operations from different locations in memory. The primary control structure for the channel that is used for the operation is configured to copy from the task list to the alternate control structure.

Figure 49-6 shows the sequence as the µDMA controller performs the three sets of copy operations. First, using the primary control structure, the µDMA controller loads the alternate control structure with task A. It then performs the copy operation specified by task A, copying the data from the source buffer A to the peripheral data register. Next, the µDMA controller again uses the primary control structure to load task B into the alternate control structure, and then performs the B operation with the alternate control structure. The process is repeated for task C.



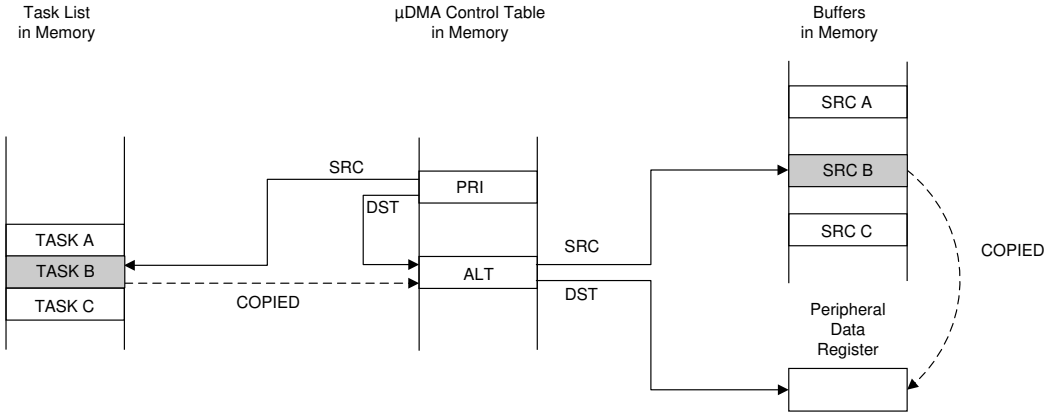
NOTES:
1. Application has a need to copy data items from three separate locations in memory into a peripheral data register.
2. Application sets up µDMA "task list" in memory, which contains the pointers and control configuration for three µDMA copy "tasks."
3. Application sets up the channel primary control structure to copy each task configuration, one at a time, to the alternate control structure, where it is executed by the µDMA controller.

Figure 49-5. Peripheral Scatter-Gather, Setup and Configuration



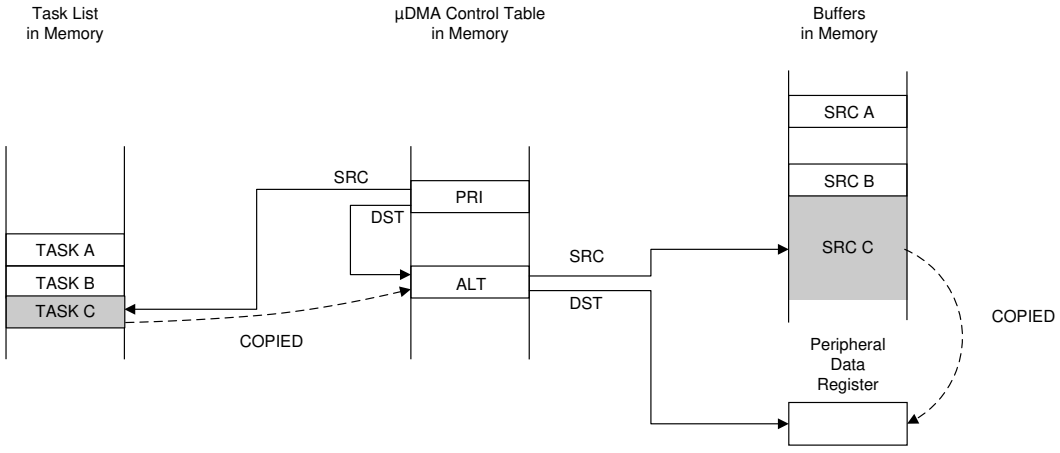
Using the channel's primary control structure, the µDMA controller copies task A configuration to the channel's alternate control structure.

Then, using the channel's alternate control structure, the µDMA controller copies data from the source buffer A to the peripheral data register.



Using the channel's primary control structure, the µDMA controller copies task B configuration to the channel's alternate control structure.

Then, using the channel's alternate control structure, the µDMA controller copies data from the source buffer B to the peripheral data register.



Using the channel's primary control structure, the µDMA controller copies task C configuration to the channel's alternate control structure.

Then, using the channel's alternate control structure, the µDMA controller copies data from the source buffer C to the peripheral data register.

Figure 49-6. Peripheral Scatter-Gather, µDMA Copy Sequence

49.2.7 Transfer Size and Increment

The µDMA controller supports transfer data sizes of 8, 16, or 32 bits. The source and destination data size must be the same for any given transfer. The source and destination address can be auto-incremented by bytes, halfwords, or words, or can be set to no increment. The source and destination address increment values can be set independently, and it is not necessary for the address increment to match the data size as long as the increment is the same or larger than the data size. For example, it is possible to perform a transfer using 8-bit data size, but using an address increment of full words (4 bytes). The data to be transferred must be aligned in memory according to the data size (8, 16, or 32 bits).

Table 49-5 shows the configuration to read from a peripheral that supplies 8-bit data.

Table 49-5. µDMA Read Example: 8-Bit Peripheral

Field	Configuration
Source data size	8 bits
Destination data size	8 bits
Source address increment	No increment
Destination address increment	Byte
Source end pointer	Peripheral read FIFO register
Destination end pointer	End of the data buffer in memory

49.2.8 Peripheral Interface

There are two main classes of µDMA-connected peripherals:

- Peripherals with FIFOs serviced by the µDMA to transmit or receive data.
- Peripherals that provide trigger inputs to the µDMA

49.2.8.1 FIFO Peripherals

FIFO peripherals contain a FIFO of data to be sent and a FIFO of data that has been received. The µDMA controller is used to transfer data between these FIFOs and system memory. For example, when a UART FIFO contains one or more entries, a single transfer request is sent to the µDMA for processing. If this request has not been processed and the UART FIFO reaches the interrupt FIFO level set in the UART Interrupt FIFO Level Select (UARTIFLS) register, another interrupt is sent to the µDMA which is higher priority than the single-transfer request. In this instance, an ARBSIZ transfer is performed as configured in the DMACHCTL register. After the transfer is complete, the DMA sends a receive or transmit complete interrupt to the UART Raw Interrupt Status (UARTRIS) register.

If the FIFO peripheral's SETn bit is set in the DMA Channel Useburst Set (DMAUSEBURSTSET) register, then the µDMA will only perform transfers defined by the ARBSIZ bit field in the DMACHCTL register for better bus utilization. For peripherals that tend to transmit and receive in bursts, such as the UART, we recommend against the use of this configuration since it could cause the tail end of transmissions to stick in the FIFO.

49.2.8.2 Trigger Peripherals

Certain peripherals, such as the general purpose timer, trigger an interrupt to the µDMA controller when a programmed event occurs. When a trigger event occurs, the µDMA executes a transfer defined by the ARBSIZ bit field in the DMACHCTL register. If only a single transfer is needed for a µDMA trigger, then the ARBSIZ field is set to 0x1.

If the trigger peripheral generates another µDMA request while the prior one is being serviced and that particular channel is the highest priority asserted channel, the second request will be processed as soon as the handling of the first is complete. If two additional trigger peripheral µDMA requests are generated prior to the completion of the first, the third request is lost.

49.2.9 Software Request

A transfer is initiated by software by first configuring and enabling the transfer, and then issuing a software request using the DMA Channel Software Request (DMASWREQ) register. For software-based transfers, the Auto transfer mode should be used.

It is possible to initiate a transfer on any available software channel using the DMASWREQ register. If a request is initiated by software using a peripheral µDMA channel, then the completion interrupt occurs on the interrupt vector for the peripheral instead of the software interrupt vector. Any peripheral channel may be used for software requests as long as the corresponding peripheral is not using µDMA for data transfer.

Note

Channels designated in the table as only "software" are dedicated software channels. When only one software request is required in an application, dedicated software channels can be used. If multiple software requests in code are required, then peripheral channel software requests should be used for proper µDMA completion acknowledgement.

49.2.10 Interrupts and Errors

Depending on the peripheral, the µDMA can indicate transfer completion at the end of an entire transfer or when a FIFO or buffer reaches a certain level (Table 49-2 and the individual peripheral chapters). When a µDMA transfer is complete, a dma_done signal is sent to the peripheral that initiated the µDMA event. Interrupts can be enabled within the peripheral to trigger on µDMA transfer completion. For more information on peripheral µDMA interrupts, see the individual peripheral chapters. If the transfer uses the software µDMA channel, then the completion interrupt occurs on the dedicated software µDMA interrupt vector (see Table 49-6).

If the µDMA controller encounters a bus or memory protection error as it attempts to perform a data transfer, it disables the µDMA channel that caused the error and generates an interrupt on the µDMA error interrupt vector. The processor can read the DMA Bus Error Clear (DMAERRCLR) register to determine if an error is pending. The ERRCLR bit is set if an error occurred. The error can be cleared by writing a 1 to the ERRCLR bit.

Table 49-6 shows the dedicated interrupt assignments for the µDMA controller.

Table 49-6. µDMA Interrupt Assignments

Interrupt	Assignment
41	µDMA software channel transfer
42	µDMA error

49.3 Initialization and Configuration

49.3.1 Module Initialization

Before the µDMA controller can be used, it must be enabled in the System Control block and in the peripheral. The location of the channel control structure must also be programmed.

The following steps should be performed one time during system initialization:

1. Enable the µDMA clock using the CMPCLKCR2 register.
2. Enable the µDMA controller by setting the MASTEN bit of the DMA Configuration (DMACFG) register.
3. Program the location of the channel control table by writing the base address of the table to the DMA Channel Control Base Pointer (DMACTLBASE) register. The base address must be aligned on a 1024-byte boundary.

49.3.2 Configuring a Memory-to-Memory Transfer

µDMA channel 30 is dedicated for software-initiated transfers. However, any channel can be used for software-initiated, memory-to-memory transfer if the associated peripheral is not being used.

49.3.2.1 Configure the Channel Attributes

First, configure the channel attributes:

1. Program bit 30 of the DMA Channel Priority Set (DMAPRIOSET) or DMA Channel Priority Clear (DMAPRIOCLR) registers to set the channel to High priority or Default priority.
2. Set bit 30 of the DMA Channel Primary Alternate Clear (DMAALTCLR) register to select the primary channel control structure for this transfer.
3. Set bit 30 of the DMA Channel Useburst Clear (DMAUSEBURSTCLR) register to allow the µDMA controller to respond to single and burst requests.
4. Set bit 30 of the DMA Channel Request Mask Clear (DMAREQMASKCLR) register to allow the µDMA controller to recognize requests for this channel.

49.3.2.2 Configure the Channel Control Structure

Now the channel control structure must be configured.

This example transfers 256 words from one memory buffer to another. Channel 30 is used for a software transfer, and the control structure for channel 30 is at offset 0x1E0 of the channel control table. The channel control structure for channel 30 is located at the offsets shown in Table 49-7.

Table 49-7. Channel Control Structure Offsets for Channel 30

Offset	Description
Control Table Base + 0x1E0	Channel 30 source end pointer
Control Table Base + 0x1E4	Channel 30 destination end pointer
Control Table Base + 0x1E8	Channel 30 control word

49.3.2.2.1 Configure the Source and Destination

The source and destination end pointers must be set to the last address for the transfer (inclusive).

1. Program the source end pointer at offset 0x1E0 to the address of the source buffer + 0x3FC.
2. Program the destination end pointer at offset 0x1E4 to the address of the destination buffer + 0x3FC.

The control word at offset 0x1E8 must be programmed according to Table 49-8.

Table 49-8. Channel Control Word Configuration for Memory Transfer Example

Field in DMACHCTL	Bits	Value	Description
DSTINC	31:30	2	32-bit destination address increment
DSTSIZE	29:28	2	32-bit destination data size
SRCINC	27:26	2	32-bit source address increment
SRCSIZE	25:24	2	32-bit source data size
reserved	23:22	0	Reserved
DSTPROT0(1)	21	0	Privileged access protection for destination data writes
reserved	20:19	0	Reserved
SRCPROT0(1)	18	0	Privileged access protection for source data reads
ARBSIZE	17:14	3	Arbitrates after 8 transfers
XFERSIZE	13:4	255	Transfer 256 items
NXTUSEBURST	3	0	N/A for this transfer type
XFERMODE	2:0	2	Use Auto-request transfer mode

(1) The value of this bit must be 1 (privileged) for AES accesses.

49.3.2.2.2 Configure Peripheral Interrupts

For memory-to-memory transfers, the peripheral involved must be configured to generate an interrupt when the µDMA has completed its transfer. Upon completion, the µDMA will send a dma_done signal to the peripheral.

49.3.2.3 Start the Transfer

Now the channel is configured and is ready to start.

1. Enable the channel by setting bit 30 of the DMA Channel Enable Set (DMAENASET) register.
2. Issue a transfer request by setting bit 30 of the DMA Channel Software Request (DMASWREQ) register.

The µDMA transfer begins. If the interrupt is enabled, then the processor is notified by interrupt when the transfer is complete. If needed, the status can be checked by reading bit 30 of the DMAENASET register. This bit is automatically cleared when the transfer is complete. The status can also be checked by reading the XFERMODE field of the channel control word at offset 0x1E8. This field is automatically cleared at the end of the transfer.

49.3.3 Configuring a Peripheral for Simple Transmit

This example configures the µDMA controller to transmit a buffer of data to a peripheral. The peripheral has a transmit FIFO with a trigger level of 4. The example peripheral uses µDMA channel 7.

49.3.3.1 Configure the Channel Attributes

First, configure the channel attributes:

1. Configure bit 7 of the DMA Channel Priority Set (DMAPRIOSET) or DMA Channel Priority Clear (DMAPRIOCLR) registers to set the channel to High priority or Default priority.
2. Set bit 7 of the DMA Channel Primary Alternate Clear (DMAALTCLR) register to select the primary channel control structure for this transfer.
3. Set bit 7 of the DMA Channel Useburst Clear (DMAUSEBURSTCLR) register to allow the µDMA controller to respond to single and burst requests.
4. Set bit 7 of the DMA Channel Request Mask Clear (DMAREQMASKCLR) register to allow the µDMA controller to recognize requests for this channel.

49.3.3.2 Configure the Channel Control Structure

This example transfers 64 bytes from a memory buffer to the peripheral's transmit FIFO register using µDMA channel 7. The control structure for channel 7 is at offset 0x070 of the channel control table. The channel control structure for channel 7 is located at the offsets shown in Table 49-9.

Table 49-9. Channel Control Structure Offsets for Channel 7

Offset	Description
Control Table Base + 0x070	Channel 7 source end pointer
Control Table Base + 0x074	Channel 7 destination end pointer
Control Table Base + 0x078	Channel 7 control word

49.3.3.2.1 Configure the Source and Destination

The source and destination end pointers must be set to the last address for the transfer (inclusive). Because the peripheral pointer does not change, it simply points to the peripheral's data register.

1. Program the source end pointer at offset 0x070 to the address of the source buffer + 0x3F.
2. Program the destination end pointer at offset 0x074 to the address of the peripheral's transmit FIFO register.

The control word at offset 0x078 must be programmed according to Table 49-10.

Table 49-10. Channel Control Word Configuration for Peripheral Transmit Example

Field in DMACHCTL	Bits	Value	Description
DSTINC	31:30	3	Destination address does not increment
DSTSIZE	29:28	0	8-bit destination data size
SRCINC	27:26	0	8-bit source address increment
SRCSIZE	25:24	0	8-bit source data size
reserved	23:22	0	Reserved
DSTPROT0(1)	21	0	Privileged access protection for destination data writes
reserved	20:19	0	Reserved
SRCPROT0(1)	18	0	Privileged access protection for source data reads
ARBSIZE	17:14	2	Arbitrates after 4 transfers
XFERSIZE	13:4	63	Transfer 64 items
NXTUSEBURST	3	0	N/A for this transfer type
XFERMODE	2:0	1	Use Basic transfer mode

(1) The value of this bit must be 1 (privileged) for AES accesses.

Note

In this example, it is not important if the peripheral makes a single request or a burst request. Because the peripheral has a FIFO that triggers at a level of 4, the arbitration size is set to 4. If the peripheral does make a burst request, then 4 bytes are transferred, which is what the FIFO can accommodate. If the peripheral makes a single request (if there is any space in the FIFO), then one byte is transferred at a time. If it is important to the application that transfers only be made in bursts, then the Channel Useburst SET[7] bit should be set in the DMA Channel Useburst Set (DMAUSEBURSTSET) register.

49.3.3.3 Start the Transfer

Now the channel is configured and is ready to start.

1. Enable the channel by setting bit 7 of the DMA Channel Enable Set (DMAENASET) register.

The µDMA controller is now configured for transfer on channel 7. The controller makes transfers to the peripheral whenever the peripheral asserts a µDMA request. The transfers continue until the entire buffer of 64 bytes has been transferred. When that happens, the µDMA controller disables the channel and sets the XFERMODE field of the channel control word to 0 (Stopped). The status of the transfer can be checked by reading bit 7 of the DMA Channel Enable Set (DMAENASET) register. This bit is automatically cleared when the transfer is complete. The status can also be checked by reading the XFERMODE field of the channel control word at offset 0x078. This field is automatically cleared at the end of the transfer.

If peripheral interrupts are enabled, then the peripheral generates an interrupt when the entire transfer is complete.

49.3.4 Configuring a Peripheral for Ping-Pong Receive

This example configures the µDMA controller to continuously receive 8-bit data from a peripheral into a pair of 64-byte buffers. The peripheral has a receive FIFO with a trigger level of 8. The example peripheral uses µDMA channel 8.

49.3.4.1 Configure the Channel Attributes

First, configure the channel attributes:

1. Configure bit 8 of the DMA Channel Priority Set (DMAPRIOSET) or DMA Channel Priority Clear (DMAPRIOCLR) registers to set the channel to High priority or Default priority.
2. Set bit 8 of the DMA Channel Primary Alternate Clear (DMAALTCLR) register to select the primary channel control structure for this transfer.
3. Set bit 8 of the DMA Channel Useburst Clear (DMAUSEBURSTCLR) register to allow the µDMA controller to respond to single and burst requests.
4. Set bit 8 of the DMA Channel Request Mask Clear (DMAREQMASKCLR) register to allow the µDMA controller to recognize requests for this channel.

49.3.4.2 Configure the Channel Control Structure

This example transfers bytes from the peripheral's receive FIFO register into two memory buffers of 64 bytes each. As data is received, when one buffer is full, the µDMA controller switches to use the other.

To use Ping-Pong buffering, both primary and alternate channel control structures must be used. The primary control structure for channel 8 is at offset 0x080 of the channel control table, and the alternate channel control structure is at offset 0x280. The channel control structures for channel 8 are located at the offsets shown in Table 49-11.

Table 49-11. Primary and Alternate Channel Control Structure Offsets for Channel 8

Offset	Description
Control Table Base + 0x080	Channel 8 primary source end pointer
Control Table Base + 0x084	Channel 8 primary destination end pointer
Control Table Base + 0x088	Channel 8 primary control word
Control Table Base + 0x280	Channel 8 alternate source end pointer
Control Table Base + 0x284	Channel 8 alternate destination end pointer
Control Table Base + 0x288	Channel 8 alternate control word

49.3.4.2.1 Configure the Source and Destination

The source and destination end pointers must be set to the last address for the transfer (inclusive). Because the peripheral pointer does not change, it simply points to the peripheral's data register. Both the primary and alternate sets of pointers must be configured.

1. Program the primary source end pointer at offset 0x080 to the address of the peripheral's receive buffer.
2. Program the primary destination end pointer at offset 0x084 to the address of ping-pong buffer A + 0x3F.
3. Program the alternate source end pointer at offset 0x280 to the address of the peripheral's receive buffer.
4. Program the alternate destination end pointer at offset 0x284 to the address of ping-pong buffer B + 0x3F.

The primary control word at offset 0x088 and the alternate control word at offset 0x288 are initially programmed the same way.

1. Program the primary channel control word at offset 0x088 according to Table 49-12.
2. Program the alternate channel control word at offset 0x288 according to Table 49-12.

Table 49-12. Channel Control Word Configuration for Peripheral Ping-Pong Receive Example

Field in DMACHCTL	Bits	Value	Description
DSTINC	31:30	0	8-bit destination address increment
DSTSIZE	29:28	0	8-bit destination data size
SRCINC	27:26	3	Source address does not increment
SRCSIZE	25:24	0	8-bit source data size
reserved	23:22	0	Reserved
DSTPROT0	21	0	Privileged access protection for destination data writes
reserved	20:19	0	Reserved
SRCPROT0	18	0	Privileged access protection for source data reads
ARBSIZE	17:14	3	Arbitrates after 8 transfers
XFERSIZE	13:4	63	Transfer 64 items
NXTUSEBURST	3	0	N/A for this transfer type
XFERMODE	2:0	3	Use Ping-Pong transfer mode

Note

In this example, it is not important if the peripheral makes a single request or a burst request. Because the peripheral has a FIFO that triggers at a level of 8, the arbitration size is set to 8. If the peripheral does make a burst request, then 8 bytes are transferred, which is what the FIFO can accommodate. If the peripheral makes a single request (if there is any data in the FIFO), then one byte is transferred at a time. If it is important to the application that transfers only be made in bursts, then the Channel Useburst SET[8] bit should be set in the DMA Channel Useburst Set (DMAUSEBURSTSET) register.

49.3.4.3 Configure and Enable the Peripheral Interrupt

An interrupt handler should be configured when using ping-pong mode. However, ping-pong mode can be configured without interrupts by polling. The interrupt handler is triggered after each buffer is complete.

1. Configure and enable an interrupt handler for the peripheral. Now the channel is configured and is ready to start.
2. Enable the channel by setting bit 8 of the DMA Channel Enable Set (DMAENASET) register.

49.3.4.4 Process Interrupts

The µDMA controller is now configured and enabled for transfer on channel 8. When the peripheral asserts the µDMA request signal, the µDMA controller makes transfers into buffer A using the primary channel control structure. When the primary transfer to buffer A is complete, it switches to the alternate channel control structure and makes transfers into buffer B. At the same time, the primary channel control word mode field is configured to indicate Stopped, and an interrupt is generated in the peripheral's raw interrupt status register .

When an interrupt is triggered, the interrupt handler must determine which buffer is complete and process the data or set a flag that the data must be processed by non-interrupt buffer processing code. Then the next buffer transfer must be set up.

In the interrupt handler:

1. Read the primary channel control word at offset 0x088 and check the XFERMODE field. If the field is 0, this means buffer A is complete. If buffer A is complete, then:
   - Process the newly received data in buffer A or signal the buffer processing code that buffer A has data available.
   - Reprogram the primary channel control word at offset 0x88 according to Table 49-12.
2. Read the alternate channel control word at offset 0x288 and check the XFERMODE field. If the field is 0, this means buffer B is complete. If buffer B is complete, then:
   - Process the newly received data in buffer B or signal the buffer processing code that buffer B has data available.
   - Reprogram the alternate channel control word at offset 0x288 according to Table 49-12.

49.3.5 Configuring Channel Assignments

Channel assignments for each µDMA channel can be changed using the DMACHMAPn registers. Each 4-bit field represents a µDMA channel. For channel assignments, see Table 49-1.

For example, to use UART0 RX on channel 8, configure the CH8SEL bit in the DMACHMAP1 register to be 0. If a peripheral is enabled on two different channels, the µDMA channel that has the highest priority for that peripheral takes precedence.

49.4 Software

49.4.1 UDMA Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/DEVICE_GPN/examples/CORE_IF_MULTICORE/udma

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

49.4.1.1 uDMA RAM to RAM transfer - CM

FILE: udma_ex1_software_trigger.c

This example showcases how to use uDMA to transfer data from one RAM location to another using software trigger.

This configures the UDMA in AUTO mode and transfers 32-bit words from one location to another using Channel 30. Software trigger is being used here. Once the transfer is completed, a check of the validity of the data will be performed in the uDMA ISR.

External Connections

- None

Watch Variables

- *srcData* - Source
- *destData* - Destination
- *errCount* - Error count

49.4.1.2 uDMA RAM to RAM transfer - CM

FILE: udma_ex2_scatter_gather_mode.c

This example showcases how to configure uDMA in memory scatter-gather mode and transfer data from varied locations in memory rather than a set of contiguous locations in a memory buffer.

This example creates an array of structs of length 20. The struct includes a header element and a data element, each of length 10. The UDMA is configured to tranfer only the data element from all the 20 data packets.

External Connections

- None

Watch Variables

- *packets* - 'data' element contains the actual data to be transferred
- *consolidatedData* - Destination buffer where the data is transferred to
- *errCount* - Error count

49.5 µDMA Registers

This section describes the Connectivity Manager Micro Direct Memory Access Registers.

49.5.1 µDMA Base Addresses

Table 49-13. UDMA Base Address Table (CM)

DriverLib Name	Base Address	uDMA Access	Ethernet DMA Access
UDMA_BASE	0x400F_F000	-	-

49.5.2 UDMAREGS Registers

Table 49-14 lists the memory-mapped registers for the UDMAREGS registers. All register offset addresses not listed in Table 49-14 should be considered as reserved locations and the register contents should not be modified.

Table 49-14. UDMAREGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	DMASTAT	DMA Status		Go
4h	DMACFG	DMA Configuration		Go
8h	DMACTLBASE	DMA Channel Control Base Pointer		Go
Ch	DMAALTBASE	DMA Alternate Channel Control Base Pointer		Go
14h	DMASWREQ	DMA Channel Software Request		Go
18h	DMAUSEBURSTSET	DMA Channel Useburst Set		Go
1Ch	DMAUSEBURSTCLR	DMA Channel Useburst Clear		Go
20h	DMAREQMASKSET	DMA Channel Request Mask Set		Go
24h	DMAREQMASKCLR	DMA Channel Request Mask Clear		Go
28h	DMAENASET	DMA Channel Enable Set		Go
2Ch	DMAENACLR	DMA Channel Enable Clear		Go
30h	DMAALTSET	DMA Channel Primary Alternate Set		Go
34h	DMAALTCLR	DMA Channel Primary Alternate Clear		Go
38h	DMAPRIOSET	DMA Channel Priority Set		Go
3Ch	DMAPRIOCLR	DMA Channel Priority Clear		Go
4Ch	DMAERRCLR	DMA Bus Error Clear		Go
510h	DMACHMAP0	DMA Channel Map Select 0		Go
514h	DMACHMAP1	DMA Channel Map Select 1		Go
518h	DMACHMAP2	DMA Channel Map Select 2		Go
51Ch	DMACHMAP3	DMA Channel Map Select 3		Go
FD0h	DMAPeriphID4	DMA Peripheral Identification 4		Go
FE0h	DMAPeriphID0	DMA Peripheral Identification 0		Go
FE4h	DMAPeriphID1	DMA Peripheral Identification 1		Go
FE8h	DMAPeriphID2	DMA Peripheral Identification 2		Go
FECh	DMAPeriphID3	DMA Peripheral Identification 3		Go
FF0h	DMAPCellID0	DMA PrimeCell Identification 0		Go
FF4h	DMAPCellID1	DMA PrimeCell Identification 1		Go
FF8h	DMAPCellID2	DMA PrimeCell Identification 2		Go
FFCh	DMAPCellID3	DMA PrimeCell Identification 3		Go

Complex bit access types are encoded to fit into small table cells. Table 49-15 shows the codes that are used for access types in this section.

Table 49-15. UDMAREGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write

Table 49-15. UDMAREGS Access Type Codes (continued)

Access Type	Code	Description
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

49.5.2.1 DMASTAT Register (Offset = 0h) [Reset = 001F0000h]

DMASTAT is shown in Figure 49-7 and described in Table 49-16.

Return to the Summary Table.

DMA Status

Figure 49-7. DMASTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED			DMACHANS				
R-0h			R-1Fh				
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
STATE				RESERVED			MASTEN
R-0h				R-0h			R-0h

Table 49-16. DMASTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-21	RESERVED	R	0h	Reserved
20-16	DMACHANS	R	1Fh	Available DMA Channels Minus 1 This field contains a value equal to the number of DMA channels the DMA controller is configured to use, minus one. The value of 0x1F corresponds to 32 DMA channels. Reset type: CM.SYSRESETn
15-8	RESERVED	R	0h	Reserved
7-4	STATE	R	0h	Control State Machine Status This field shows the current status of the control state machine. Status can be one of the following. Value Description 0x0 Idle 0x1 Reading channel controller data. 0x2 Reading source end pointer. 0x3 Reading destination end pointer. 0x4 Reading source data. 0x5 Writing destination data. 0x6 Waiting for ÂuDMA request to clear. 0x7 Writing channel controller data. 0x8 Stalled 0x9 Done 0xA-0xF Undefined Reset type: CM.SYSRESETn
3-1	RESERVED	R	0h	Reserved
0	MASTEN	R	0h	Master Enable Status Value Description 0 The DMA controller is disabled. 1 The DMA controller is enabled. Reset type: CM.SYSRESETn

49.5.2.2 DMACFG Register (Offset = 4h) [Reset = X]

DMACFG is shown in Figure 49-8 and described in Table 49-17.

Return to the Summary Table.

DMA Configuration

Figure 49-8. DMACFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							MASTEN
R-0h							R-0/W-X

Table 49-17. DMACFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	MASTEN	R-0/W	X	Controller Master Enable Value Description 0 Disables the DMA controller. 1 Enables DMA controller. Reset type: CM.SYSRESETn

49.5.2.3 DMACTLBASE Register (Offset = 8h) [Reset = 0h]

DMACTLBASE is shown in Figure 49-9 and described in Table 49-18.

Return to the Summary Table.

DMA Channel Control Base Pointer

Figure 49-9. DMACTLBASE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																						RESERVED									
R/W-0h																						R-0h									

Table 49-18. DMACTLBASE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	ADDR	R/W	0h	Channel Control Base Address This field contains the pointer to the base address of the channel control table. The base address must be 1024-byte aligned. Reset type: CM.SYSRESETn
9-0	RESERVED	R	0h	Reserved

49.5.2.4 DMAALTBASE Register (Offset = Ch) [Reset = 200h]

DMAALTBASE is shown in Figure 49-10 and described in Table 49-19.

Return to the Summary Table.

DMA Alternate Channel Control Base Pointer

Figure 49-10. DMAALTBASE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R-200h																															

Table 49-19. DMAALTBASE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R	200h	Alternate Channel Address Pointer This field provides the base address of the alternate channel control structures. Reset type: CM.SYSRESETn

49.5.2.5 DMASWREQ Register (Offset = 14h) [Reset = X]

DMASWREQ is shown in Figure 49-11 and described in Table 49-20.

Return to the Summary Table.

DMA Channel Software Request

Figure 49-11. DMASWREQ Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWREQ																															
R-0/W-X																															

Table 49-20. DMASWREQ Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SWREQ	R-0/W	X	Channel [n] Software Request These bits generate software requests. Bit 0 corresponds to channel 0. Value Description 1 Generate a software request for the corresponding channel. 0 No request generated. These bits are automatically cleared when the software request has been completed. Reset type: CM.SYSRESETn

49.5.2.6 DMAUSEBURSTSET Register (Offset = 18h) [Reset = 0h]

DMAUSEBURSTSET is shown in Figure 49-12 and described in Table 49-21.

Return to the Summary Table.

DMA Channel Useburst Set

Figure 49-12. DMAUSEBURSTSET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SET																															
R/W-0h																															

Table 49-21. DMAUSEBURSTSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SET	R/W	0h	Channel [n] Useburst Set Value Description 0 DMA channel [n] responds to single or burst requests. 1 DMA channel [n] responds only to burst requests. Bit 0 corresponds to channel 0. This bit is automatically cleared as described above. A bit can also be manually cleared by setting the corresponding CLR[n] bit in the DMAUSEBURSTCLR register. Reset type: CM.SYSRESETn

49.5.2.7 DMAUSEBURSTCLR Register (Offset = 1Ch) [Reset = X]

DMAUSEBURSTCLR is shown in Figure 49-13 and described in Table 49-22.

Return to the Summary Table.

DMA Channel Useburst Clear

Figure 49-13. DMAUSEBURSTCLR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLR																															
R-0/W-X																															

Table 49-22. DMAUSEBURSTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLR	R-0/W	X	Channel [n] Useburst Clear Value Description 0 No effect. 1 Setting a bit clears the corresponding SET[n] bit in the DMAUSEBURSTSET register meaning that ÂuDMA channel [n] responds to single and burst requests. Reset type: CM.SYSRESETn

49.5.2.8 DMAREQMASKSET Register (Offset = 20h) [Reset = 0h]

DMAREQMASKSET is shown in Figure 49-14 and described in Table 49-23.

Return to the Summary Table.

DMA Channel Request Mask Set

Figure 49-14. DMAREQMASKSET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SET																															
R/W-0h																															

Table 49-23. DMAREQMASKSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SET	R/W	0h	Channel [n] Request Mask Set Value Description 0 The peripheral associated with channel [n] is enabled to request DMA transfers. 1 The peripheral associated with channel [n] is not able to request DMA transfers. Channel [n] may be used for software-initiated transfers. Bit 0 corresponds to channel 0. A bit can only be cleared by setting the corresponding CLR[n] bit in the DMAREQMASKCLR register. Reset type: CM.SYSRESETn

49.5.2.9 DMAREQMASKCLR Register (Offset = 24h) [Reset = X]

DMAREQMASKCLR is shown in Figure 49-15 and described in Table 49-24.

Return to the Summary Table.

DMA Channel Request Mask Clear

Figure 49-15. DMAREQMASKCLR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLR																															
R-0/W-X																															

Table 49-24. DMAREQMASKCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLR	R-0/W	X	Channel [n] Request Mask Clear Value Description 0 No effect. 1 Setting a bit clears the corresponding SET[n] bit in the DMAREQMASKSET register meaning that the peripheral associated with channel [n] is enabled to request DMA transfers. Reset type: CM.SYSRESETn

49.5.2.10 DMAENASET Register (Offset = 28h) [Reset = 0h]

DMAENASET is shown in Figure 49-16 and described in Table 49-25.

Return to the Summary Table.

DMA Channel Enable Set

Figure 49-16. DMAENASET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SET																															
R/W-0h																															

Table 49-25. DMAENASET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SET	R/W	0h	Channel [n] Enable Set Value Description 0 uDMA Channel [n] is disabled. 1 uDMA Channel [n] is enabled. Bit 0 corresponds to channel 0. A bit can only be cleared by setting the corresponding CLR[n] bit in the DMAENACLR register or when the end of a ÂuDMA transfer occurs. Reset type: CM.SYSRESETn

49.5.2.11 DMAENACLR Register (Offset = 2Ch) [Reset = X]

DMAENACLR is shown in Figure 49-17 and described in Table 49-26.

Return to the Summary Table.

DMA Channel Enable Clear

Figure 49-17. DMAENACLR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLR																															
R-0/W-X																															

Table 49-26. DMAENACLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLR	R-0/W	X	Clear Channel [n] Enable Clear Value Description 0 No effect. 1 Setting a bit clears the corresponding SET[n] bit in the DMAENASET register meaning that channel [n] is disabled for DMA transfers. The controller disables a channel when it completes the DMA cycle. Reset type: CM.SYSRESETn

49.5.2.12 DMAALTSET Register (Offset = 30h) [Reset = 0h]

DMAALTSET is shown in Figure 49-18 and described in Table 49-27.

Return to the Summary Table.

DMA Channel Primary Alternate Set

Figure 49-18. DMAALTSET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SET																															
R/W-0h																															

Table 49-27. DMAALTSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SET	R/W	0h	Channel [n] Alternate Set Value Description 0 ÂuDMA channel [n] is using the primary control structure. 1 ÂuDMA channel [n] is using the alternate control structure. Bit 0 corresponds to channel 0. A bit can only be cleared by setting the corresponding CLR[n] bit in the DMAALTCLR register. For Ping-Pong and Scatter-Gather cycle types, the ÂuDMA controller automatically sets these bits to select the alternate channel control data structure. Reset type: CM.SYSRESETn

49.5.2.13 DMAALTCLR Register (Offset = 34h) [Reset = X]

DMAALTCLR is shown in Figure 49-19 and described in Table 49-28.

Return to the Summary Table.

DMA Channel Primary Alternate Clear

Figure 49-19. DMAALTCLR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLR																															
R-0/W-X																															

Table 49-28. DMAALTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLR	R-0/W	X	Channel [n] Alternate Clear Value Description 0 No effect. 1 Setting a bit clears the corresponding SET[n] bit in the DMAALTSET register meaning that channel [n] is using the primary control structure. For Ping-Pong and Scatter-Gather cycle types, the ÂuDMA controller automatically sets these bits to select the alternate channel control data structure. Reset type: CM.SYSRESETn

49.5.2.14 DMAPRIOSET Register (Offset = 38h) [Reset = 0h]

DMAPRIOSET is shown in Figure 49-20 and described in Table 49-29.

Return to the Summary Table.

DMA Channel Priority Set

Figure 49-20. DMAPRIOSET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SET																															
R/W-0h																															

Table 49-29. DMAPRIOSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SET	R/W	0h	Channel [n] Priority Set Value Description 0 ÂuDMA channel [n] is using the default priority level. 1 ÂuDMA channel [n] is using a high priority level. Bit 0 corresponds to channel 0. A bit can only be cleared by setting the corresponding CLR[n] bit in the DMAPRIOCLR register. Reset type: CM.SYSRESETn

49.5.2.15 DMAPRIOCLR Register (Offset = 3Ch) [Reset = X]

DMAPRIOCLR is shown in Figure 49-21 and described in Table 49-30.

Return to the Summary Table.

DMA Channel Priority Clear

Figure 49-21. DMAPRIOCLR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLR																															
R-0/W-X																															

Table 49-30. DMAPRIOCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CLR	R-0/W	X	Channel [n] Priority Clear Value Description 0 No effect. 1 Setting a bit clears the corresponding SET[n] bit in the DMAPRIOSET register meaning that channel [n] is using the default priority level. Reset type: CM.SYSRESETn

49.5.2.16 DMAERRCLR Register (Offset = 4Ch) [Reset = 0h]

DMAERRCLR is shown in Figure 49-22 and described in Table 49-31.

Return to the Summary Table.

DMA Bus Error Clear

Figure 49-22. DMAERRCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ERRCLR
R-0h							R/W1S-0h

Table 49-31. DMAERRCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	ERRCLR	R/W1S	0h	DMA Bus Error Status Value Description 0 No bus error is pending. 1 A bus error is pending. This bit is cleared by writing a 1 to it. Reset type: CM.SYSRESETn

49.5.2.17 DMACHMAP0 Register (Offset = 510h) [Reset = 0h]

DMACHMAP0 is shown in Figure 49-23 and described in Table 49-32.

Return to the Summary Table.

DMA Channel Map Select 0

Figure 49-23. DMACHMAP0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH7SEL				CH6SEL				CH5SEL				CH4SEL			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3SEL				CH2SEL				CH1SEL				CH0SEL			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 49-32. DMACHMAP0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	CH7SEL	R/W	0h	DMA Channel 7 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
27-24	CH6SEL	R/W	0h	DMA Channel 6 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
23-20	CH5SEL	R/W	0h	DMA Channel 5 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
19-16	CH4SEL	R/W	0h	DMA Channel 4 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
15-12	CH3SEL	R/W	0h	DMA Channel 3 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
11-8	CH2SEL	R/W	0h	DMA Channel 2 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
7-4	CH1SEL	R/W	0h	DMA Channel 1 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
3-0	CH0SEL	R/W	0h	DMA Channel 0 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn

49.5.2.18 DMACHMAP1 Register (Offset = 514h) [Reset = 0h]

DMACHMAP1 is shown in Figure 49-24 and described in Table 49-33.

Return to the Summary Table.

DMA Channel Map Select 1

Figure 49-24. DMACHMAP1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH15SEL				CH14SEL				CH13SEL				CH12SEL			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH11SEL				CH10SEL				CH9SEL				CH8SEL			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 49-33. DMACHMAP1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	CH15SEL	R/W	0h	DMA Channel 15 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
27-24	CH14SEL	R/W	0h	DMA Channel 14 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
23-20	CH13SEL	R/W	0h	DMA Channel 13 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
19-16	CH12SEL	R/W	0h	DMA Channel 12 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
15-12	CH11SEL	R/W	0h	DMA Channel 11 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
11-8	CH10SEL	R/W	0h	DMA Channel 10 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
7-4	CH9SEL	R/W	0h	DMA Channel 9 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
3-0	CH8SEL	R/W	0h	DMA Channel 8 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn

49.5.2.19 DMACHMAP2 Register (Offset = 518h) [Reset = 0h]

DMACHMAP2 is shown in Figure 49-25 and described in Table 49-34.

Return to the Summary Table.

DMA Channel Map Select 2

Figure 49-25. DMACHMAP2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH23SEL				CH22SEL				CH21SEL				CH20SEL			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH19SEL				CH18SEL				CH17SEL				CH16SEL			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 49-34. DMACHMAP2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	CH23SEL	R/W	0h	DMA Channel 23 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
27-24	CH22SEL	R/W	0h	DMA Channel 22 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
23-20	CH21SEL	R/W	0h	DMA Channel 21 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
19-16	CH20SEL	R/W	0h	DMA Channel 20 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
15-12	CH19SEL	R/W	0h	DMA Channel 19 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
11-8	CH18SEL	R/W	0h	DMA Channel 18 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
7-4	CH17SEL	R/W	0h	DMA Channel 17 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
3-0	CH16SEL	R/W	0h	DMA Channel 16 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn

49.5.2.20 DMACHMAP3 Register (Offset = 51Ch) [Reset = 0h]

DMACHMAP3 is shown in Figure 49-26 and described in Table 49-35.

Return to the Summary Table.

DMA Channel Map Select 3

Figure 49-26. DMACHMAP3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH31SEL				CH30SEL				CH29SEL				CH28SEL			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH27SEL				CH26SEL				CH25SEL				CH24SEL			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 49-35. DMACHMAP3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	CH31SEL	R/W	0h	DMA Channel 31 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
27-24	CH30SEL	R/W	0h	DMA Channel 30 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
23-20	CH29SEL	R/W	0h	DMA Channel 29 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
19-16	CH28SEL	R/W	0h	DMA Channel 28 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
15-12	CH27SEL	R/W	0h	DMA Channel 27 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
11-8	CH26SEL	R/W	0h	DMA Channel 26 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
7-4	CH25SEL	R/W	0h	DMA Channel 25 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn
3-0	CH24SEL	R/W	0h	DMA Channel 24 Source Select See device spec for channel assignments. Reset type: CM.SYSRESETn

49.5.2.21 DMAPeriphID4 Register (Offset = FD0h) [Reset = 4h]

DMAPeriphID4 is shown in Figure 49-27 and described in Table 49-36.

Return to the Summary Table.

DMA Peripheral Identification 4

Figure 49-27. DMAPeriphID4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID4							
R-0h																								R-4h							

Table 49-36. DMAPeriphID4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID4	R	4h	DMA Peripheral ID Register Can be used by software to identify the presence of this peripheral. Reset type: CM.SYSRESETn

49.5.2.22 DMAPeriphID0 Register (Offset = FE0h) [Reset = 30h]

DMAPeriphID0 is shown in Figure 49-28 and described in Table 49-37.

Return to the Summary Table.

DMA Peripheral Identification 0

Figure 49-28. DMAPeriphID0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID0							
R-0h																								R-30h							

Table 49-37. DMAPeriphID0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID0	R	30h	DMA Peripheral ID Register [7:0] Can be used by software to identify the presence of this peripheral. Reset type: CM.SYSRESETn

49.5.2.23 DMAPeriphID1 Register (Offset = FE4h) [Reset = B2h]

DMAPeriphID1 is shown in Figure 49-29 and described in Table 49-38.

Return to the Summary Table.

DMA Peripheral Identification 1

Figure 49-29. DMAPeriphID1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID1							
R-0h																								R-B2h							

Table 49-38. DMAPeriphID1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID1	R	B2h	DMA Peripheral ID Register [15:8] Can be used by software to identify the presence of this peripheral. Reset type: CM.SYSRESETn

49.5.2.24 DMAPeriphID2 Register (Offset = FE8h) [Reset = Bh]

DMAPeriphID2 is shown in Figure 49-30 and described in Table 49-39.

Return to the Summary Table.

DMA Peripheral Identification 2

Figure 49-30. DMAPeriphID2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID2							
R-0h																								R-Bh							

Table 49-39. DMAPeriphID2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID2	R	Bh	DMA Peripheral ID Register [23:16] Can be used by software to identify the presence of this peripheral. Reset type: CM.SYSRESETn

49.5.2.25 DMAPeriphID3 Register (Offset = FECh) [Reset = 0h]

DMAPeriphID3 is shown in Figure 49-31 and described in Table 49-40.

Return to the Summary Table.

DMA Peripheral Identification 3

Figure 49-31. DMAPeriphID3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								PID3							
R-0h																								R-0h							

Table 49-40. DMAPeriphID3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	PID3	R	0h	DMA Peripheral ID Register [31:24] Can be used by software to identify the presence of this peripheral. Reset type: CM.SYSRESETn

49.5.2.26 DMAPCellID0 Register (Offset = FF0h) [Reset = Dh]

DMAPCellID0 is shown in Figure 49-32 and described in Table 49-41.

Return to the Summary Table.

DMA PrimeCell Identification 0

Figure 49-32. DMAPCellID0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CID0							
R-0h																								R-Dh							

Table 49-41. DMAPCellID0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	CID0	R	Dh	DMA PrimeCell ID Register [7:0] Provides software a standard cross-peripheral identification system. Reset type: CM.SYSRESETn

49.5.2.27 DMAPCellID1 Register (Offset = FF4h) [Reset = F0h]

DMAPCellID1 is shown in Figure 49-33 and described in Table 49-42.

Return to the Summary Table.

DMA PrimeCell Identification 1

Figure 49-33. DMAPCellID1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CID1							
R-0h																								R-F0h							

Table 49-42. DMAPCellID1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	CID1	R	F0h	DMA PrimeCell ID Register [15:8] Provides software a standard cross-peripheral identification system. Reset type: CM.SYSRESETn

49.5.2.28 DMAPCellID2 Register (Offset = FF8h) [Reset = 5h]

DMAPCellID2 is shown in Figure 49-34 and described in Table 49-43.

Return to the Summary Table.

DMA PrimeCell Identification 2

Figure 49-34. DMAPCellID2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CID2							
R-0h																								R-5h							

Table 49-43. DMAPCellID2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	CID2	R	5h	DMA PrimeCell ID Register [23:16] Provides software a standard cross-peripheral identification system. Reset type: CM.SYSRESETn

49.5.2.29 DMAPCellID3 Register (Offset = FFCh) [Reset = B1h]

DMAPCellID3 is shown in Figure 49-35 and described in Table 49-44.

Return to the Summary Table.

DMA PrimeCell Identification 3

Figure 49-35. DMAPCellID3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								CID3							
R-0h																								R-B1h							

Table 49-44. DMAPCellID3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	CID3	R	B1h	DMA PrimeCell ID Register [31:24] Provides software a standard cross-peripheral identification system. Reset type: CM.SYSRESETn

49.5.3 UDMACHDES Registers

Table 49-45 lists the memory-mapped registers for the UDMACHDES registers. All register offset addresses not listed in Table 49-45 should be considered as reserved locations and the register contents should not be modified.

Table 49-45. UDMACHDES Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	DMASRCENDP	DMA Channel Source Address End Pointer		Go
4h	DMADSTENDP	DMA Channel Destination Address End Pointer		Go
8h	DMACHCTL	DMA Channel Control Word		Go

Complex bit access types are encoded to fit into small table cells. Table 49-46 shows the codes that are used for access types in this section.

Table 49-46. UDMACHDES Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

49.5.3.1 DMASRCENDP Register (Offset = 0h) [Reset = X]

DMASRCENDP is shown in Figure 49-36 and described in Table 49-47.

Return to the Summary Table.

DMA Channel Source Address End Pointer

Figure 49-36. DMASRCENDP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R/W-X																															

Table 49-47. DMASRCENDP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R/W	X	Source Address End Pointer This field points to the last address of the DMA transfer source (inclusive). If the source address is not incrementing (the SRCINC field in the DMACHCTL register is 0x3), then this field points at the source location itself (such as a peripheral data register). Reset type: CM.SYSRESETn

49.5.3.2 DMADSTENDP Register (Offset = 4h) [Reset = X]

DMADSTENDP is shown in Figure 49-37 and described in Table 49-48.

Return to the Summary Table.

DMA Channel Destination Address End Pointer

Figure 49-37. DMADSTENDP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R/W-X																															

Table 49-48. DMADSTENDP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R/W	X	Destination Address End Pointer This field points to the last address of the DMA transfer destination (inclusive). If the destination address is not incrementing (the DSTINC field in the DMACHCTL register is 0x3), then this field points at the destination location itself (such as a peripheral data register). Reset type: CM.SYSRESETn

49.5.3.3 DMACHCTL Register (Offset = 8h) [Reset = X]

DMACHCTL is shown in Figure 49-38 and described in Table 49-49.

Return to the Summary Table.

DMA Channel Control Word

Figure 49-38. DMACHCTL Register

31	30	29	28	27	26	25	24
DSTINC		DSTSIZE		SRCINC		SRCSIZE	
R/W-X		R/W-X		R/W-X		R/W-X	
23	22	21	20	19	18	17	16
RESERVED		DSTPROT0	RESERVED		SRCPROT0	ARBSIZE	
R-0h		R/W-0h	R-0h		R/W-0h	R/W-X	
15	14	13	12	11	10	9	8
ARBSIZE		XFERSIZE					
R/W-X		R/W-X					
7	6	5	4	3	2	1	0
XFERSIZE				NXTUSEBURST	XFERMODE		
R/W-X				R/W-X	R/W-X		

Table 49-49. DMACHCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	DSTINC	R/W	X	Destination Address Increment This field configures the destination address increment. The address increment value must be equal or greater than the value of the destination size (DSTSIZE). Value Description 0x0 ByteIncrement by 8-bit locations 0x1 Half-wordIncrement by 16-bit locations 0x2 WordIncrement by 32-bit locations 0x3 No incrementAddress remains set to the value of the Destination Address End Pointer (DMADSTENDP) for the channel Reset type: CM.SYSRESETn
29-28	DSTSIZE	R/W	X	Destination Data Size This field configures the destination item data size. DSTSIZE must be the same as SRCSIZE. Value Description 0x0 Byte8-bit data size 0x1 Half-word16-bit data size 0x2 Word32-bit data size 0x3 Reserved Reset type: CM.SYSRESETn
27-26	SRCINC	R/W	X	Source Address Increment This field configures the source address increment. The address increment value must be equal or greater than the value of the source size (SRCSIZE). Value Description 0x0 ByteIncrement by 8-bit locations 0x1 Half-wordIncrement by 16-bit locations 0x2 WordIncrement by 32-bit locations 0x3 No incrementAddress remains set to the value of the Source Address End Pointer (DMASRCENDP) for the channel Reset type: CM.SYSRESETn

Table 49-49. DMACHCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-24	SRCSIZE	R/W	X	Source Data Size This field configures the source item data size. DSTSIZE must be the same as SRCSIZE. Value Description 0x0 Byte8-bit data size. 0x1 Half-word16-bit data size. 0x2 Word32-bit data size. 0x3 Reserved Reset type: CM.SYSRESETn
23-22	RESERVED	R	0h	Reserved
21	DSTPROT0	R/W	0h	Destination Privilege Access This bit controls the privilege access protection for destination data writes. Value Description 0 The access is non-privileged. 1 The access is privileged. Reset type: CM.SYSRESETn
20-19	RESERVED	R	0h	Reserved
18	SRCPROT0	R/W	0h	Source Privilege Access This bit controls the privilege access protection for source data reads. Value Description 0 The access is non-privileged. 1 The access is privileged. Reset type: CM.SYSRESETn
17-14	ARBSIZE	R/W	X	Arbitration Size This field configures the number of transfers that can occur before the DMA controller re-arbitrates. The possible arbitration rate configurations represent powers of 2 and are shown below. Value Description 0x0 1 TransferArbitrates after each DMA transfer 0x1 2 Transfers 0x2 4 Transfers 0x3 8 Transfers 0x4 16 Transfers 0x5 32 Transfers 0x6 64 Transfers 0x7 128 Transfers 0x8 256 Transfers 0x9 512 Transfers 0xA-0xF 1024 TransfersIn this configuration, no arbitration occurs during the DMA transfer because the maximum transfer size is 1024. Reset type: CM.SYSRESETn
13-4	XFERSIZE	R/W	X	Transfer Size (minus 1) This field configures the total number of items to transfer. The value of this field is 1 less than the number to transfer (value 0 means transfer 1 item). The maximum value for this 10-bit field is 1023 which represents a transfer size of 1024 items. The transfer size is the number of items, not the number of bytes. If the data size is 32 bits, then this value is the number of 32-bit words to transfer. The DMA controller updates this field immediately prior to entering the arbitration process, so it contains the number of outstanding items that is necessary to complete the DMA cycle. Reset type: CM.SYSRESETn

Table 49-49. DMACHCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	NXTUSEBURST	R/W	X	Next Useburst This field controls whether the Useburst SET[n] bit is automatically set for the last transfer of a peripheral scatter-gather operation. Normally, for the last transfer, if the number of remaining items to transfer is less than the arbitration size, the DMA controller uses single transfers to complete the transaction. If this bit is set, then the controller uses a burst transfer to complete the last transfer. Reset type: CM.SYSRESETn
2-0	XFERMODE	R/W	X	DMA Transfer Mode This field configures the operating mode of the DMA cycle. Refer to for a detailed explanation of transfer modes. Because this register is in system RAM, it has no reset value. Therefore, this field should be initialized to 0 before the channel is enabled. Value Description 0x0 Stop 0x1 Basic 0x2 Auto-Request 0x3 Ping-Pong 0x4 Memory Scatter-Gather 0x5 Alternate Memory Scatter-Gather 0x6 Peripheral Scatter-Gather 0x7 Alternate Peripheral Scatter-Gather Reset type: CM.SYSRESETn

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from July 8, 2022 to September 27, 2023 (from Revision D (July 2022) to Revision E (September 2023)) **Page**

- Changed Name of Vector ID 17 in Table 3-3 153
- Changed Section 3.7.2.2 167
- Changed Section 3.7.2.4 168
- Added last paragraph in Section 3.7.3.3 168
- Added Section 3.7.7 174
- Moved Section 3.7.7.1 174
- Changed last paragraph in Section 3.12.1.8.7 187
- Added last sentence in Section 3.13 192
- Added Section 3.13.1 192
- Changed paragraph in Section 5.7.1.1 719
- Added Section 8.3.2 976
- Added Section 8.5.4 988
- Added list of Instructions in Section 8.7.3 998
- Changed Note in Section 9.3.3 1198
- Changed Figure 9-9 1198
- Changed Figure 9-10 1200
- Changed Table 9-6 1200
- Added Caution in Section 11.2.1 1350
- Added Section 14.7 1589
- Added Note in Section 24.5.6 2825
- Added Note before Immediate Load Mode in Section 26.5.3 2905
- Changed Figure 26-22 2910
- Changed Figure 26-71 2966
- Changed Step 2 in Section 26.15.1.5.4 2987
- Added Section 27.9 3174
- Corrected COMPL1 to COMPL2 for Lower Threshold 2 (LLT2) Comparator bullet in Section 28.8.2 3239
- Added last sentence in first paragraph in Section 30.1.3.4 3356
- Added Figure 30-2 3356
- Added SII to Table 31-1 3460
- Changed Section 31.2.7 3478
- Added "Externally triggered frame generation" in Section 32.1.2 3538
- Added Section 32.3.11 3569
- Changed Figure 37-2 3947
- Changed Table 43-3 4555
- Changed Figure 43-3 4555
- Changed Table 45-3 5093
- Added Figure 45-9 5104

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265